ROBUST REINFORCEMENT LEARNING: THEORY AND ALGORITHMS

A Dissertation

by

KISHAN PANAGANTI BADRINATH

Submitted to the Graduate and Professional School of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Dileep Kalathil
Committee Members,	P. R. Kumar
	Bani Mallick
	Srinivas Shakkottai
Head of Department,	Costas Georghiades

August 2023

Major Subject: Electrical Engineering

ABSTRACT

This research dissertation explores novel algorithms in the field of robust reinforcement learning (RL) that address the challenges of controlling dynamical systems in real-world scenarios. Classical reinforcement learning is a powerful sub-field in machine learning for training intelligent sequential decision-making agents in complex environments. However, these algorithms often face challenges when it comes to uncertainties and variations in the environment, as well as the requirement for a large number of training samples. In this work, we present novel robust reinforcement learning algorithms that address these challenges. Our algorithms focus on robustness to uncertainties in the environment through the transition dynamics variations. By leveraging techniques such as distributionally robust optimization, our algorithms aim to learn policies that can withstand these uncertainties. We study robust reinforcement learning in online environment interactions setting as well as when we are given historical without access to the environment. We also study the problems of imitation learning and offline reinforcement learning that is relevant for real-world applications where the goals differ from robust reinforcement learning, but we see the tools of distributionally robust optimization and model pessimism involves crucial roles in helping improve these areas of learning domain. The experimental results demonstrate the effectiveness of our robust algorithms, showcasing their potential for real-world applications where uncertainties and variations are prevalent.

DEDICATION

In wholehearted appreciation to my parents, whose unconditional love and sacrifices have made my educational pursuits possible. This dissertation stands as my gratitude for your unwavering support and belief in me.

ACKNOWLEDGMENTS

I extend my heartfelt gratitude to my advisor, Dr. Dileep Kalathil, for his exceptional mentorship, unwavering generosity, and support throughout my PhD. This dissertation would not have come to fruition without his invaluable guidance, extensive knowledge, infectious enthusiasm, and continuous support. I am immensely thankful for the insightful feedback provided by my committee members, Dr. P. R. Kumar, Dr. Bani Mallick, and Dr. Srinivas Shakkottai, which significantly contributed to the improvement of my research and this dissertation.

I am indebted to Dr. Mohammad Ghavamzadeh and Zaiyan Xu, who coauthored with me for publications of my research, and their invaluable ideas and constructive comments helped improve my research which shaped into this dissertation. My sincere thanks go out to the members of the Learning and Emerging Networked Systems (LENS) Lab and members of Prof. P. R. Kumar's group at Texas A&M University, including (alphabetical) Akshay Sarvesh, Akshay Mete, Amit Jena, Archana Bura, Aria HasanzadeZonuzy, Desik Rengarajan, Jaewon Kim, Khaled Nakhleh, Nitin Ragothaman, Prabhasa Kalkur, Ruida Zhou, Sapana Chaudhary, Sarat Bobbili, Tao Liu, Ujwal Dinesha, and, many more members met over the years, for fostering a conducive environment for intellectual discussions and making my PhD journey truly enjoyable.

During my academic journey, I had the privilege of completing my Masters in Communication and Networks at the Department of Electrical Communication Engineering, Indian Institute of Science (IISc), under the invaluable guidance of Dr. Rajesh Sundaresan. It was during this transformative period that I was first introduced to the captivating realm of research. I extend my sincerest gratitude to Rajesh for illuminating the path and instilling in me a profound appreciation for the world of academic research. To my dear friends outside of work in India, Germany, USA, especially (random order) Karthik, Ganesh, Bhagyashree, Darshan, Dheeraj, Chiranth, Akash, Akshay, Anuj, Harsha, Pavan, Lohith, Yeshashwini, Sachin, Sharath, and more, for whom I consider myself fortunate to have such amazing individuals in my life.

I would like to express my deepest appreciation and heartfelt thanks to my parents, Badri-

nath and Indrani, for their unwavering support and boundless patience, particularly during the most challenging moments of this journey. I am particularly grateful for their understanding and willingness to shoulder additional responsibilities, ensuring that I had the freedom to pursue my dreams without any concerns about supporting our family. Their love and sacrifice have been a constant source of motivation and inspiration, and I am truly blessed to have them as my parents. I want to convey my heartfelt thanks also to my sisters, Bindu and Ashwini, and my brother-in-laws Narayan and Raghu, for their support to my parents when I was almost eight thousand miles away from home. Lastly, again, I am profoundly indebted to my father, Badrinath, thank you for being a role model during my formative years and inspiring me to explore exciting endeavors beyond work. Your unwavering support in all my endeavors has been exceptional, and I am grateful for the lengths you went to in providing my sisters and me with the best possible education. I would not have come this far without your own research experience in Botany that boosted me to pursue this research journey ending up with this dissertation.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Dr. Dileep Kalathil (advisor), Dr. Srinivas Shakkottai, and Dr. P.R. Kumar of the Department of Electrical and Computer Engineering, and Dr. Bani Mallick of the Department of Statistics.

The work in Chapter 2 was published in 2021 at a machine learning conference called International Conference on Machine Learning (ICML). The work in Chapter 3 was published in 2022 at a machine learning conference called Artificial Intelligence and Statistics (AISTATS). The work in Chapter 4 was published in 2022 at a machine learning conference called Neural Information Processing Systems (NeurIPS). The works in Chapters 5 and 6 are new contributions that are preprints at the time of publication of this dissertation. The works in Chapters 4 to 6 was performed in collaboration with Dr. Mohammad Ghavamzadeh, Senior Staff Research Scientist at Google Research at the time of publication of this dissertation, and the supporting experiments for these research works were conducted in collaboration with Zaiyan Xu.

All other work conducted for the dissertation was completed by the student independently.

Funding Sources

Graduate study was supported by a research assistantship from Texas A&M University. This work was supported in part by the National Science Foundation (NSF) grants NSF-CRII-CPS-1850206, NSF-CAREER-EPCN-2045783, and NSF ECCS 2038963. The views and opinions expressed in this material are solely those of the authors and do not necessarily represent the opinions, findings, or recommendations of the sponsoring agencies.

TABLE OF CONTENTS

Page

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

Many real-world problems require learning to actively control a dynamical system in a sequential way [6, 7], rather than making predictions like the state-of-the-art machine learning algorithms [8] on well-curated historical data. For example, agile robots such as Google's Robotics Benchmarks for Learning (ROBEL) [9] have to learn to make appropriate decisions to achieve desired objectives often in unknown environments in the real-world. The class of machine learning that addresses the problem of learning to control such systems is called Reinforcement Learning (RL). RL has recently acquired remarkable feats by solutions such as DeepMind's MuZero [10], Google's Dreamer [11], IBM's Watson Health [12], UC Berkeley's simulated robots [13], and many more. However, most of these success stories are limited to such simulated environments. Major RL algorithms [14, 15, 16] rely on exploring the dynamical systems which is considered dangerous [17, 18, 19] in the real-world systems scenario. These RL algorithms also suffer from the issue of epistemic uncertainty [20, 21, 22] that degrades their performance. This issue is coined as the *simulator-to-reality gap* [23, 24] in RL literature and resolutions to it are an active area of research in both academia [25, 26, 27] and industry [28, 29]. The goal of this research document is to develop novel algorithms that are robust to changes in the real-world, robust to transitioning from simulator to the real-world systems, improving the existing approaches through innovations imparting robust, and contributing to the artificial intelligence revolution by enabling intelligent systems designed and operated by RL.

In order to develop a control strategy/policy, reinforcement learning algorithms frequently need a lot of data samples. Because of this, training RL algorithms directly on real-world systems is costly and could be hazardous. They are often trained on a simulator (online RL) or using pregathered offline datasets to solve this challenge (offline RL). The offline dataset is often compiled from either historical measurements or a sophisticated simulation of the real-world system. The trained RL policy is subsequently put into use under the presumption that the simulator, offline data, or training environment accurately represents the model of the real-world system. This as-

sumption is frequently false for a variety of reasons [23, 25, 28], including the modeling approximation mistakes, changes in the real-world parameters over time, and potential hostile disruptions in the real-world. In addition to alterations in the testing environment's topography, weather, illumination, and obstacle densities, conventional simulator settings for a mobile robot's sensor noise, action latency, friction, and mass may differ from those of the actual real-world robot. Unfortunately, even little changes to the training and testing environments can cause the present RL control mechanisms to drastically fail [30, 31, 32].

Learning a policy that is resilient to model parameter mismatches between the training and testing environments is the aim of robust reinforcement learning (RL). The Robust Markov Decision Process (RMDP) framework is used to model the robust planning issue [33, 34]. The RMDP formulation takes into account a collection of models known as the "uncertainty set", as opposed to the traditional MDP, which only takes into account one model (the transition probability function). In this uncertainty set, the objective is to identify an optimal robust policy that operates at its best under the worst model. The robust MDP and robust RL issues are much more difficult than their non-robust counterparts due to the minimization across the uncertainty set.

We address the problem of model-free reinforcement learning for RMDPs with large state spaces in Chapter 2, our work [1]. Finding a policy that is resistant to parameter uncertainties resulting from the mismatch between the simulator model and real-world conditions is the aim of the RMDP framework. As a multi-step online model-free learning method for policy evaluation, we first propose the Robust Least Squares Policy Evaluation algorithm. Using stochastic approximation techniques, we demonstrate the convergence of this approach. The Robust Least Squares Policy Iteration (RLSPI) algorithm is then suggested as a method for discovering the near-optimal robust policy. We also provide a general weighted Euclidean norm bound on the error of the algorithm policy (its proximity to optimality). Finally, we demonstrate the performance of our RLSPI algorithm on some standard benchmark problems.

We examine the problem of model-based reinforcement learning for RMDPs in Chapter 3, our work [2]. Here, we focus on the tabular episodic learning environment, where the algorithm

has access to a generative model of the nominal (training) environment that the uncertainty set is based on. To address this problem for the uncertainty sets defined by three distinct metrics - total variation, chi-square, Kullback-Leibler - we present the Robust Empirical Value Iteration (REVI) algorithm. We demonstrate that our algorithm achieves a sample complexity of $\mathcal{O}(|\mathcal{S}|^2 |\mathcal{A}| H^4)$, which is uniformly better than the current findings by a factor of $|\mathcal{S}|$. Here, $|\mathcal{S}|$ denotes the number of states, $|\mathcal{A}|$ the number of actions, and H the horizon length. In our recent work [35], we provide improved sample complexity bounds to $\mathcal{O}(|\mathcal{S}| |\mathcal{A}| H^4)$. We also offer the Wasserstein uncertainty set's sample complexity, which is a first. We also demonstrate the performance of our algorithm on some standard benchmark problems in both works.

We propose a robust RL technique called Robust Fitted Q-Iteration (RFQI) in Chapter 4, our work [3], which solely makes use of an offline dataset to learn the best robust policy. Because the robust Bellman operator minimizes over all models, robust RL with offline data is far more difficult to implement than its non-robust equivalent. This creates difficulties for offline data gathering, model optimization, and unbiased estimation. In this study, we suggest a methodical strategy to address these difficulties, leading to our RFQI algorithm. We illustrate RFQI's improved performance on common benchmark issues and demonstrate that it learns a robust policy that is close to optimal under the usual assumptions.

In Chapter 5, we study Imitation Learning (IL) where an agent learns to mimic an expert demonstrator without additional online environment interactions. The agent is only provided with a dataset of state-action pairs from the expert, without any information about the true rewards from the environment. Behavioral cloning (BC), a supervised learning method, is commonly used in IL but can result in abnormal behavior (covariate shift issue) when the agent encounters states not seen by the expert. We propose Distributionally Robust Behavioral Cloning (DR-BC) algorithm: an algorithmic framework that utilizes the distributionally robust optimization (DRO) technique with BC to solve the covariate shift in the IL problem efficiently both in theory and in practice. In theory, from the adversarial nature of DRO picking the appropriate state visitation distributions, we show that DR-BC can provably combat the covariate shift issue in IL. Additionally, we also show

that DR-BC's performance is robust to the parameter uncertainties, due to the gap between the simulator and real-world, under the IL problem both in theory and in practice. We also demonstrate that practical implementation of our approach mitigates both covariate shift and model perturbations on benchmark MuJoCo continuous control tasks.

We propose an offline RL algorithm called Model-Pessimistic Q-Iteration (MPQI) in Chapter 6, which solely makes use of offline data to learn any comparator policy. We use the pessimism inbuild in robust RL formulation to be able to disincentivize unseen state-action pairs in the available offline data. In this study, we suggest a methodical strategy to address these difficulties, leading to our MPQI algorithm. We compare the theoretical performance of our MPQI algorithm with other state-of-the-art algorithms in terms of the provable suboptimality guarantees under the common data coverage assumption. We study both finite state space and large state space cases using tabular and linear architectures respectively but with finite actions.

Lastly, in Chapter 7, we summarize our research carried out in this dissertation. We also discuss about a new algorithm for robust RL based on policy gradient reinforcement learning algorithms, as a potential future work, that can handle large scale or high dimensional problems. We put forth all the important and open questions that needs addressing. We are excited to see that our current research and future directions opening more avenues to more robust RL solutions that are necessary to close the gap between simulator world and the real-world.

2. ROBUST REINFORCEMENT LEARNING USING LEAST SQUARES POLICY ITERATION WITH PROVABLE PERFORMANCE GUARANTEES*

In this chapter, we focus on developing robust reinforcement learning algorithm that uses stochastic approximation techniques.

2.1 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 [33, 34] 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 [33, 34, 36, 37, 38] and under the linear function approximation [39]. Algorithms for learning the optimal robust policy with provable guarantees have been proposed, both in the model-free [40] and model-based [41] 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

*Reprinted with permission from Kishan Panaganti, Dileep Kalathil, "Robust Reinforcement Learning using Least Squares Policy Iteration with Provable Performance Guarantees." International Conference on Machine Learning. PMLR, 2021.

using function approximation architectures, such as linear basis functions [42], reproducing kernel Hilbert spaces (RKHS) [43] and deep neural networks [44]. Recently, robust reinforcement learning problem has been addressed using deep RL methods [45, 46, 47, 48, 49]. 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 guarantees for many challenging problems in RL, including online model-free exploration [50, 43], imitation learning [51, 52], meta reinforcement learning [53, 54], and offline reinforcement learning [55, 56]. 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.

2.1.1 Related Work

RMDP formulation to address the parameter uncertainty problem was first proposed by [33] and [34]. [33] 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 [37], modified policy iteration [57], and more general uncertainty set [36]. 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 [39]. Though this work suggests a sampling based approach, a general model-free learning algorithm and analysis was not included. [40] 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. [48] introduced soft-robust actor-critic algorithms using neural networks, but does not provide any global convergence guarantees for the learned policy. [58] proposed a min-max game framework to address the robust learning problem focusing on the tabular setting. [59] proposed a kernel-based RL algorithm for finding the robust value function in a batch learning setting. [46] 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) [60, 61, 62], 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 [63], 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.

2.2 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 $\pi_P^* = \operatorname{argmax}_\pi V_{\pi,P}$.

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* [33]. 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 [33, 34]

$$V_\pi = \inf_{P \in \mathcal{P}} V_{\pi,P}, \quad V^* = \sup_\pi \inf_{P \in \mathcal{P}} V_{\pi,P}. \tag{2.1}$$

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 [40].

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 $\mathcal{U}_{s,a} := \{x \mid \|x\|_2 \leq r, \sum_{s \in \mathcal{S}} x_s = 0, -P^o_{s,a}(s') \leq x_{s'} \leq 1 - P^o_{s,a}(s'), \forall s' \in \mathcal{S}\}$, 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 2.3.2, we introduce an approximate uncertainty set without using this information.

We consider *robust Bellman operator for policy evaluation*, defined as [33]

$$T_\pi(V)(s) = r(s, \pi(s)) + \alpha \inf_{P \in \mathcal{P}} \sum_{s'} P_{s,\pi(s)}(s') V(s'), \tag{2.2}$$

a popular approach to solve (2.1). Using our characterization of the uncertainty set, we can rewrite (2.2) as

$$T_\pi(V)(s) = r(s,\pi(s)) + \alpha \sum_{s'} P^o_{s,\pi(s)}(s')V(s') + \alpha \inf_{U \in \mathcal{U}_{s,\pi(s)}} \sum_{s'} U_{s,\pi(s)}(s')V(s'). \tag{2.3}$$

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, (2.3) can be written in the matrix form as

$$T_\pi(V) = r_\pi + \alpha P^o_\pi V + \alpha \sigma_{U_\pi}(V). \tag{2.4}$$

It is known [33] 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,

$$T(V) = \max_\pi T_\pi(V). \tag{2.5}$$

It is also known [33] 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 $\mathcal{P}$.

2.3 Robust Least Squares Policy Evaluation

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

2.3.1 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 [64, 62].

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 $T_\pi^{(\lambda)} : \mathbb{R}^{|\mathcal{S}|} \to \mathbb{R}^{|\mathcal{S}|}$ is defined as

$$T_\pi^{(\lambda)}(V) = (1 - \lambda) \sum_{m=0}^{\infty} \lambda^m T_\pi^{m+1}(V). \tag{2.6}$$

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 (2.3) or (2.6) 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 [62]. In linear function approximation architectures, the value function is represented as the weighted sum of features as, $\bar{V}(s) = \varphi(s)^\top w, \forall s \in \mathcal{S}$, where $\varphi(s) = (\varphi_1(s), \varphi_2(s), \ldots, \varphi_L(s))^\top$ 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 $\varphi(s)^\top$. 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, [39] 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 [39]. Also, a similar assumption is used in proving the convergence of off-policy reinforcement learning algorithm [65]. 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,* $\Pi T^{(\lambda)}_\pi$ *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, $\Phi w_{k+1} = \Pi T^{(\lambda)}_\pi \Phi w_k$. Since $\Pi T^{(\lambda)}_\pi$ 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 $w_{k+1} = \arg\min_w \|\Phi w - \Pi T^{(\lambda)}_\pi \Phi w_k\|^2_d$. It can be shown that (details are given in the supplementary material), we can get a closed form solution for w_{k+1} as

$$w_{k+1} = w_k + (\Phi^\top D \Phi)^{-1} \Phi^\top D (T^{(\lambda)}_\pi \Phi w_k - \Phi w_k). \tag{2.7}$$

This is similar to the projected equation approach [62] in the non-robust setting. Even in the non-robust setting, iterations using the (2.7) is intractable for MDPs with large state space. Moreover, when the transition matrix is unknown, it is not feasible to use (2.7) 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 [61, 62] 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).

$$T_\pi^{(\lambda)}(V) = (1-\lambda)\sum_{m=0}^{\infty}\lambda^m\Big(\sum_{k=0}^{m}(\alpha P_\pi^o)^k r_\pi + (\alpha P_\pi^o)^{m+1}V + \alpha\sum_{k=0}^{m}(\alpha P_\pi^o)^k \sigma_{\mathcal{U}_\pi}(T_\pi^{(m-k)}V)\Big). \quad (2.8)$$

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.

2.3.2 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

$$\widetilde{T}_\pi^{(\lambda)}(V) = (1-\lambda)\sum_{m=0}^{\infty}\lambda^m\left[\sum_{t=0}^{m}(\alpha P_\pi^o)^t r_\pi + \alpha\sum_{t=0}^{m}(\alpha P_\pi^o)^t\,\sigma_{\widehat{\mathcal{U}}_\pi}(V) + (\alpha P_\pi^o)^{m+1}V\right]. \tag{2.9}$$

Note that even with $\widehat{\mathcal{U}}_\pi = \mathcal{U}_\pi$, (2.9) is different from (2.8). We will show that this clever approximation helps to overcome the challenges due to the nonliterary associated with (2.8).

However, we emphasize that (2.9) 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

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

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 $\lambda \in [0,1)$,*

$$\|\Pi\widetilde{T}_\pi^{(\lambda)}V_1 - \Pi\widetilde{T}_\pi^{(\lambda)}V_2\|_d \le c(\alpha,\beta,\rho,\lambda)\,\|V_1 - V_2\|_d, \tag{2.10}$$

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*,*

$$\|V_\pi - \Phi w_\pi\|_d \leq \frac{1}{1 - c(\alpha, \beta, \rho, \lambda)} \left(\|V_\pi - \Pi V_\pi\|_d + \frac{\beta\rho\|V_\pi\|_d}{1-\beta\lambda} \right). \tag{2.11}$$

We note that despite the assumption on the discount factor, we empirically show in Section 2.5 that our learning algorithm converges to a robust policy even if this assumption is violated. We also note that the upper bound in (2.11) 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 (2.7), we can get a closed form solution for w_{k+1} using **least squares approach** as

$$w_{k+1} = w_k + (\Phi^\top D\Phi)^{-1}\Phi^\top D(\widetilde{T}_\pi^{(\lambda)}\Phi w_k - \Phi w_k). \tag{2.12}$$

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

$$w_{k+1} = w_k + B^{-1}(Aw_k + C(w_k) + b), \text{ where,} \tag{2.13}$$

$$A = \Phi^\top D(\alpha P_\pi^o - I)\sum_{m=0}^{\infty}(\alpha\lambda P_\pi^o)^m\Phi, \tag{2.14}$$

$$B = \Phi^\top D\Phi, \tag{2.15}$$

$$C(w) = \alpha\Phi^\top D\sum_{t=0}^{\infty}(\alpha\lambda P_\pi^o)^t\sigma_{\widehat{\mathcal{U}}_\pi}(\Phi w), \tag{2.16}$$

$$b = \Phi^\top D\sum_{t=0}^{\infty}(\alpha\lambda P_\pi^o)^t r_\pi. \tag{2.17}$$

Iterations by evaluating (2.13) exactly is intractable for MDPs with large state space, and infeasible if we do not know the transition probability P^o_π. 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

$$w_{t+1} = w_t + \gamma_t B_t^{-1}(A_t w_t + b_t + C_t(w_t)), \text{ where,} \tag{2.18}$$

$$A_t = \frac{1}{t+1}\sum_{\tau=0}^{t} z_\tau \left(\alpha\varphi^\top(s_{\tau+1}) - \varphi^\top(s_\tau)\right), \tag{2.19}$$

$$B_t = \frac{1}{t+1}\sum_{\tau=0}^{t} \varphi(s_\tau)\varphi^\top(s_\tau), \tag{2.20}$$

$$C_t(w) = \frac{\alpha}{t+1}\sum_{\tau=0}^{t} z_\tau \, \sigma_{\widehat{\mathcal{U}}_{s_\tau,\pi(s_\tau)}}(\Phi w), \tag{2.21}$$

$$b_t = \frac{1}{t+1}\sum_{\tau=0}^{t} z_\tau r(s_\tau, \pi(s_\tau)), \tag{2.22}$$

$$z_\tau = \sum_{m=0}^{\tau} (\alpha\lambda)^{\tau-m}\varphi(s_m), \tag{2.23}$$

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 [65]. 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* (2.18)*. 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 (2.18) approximates the exact update equation (2.12) 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 [66, 61] 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.

2.4 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 [42]. 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

$$\pi_{k+1} = \operatorname*{argmax}_{\pi} \widetilde{T}_\pi^{(\lambda)}(\bar{V}_k), \tag{2.24}$$

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,

$$Q_\pi(s,a) = \inf_{P\in\mathcal{P}} \mathbb{E}_P[\sum_{t=0}^{\infty} \alpha^t r(s_t, a_t) \,|s_0 = s, a_0 = a]. \tag{2.25}$$

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 $\varphi(s,a)$ where $\varphi(s,a) = (\varphi_1(s,a), \ldots, \varphi_L(s,a))^\top$ and the linear approximation of the form $\bar{Q}_\pi(s,a) = w^\top\varphi(s,a)$ 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 $\pi_{k+1}(s) = \text{argmax}_{a\in\mathcal{A}}\, \bar{Q}_{\pi_k}(s,a)$. For empirical evaluation purposes, we terminate the policy iteration for some finite value K. RLSPI algorithm is summarized in Algorithm 1.

Algorithm 1 RLSPI Algorithm

Initialization: Policy evaluation weights error ε_0, initial policy π_0.
for $k = 0 \ldots K$ **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. (2.18)-(2.23))
    $t \leftarrow t + 1$
  until $\|w_t - w_{t-1}\|_2 < \varepsilon_0$
  $w_{\pi_k} \leftarrow w_t$
  Update the policy

$$\pi_{k+1}(s) = \operatorname*{argmax}_{a\in\mathcal{A}} \varphi(s,a)^\top w_{\pi_k}$$

end for

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} 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 [67, 68, 69]. 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 [67], 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 $c(\alpha, \beta, 0, \lambda) < 1$,*

$$\limsup_{k\to\infty} \|V^* - V_{\pi_k}\|_\mu \leq \frac{2\sqrt{C_1 C_2}\, c(\alpha, \beta, 0, \lambda)}{(1 - c(\alpha, \beta, 0, \lambda))^2} \limsup_{k\to\infty} \|V_{\pi_k} - \bar{V}_k\|_{d_{\pi_k}}. \tag{2.26}$$

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

$$\limsup_{k\to\infty} \|V^* - V_{\pi_k}\|_\mu \leq \frac{2\sqrt{C_1 C_2}\; c(\alpha,\beta,0,\lambda)}{(1-c(\alpha,\beta,0,\lambda))^3}\; \delta. \tag{2.27}$$

The above theorem, in particular (2.27), gives a (worst case) guarantee for the performance of the policy learned using the RLSPI algorithm. Note that the upper bound in (2.27) is a constant where δ represents the (unavoidable) error due to the linear function approximation. We also note that using '$\limsup$' is necessary due to the policy chattering phenomenon in approximate policy iteration algorithms which exists even in the non-robust case [62].

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 A.3 in the supplementary material due to page limitation.

2.5 Experiments

We implemented our RLSPI algorithm using the MushroomRL library [70], and evaluated its performance against Q-learning algorithm for an environment with discrete action space, deep deterministic policy gradient (DDPG) [44] algorithm for continuous action space, and LSPI algorithm [42]. For comparing with the performance of our RLSPI algorithm against another robust RL algorithm, we implemented the soft-robust algorithms proposed in [48] 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 A.4 of supplementary.

We used the CartPole, MountainCar, and Acrobot environments from OpenAI Gym [71]. We trained LSPI algorithm and our RLSPI algorithm on these environments with nominal parame-

ters (default parameters in OpenAI Gym [71]). We also trained Q-learning with linear function approximation and soft-robust deep Q-network(DQN) [48] algorithms on CartPole environment, DQN and soft-robust DQN [48] algorithms on Acrobot environment, and DDPG and soft-robust DDPG [48] 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 2.1-2.3, 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 2.1 shows the change in the average episodic reward against the probability of picking a random action for the CartPole environment. Figure 2.2 shows the average number of time steps to reach the goal in the MountainCar environment. Figure 2.3 shows the average episodic reward in the Acrobot environment. In all three cases, RLSPI algorithm shows robust performance against the perturbations.

Figures 2.4-2.6 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 [48], 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 [48] does not provide any such guarantees.

2.6 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.

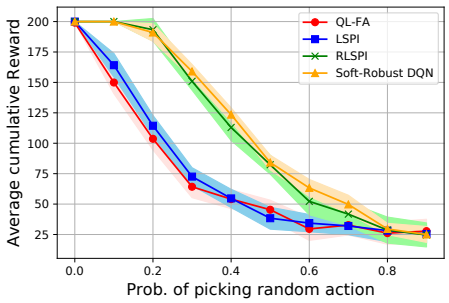

Figure 2.1: CartPole: Performance of RLSPI algorithm with random action sim2real parameter. (Reprinted from [1])

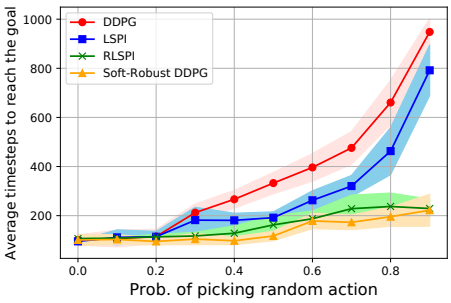

Figure 2.2: MoutainCar: Performance of RLSPI algorithm with random action sim2real parameter. (Reprinted from [1])

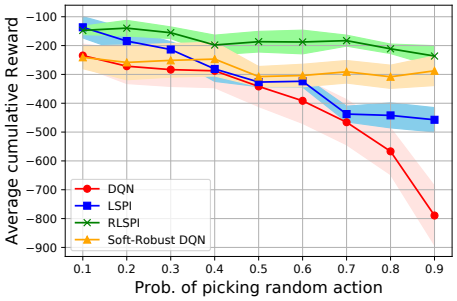

Figure 2.3: Acrobot: Performance of RLSPI algorithm with random action sim2real parameter. (Reprinted from [1])

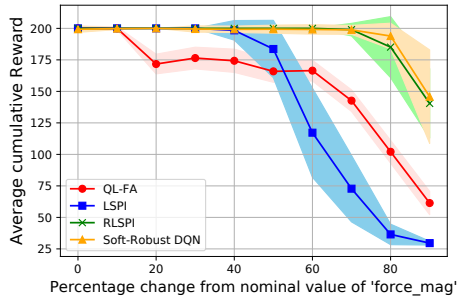

Figure 2.4: CartPole: Performance of RLSPI algorithm with *force-mag* sim2real parameter. (Reprinted from [1])

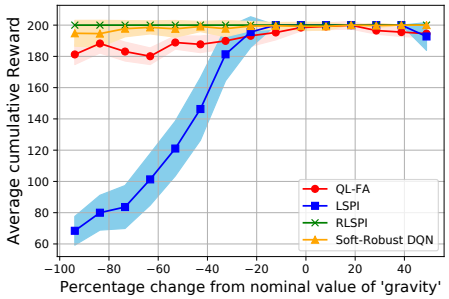

Figure 2.5: CartPole: Performance of RLSPI algorithm with *gravity of Earth* sim2real parameter. (Reprinted from [1])

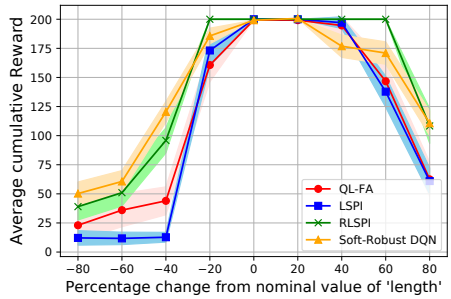

Figure 2.6: CartPole: Performance of RLSPI algorithm with pole length sim2real parameter. (Reprinted from [1])

3. SAMPLE COMPLEXITY OF ROBUST REINFORCEMENT LEARNING WITH A GENERATIVE MODEL*

In this chapter, we focus on providing sample complexity bounds for robust reinforcement learning in finite state and action spaces.

3.1 Introduction

Reinforcement Learning (RL) algorithms typically require a large number of data samples to learn a control policy, which makes the training of RL algorithms directly on the real-world systems expensive and potentially dangerous. This problem is typically avoided by training the RL algorithm on a simulator and transferring the trained policy to the real-world system. However, due to multiple reasons such as the approximation errors incurred while modeling, changes in the real-world parameters over time and possible adversarial disturbances in the real-world, there will be inevitable mismatches between the simulator model and the real-world system. For example, the standard simulator settings of the sensor noise, action delays, friction, and mass of a mobile robot can be different from that of the actual real-world robot. This mismatch between the simulator and real-world model parameters, often called 'simulation-to-reality gap', can significantly degrade the real-world performance of the RL algorithms trained on a simulator model.

Robust Markov Decision Process (RMDP) addresses the *planning* problem of computing the optimal policy that is robust against the parameter mismatch between the simulator and real-world system. The RMDP framework was first introduced in [33, 34]. The RMDP problem has been analyzed extensively in the literature [37, 36, 38, 72, 73], considering different types of uncertainty set and computationally efficient algorithms. However, these works are limited to the planning problem, which assumes the knowledge of the system. Robust RL algorithms for learning the optimal robust policy have also been proposed [40, 1], but they only provide asymptotic convergence guarantees. Robust RL problem has also been addressed using deep RL methods [45, 48, 74, 46, 47].

*Reprinted with permission from Kishan Panaganti, Dileep Kalathil, "Sample Complexity of Robust Reinforcement Learning with a Generative Model." AISTATS 2022.

Table 3.1: Comparison of the sample complexities of different uncertainty sets and the best known result in the non-robust setting [4]. Here $|\mathcal{S}|$ and $|\mathcal{A}|$ are the cardinality of the state and action spaces, c_r is the robust RL problem parameter, and γ is the discount factor. We consider the optimality gap $\varepsilon \in (0, c/(1-\gamma))$, where $c > 0$ is a constant. We refer to Section 3.3.2 for further details. (Reprinted from [2])

UNCERTAINTY SET	TV	CHI-SQUARE	KL	NON-ROBUST
SAMPLE COMPLEXITY	$\mathcal{O}(\frac{\|\mathcal{S}\|^2\|\mathcal{A}\|}{(1-\gamma)^4\varepsilon^2})$	$\mathcal{O}(\frac{c_r\|\mathcal{S}\|^2\|\mathcal{A}\|}{(1-\gamma)^4\varepsilon^2})$	$\mathcal{O}(\frac{\|\mathcal{S}\|^2\|\mathcal{A}\|\exp(1/(1-\gamma))}{c_r^2(1-\gamma)^4\varepsilon^2})$	$\mathcal{O}(\frac{\|\mathcal{S}\|\|\mathcal{A}\|}{(1-\gamma)^3\varepsilon^2})$

However, these works are empirical in nature and do not provide any theoretical guarantees for the learned policies. In particular, there are few works that provide *robust RL algorithms with provable (non-asymptotic) finite-sample performance guarantees.*

In this work, we address the problem of developing a model-based robust RL algorithm with provable finite-sample guarantees on its performance, characterized by the metric of sample complexity in a PAC (probably approximately correct) sense. The RMDP framework assumes that the real-world model lies within some uncertainty set $\mathcal{P}$ around a nominal (simulator) model P^o. The goal is to learn a policy that performs the best under the worst possible model in this uncertainty set. We do not assume that the algorithm knows the exact simulator model (and hence the exact uncertainty set). Instead, similar to the standard (non-robust) RL setting [75, 76, 77, 78, 79, 4, 80], we assume that the algorithm has access to a generative sampling model that can generate next-state samples for all state-action pairs according to the nominal simulator model. In this context, we answer the following important question: *How many samples from the nominal simulator model do we need to learn an ε-optimal robust policy with high probability?*

Our contributions: The main contributions of our work are as follows:

(1) We propose a model-based robust RL algorithm, which we call the robust empirical value iteration algorithm (REVI), for learning an approximately optimal robust policy. We consider three different classes of RMDPs with three different uncertainty sets: (i) Total Variation (TV) uncertainty set, (ii) Chi-square uncertainty set, and (iii) Kullback-Leibler (KL) uncertainty set, each

characterized by its namesake distance measure. Robust RL problem is much more challenging than the standard (non-robust) RL problems due to the inherent nonlinearity associated with the robust dynamic programming and the resulting unbiasedness of the empirical estimates. We overcome this challenge analytically by developing a series of upperbounds that are amenable to using concentration inequality results (which are typically useful only in the unbiased setting), where we exploit a uniform concentration bound with a covering number argument. We rigorously characterize the sample complexity of the proposed algorithm for each of these uncertainty sets. We also make a precise comparison with the sample complexity of non-robust RL.

(2) We give a formal argument for the need for using a robust policy when the simulator model is different from the real-world model. More precisely, we analytically address the question *'why do we need robust policies?'*, by showing that the worst case performance of a non-robust policy can be arbitrarily bad (as bad as a random policy) when compared to that of a robust policy. While the need for robust policies have been discussed in the literature qualitatively, to the best of our knowledge, this is the first work that gives an analytical answer to the above question.

(3) Finally, we demonstrate the performance of our REVI algorithm in two experiment settings and for two different uncertainty sets. In each setting, we show that the policy learned by our proposed REVI algorithm is indeed robust against the changes in the model parameters. We also illustrate the convergence of our algorithm with respect to the number of samples and the number of iterations.

3.1.1 Related Work

Robust RL: An RMDP setting where some state-action pairs are adversarial and the others are stationary was considered by [41], who proposed an online algorithm to address this problem. An approximate robust dynamic programming approach with linear function approximation was proposed in [39]. State aggregation and kernel-based function approximation for robust RL were studied in [81, 59]. [40] proposed a robust version of the Q-learning algorithm. [1] developed a least squares policy iteration approach to learn the optimal robust policy using linear function approximation with provable guarantees. A soft robust RL algorithm was proposed in [48] and

a maximum a posteriori policy optimization approach was used in [46]. While the above mentioned works make interesting contributions to the area of robust RL, they focus either on giving asymptotic performance guarantees or on the empirical performance without giving provable guarantees. In particular, they do not provide provable guarantees on the finite-sample performance of the robust RL algorithms.

The closest to our work is [82], which analyzed the sample complexity with a KL uncertainty set. Our work is different in two significant aspects: Firstly, we consider the total variation uncertainty set and chi-square uncertainty set, in addition to the KL uncertainty set. The analysis for these uncertainty sets are very challenging and significantly different from that of the KL uncertainty set. Secondly, we give a more precise characterization of the sample complexity bound for the KL uncertainty set by clearly specifying the exponential dependence on $(1-\gamma)^{-1}$, where γ is the discount factor, which was left unspecified in [82].

While this paper was under review, we were notified of a concurrent work [83], which also provides similar sample complexity bounds for robust RL. Our proof technique is significantly different from their work. Moreover, we also provide open-source experimental results that illustrate the performance of our robust RL algorithm.

Other related works: Robust control is a well-studied area [84, 85] in the classical control theory. Recently, there are some interesting works that address the robust RL problem using this framework, especially focusing on the linear quadratic regulator setting [86]. Our framework of robust MDP is significantly different from this line of work. Risk sensitive RL algorithms [87] and adversarial RL algorithms [45] also address the robustness problem implicitly. Our approach is different from these works also.

Notations: For any set $\mathcal{X}$, $|\mathcal{X}|$ denotes its cardinality. For any vector x, $\|x\|$ denotes its infinity norm $\|x\|_\infty$.

3.2 Preliminaries: Robust Markov Decision Process

A Markov Decision Process (MDP) is a tuple $(\mathcal{S}, \mathcal{A}, r, P, \gamma)$, 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 $\gamma \in (0,1)$ is the discount factor. The

transition probability function $P_{s,a}(s')$ represents the probability of transitioning to state s' when action a is taken at state s. P is also called the the model of the system. We consider a finite MDP setting where $|\mathcal{S}|$ and $|\mathcal{A}|$ are finite. We will also assume that $r(s,a) \in [0,1]$, for all $(s,a) \in \mathcal{S} \times \mathcal{A}$, without loss of generality.

A (deterministic) policy π maps each state to an action. The value of a policy π for an MDP with model P, evaluated at state s is given by

$$V_{\pi,P}(s) = \mathbb{E}_{\pi,P}[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s], \tag{3.1}$$

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

$$V_P^* = \max_{\pi} V_{\pi,P}, \quad \pi_P^* = \operatorname*{argmax}_{\pi} V_{\pi,P}. \tag{3.2}$$

Uncertainty set: Unlike the standard MDP which considers a single model (transition probability function), the RMDP formulation considers a set of models. We call this set as the *uncertainty set* and denote it as $\mathcal{P}$. We assume that the set $\mathcal{P}$ satisfies the standard *rectangularity condition* [33]. We note that a similar uncertainty set can be considered for the reward function at the expense of additional notations. However, since the analysis will be similar and the samples complexity guarantee will be identical upto a constant, without loss of generality, we assume that the reward function is known and deterministic. We specify a robust MDP as a tuple $M = (\mathcal{S}, \mathcal{A}, r, \mathcal{P}, \gamma)$.

The uncertainty set $\mathcal{P}$ is typically defined as

$$\begin{aligned} \mathcal{P} = \otimes \mathcal{P}_{s,a}, \text{ where, } \mathcal{P}_{s,a} = \{P_{s,a} \in [0,1]^{|\mathcal{S}|} \ : \\ D(P_{s,a}, P^o_{s,a}) \leq c_r, \sum_{s' \in \mathcal{S}} P_{s,a}(s') = 1\}, \end{aligned} \tag{3.3}$$

where $P^o = (P^o_{s,a}, (s,a) \in \mathcal{S} \times \mathcal{A})$ is the nominal transition probability function, $c_r > 0$ indicates

the level of robustness, and $D(\cdot, \cdot)$ is a distance metric between two probability distributions. In the following, we call P^o as the nominal model. In other words, $\mathcal{P}$ is the set of all valid transition probability functions in the neighborhood of the nominal model P^o, where the neighborhood is defined using the distance metric $D(\cdot, \cdot)$. We note that the radius c_r can depend on the state-action pair (s, a). We omit this to reduce the notation complexity. We also note that for $c_r \downarrow 0$, we recover the non-robust regime.

We consider three different uncertainty sets corresponding to three different distance metrics $D(\cdot, \cdot)$.

1. *Total Variation (TV) uncertainty set ($\mathcal{P}^{\mathrm{tv}}$)*: We define $\mathcal{P}^{\mathrm{tv}} = \otimes \mathcal{P}^{\mathrm{tv}}_{s,a}$, where $\mathcal{P}^{\mathrm{tv}}_{s,a}$ is defined as in (3.3) using the total variation distance

$$D_{\mathrm{tv}}(P_{s,a}, P^o_{s,a}) = (1/2)\|P_{s,a} - P^o_{s,a}\|_1. \tag{3.4}$$

2. *Chi-square uncertainty set ($\mathcal{P}^{\mathrm{c}}$)*: We define $\mathcal{P}^{\mathrm{c}} = \otimes \mathcal{P}^{\mathrm{c}}_{s,a}$, where $\mathcal{P}^{\mathrm{c}}_{s,a}$ is defined as in (3.3) using the Chi-square distance

$$D_{\mathrm{c}}(P_{s,a}, P^o_{s,a}) = \sum_{s' \in \mathcal{S}} \frac{(P_{s,a}(s') - P^o_{s,a}(s'))^2}{P^o_{s,a}(s')}. \tag{3.5}$$

3. *Kullback-Leibler (KL) uncertainty set ($\mathcal{P}^{\mathrm{kl}}$)*: We define $\mathcal{P}^{\mathrm{kl}} = \otimes \mathcal{P}^{\mathrm{kl}}_{s,a}$, where $\mathcal{P}^{\mathrm{kl}}_{s,a}$ is defined as in (3.3) using the Kullback-Leibler (KL) distance

$$D_{\mathrm{kl}}(P_{s,a}, P^o_{s,a}) = \sum_{s'} P_{s,a}(s') \log \frac{P_{s,a}(s')}{P^o_{s,a}(s')}. \tag{3.6}$$

We note that the sample complexity and its analysis will depend on the specific form of the uncertainty set.

Robust value iteration: The goal of the RMDP problem is to compute the optimal robust policy which maximizes the value even under the worst model in the uncertainty set. Formally, the *robust value function V_π* corresponding to a policy π and the *optimal robust value function V^** are

defined as [33, 34]

$$V^{\pi} = \inf_{P \in \mathcal{P}} V_{\pi,P}, \quad V^{*} = \sup_{\pi} \inf_{P \in \mathcal{P}} V_{\pi,P}. \tag{3.7}$$

The *optimal robust policy* π^* is such that the robust value function corresponding to it matches the optimal robust value function, i.e., $V^{\pi^*} = V^*$. It is known that there exists a deterministic optimal policy [33] for the RMDP problem. So, we will restrict our attention to the class of deterministic policies.

For any set $\mathcal{B}$ and a vector v, let

$$\sigma_{\mathcal{B}}(v) = \inf\{u^\top v : u \in \mathcal{B}\}.$$

Using this notation, we can define the *robust Bellman operator* [33] as $T(V)(s) = \max_a (r(s,a) + \gamma\sigma_{\mathcal{P}_{s,a}}(V))$. It is known that T is a contraction mapping in infinity norm and the V^* is the unique fixed point of T [33]. Since T is a contraction, *robust value iteration* can be used to compute V^*, similar to the non-robust MDP setting [33]. More precisely, the robust value iteration, defined as $V_{k+1} = TV_k$, converges to V^*, i.e., $V_k \to V^*$. Similar to the optimal robust value function, we can also define the optimal robust action-value function as $Q^*(s,a) = r(s,a) + \gamma\sigma_{\mathcal{P}_{s,a}}(V^*)$. Similar to the non-robust setting, it is straight forward to show that $\pi^*(s) = \operatorname{argmax}_a Q^*(s,a)$ and $V^*(s) = \max_a Q^*(s,a)$.

3.3 Algorithm and Sample Complexity

The robust value iteration requires the knowledge of the nominal model P^o and the radius of the uncertainty set c_r to compute V^* and π^*. While c_r may be available as design parameter, the form of the nominal model may not be available in most practical problems. So, we do not assume the knowledge of the nominal model P^o. Instead, similar to the non-robust RL setting, we assume only to have access to the samples from a generative model, which can generate samples of the next state s' according to $P^o_{s,a}(\cdot)$, given the state-action pair (s,a) as the input. We propose a model-

Algorithm 2 Robust Empirical Value Iteration (REVI) Algorithm

Input: Loop termination number K
Initialize: $Q_0 = 0$
Compute the empirical uncertainty set $\widehat{\mathcal{P}}$ according to (3.8)
for $k = 0, \cdots, K-1$ **do**
    $V_k(s) = \max_a Q_k(s,a),\ \forall s$
    $Q_{k+1}(s,a) = r(s,a) + \gamma \sigma_{\widehat{\mathcal{P}}_{s,a}}(V_k),\ \forall (s,a)$
end for
Output: $\pi_K(s) = \operatorname{argmax}_a Q_K(s,a), \forall s \in \mathcal{S}$

based robust RL algorithm that uses these samples to estimate the nominal model and uncertainty set.

3.3.1 Robust Empirical Value Iteration (REVI) Algorithm

We first get a maximum likelihood estimate $\widehat{P}^o$ of the nominal model P^o by following the standard approach [76, Algorithm 3]. More precisely, we generate N next-state samples corresponding to each state-action pairs. Then, the maximum likelihood estimate $\widehat{P}^o$ is given by $\widehat{P}^o_{s,a}(s') = N(s,a,s')/N$, where $N(s,a,s')$ is the number of times the state s' is realized out of the total N transitions from the state-action pair (s,a). Given $\widehat{P}^o$, we can get an empirical estimate $\widehat{\mathcal{P}}$ of the uncertainty set $\mathcal{P}$ as,

$$\widehat{\mathcal{P}} = \otimes\ \widehat{\mathcal{P}}_{s,a},\ \text{where},\ \ \widehat{\mathcal{P}}_{s,a} = \{P \in [0,1]^{\mathcal{S}}\ : \\ D(P_{s,a}, \widehat{P}_{s,a}) \leq c_r, \sum_{s' \in \mathcal{S}} P_{s,a}(s') = 1\}, \tag{3.8}$$

where D is one of the metrics specified in (3.4) - (3.6).

For finding an approximately optimal robust policy, we now consider the empirical RMDP $\widehat{M} = (\mathcal{S}, \mathcal{A}, r, \widehat{\mathcal{P}}, \gamma)$ and perform robust value iteration using $\widehat{\mathcal{P}}$. This is indeed our approach, which we call the Robust Empirical Value Iteration (REVI) Algorithm. The optimal robust policy and value function of $\widehat{M}$ are denoted as $\widehat{\pi}^\star, \widehat{V}^\star$, respectively.

3.3.2 Sample Complexity

In this section we give the sample complexity guarantee of the REVI algorithm for the three uncertainty sets. We first consider the TV uncertainty set.

Theorem 4 (TV Uncertainty Set)**.** *Consider an RMDP with a total variation uncertainty set $\mathcal{P}^{\mathrm{tv}}$. Fix $\delta \in (0,1)$ and $\varepsilon \in (0, 24\gamma/(1-\gamma))$. Consider the REVI algorithm with $K \geq K_0$ and $N \geq N^{\mathrm{tv}}$, where*

$$K_0 = \frac{1}{\log(1/\gamma)} \log(\frac{8\gamma}{\varepsilon(1-\gamma)^2}) \text{ and} \tag{3.9}$$

$$N^{\mathrm{tv}} = \frac{72\gamma^2|\mathcal{S}|}{(1-\gamma)^4\varepsilon^2} \log(\frac{144\gamma|\mathcal{S}||\mathcal{A}|}{(\delta\varepsilon(1-\gamma)^2)}). \tag{3.10}$$

Then, $\|V^* - V^{\pi_K}\| \leq \varepsilon$ with probability at least $1-2\delta$.

Remark 1. *The total number of samples needed in the REVI algorithm is $N_{\mathrm{total}} = N|\mathcal{S}||\mathcal{A}|$. So the sample complexity of the REVI algorithm with the TV uncertainty set is $\mathcal{O}(\frac{|\mathcal{S}|^2|\mathcal{A}|}{(1-\gamma)^4\varepsilon^2})$.*

Remark 2 (Comparison with the sample complexity of the non-robust RL)**.** *For the non-robust setting, the lowerbound for the total number of samples from the generative sampling device is $\Omega(\frac{|\mathcal{S}||\mathcal{A}|}{\varepsilon^2(1-\gamma)^3} \log \frac{|\mathcal{S}||\mathcal{A}|}{\delta})$ [76, Theorem 3]. The variance reduced value iteration algorithm proposed in [78] achieves a sample complexity of $\mathcal{O}(\frac{|\mathcal{S}||\mathcal{A}|}{\varepsilon^2(1-\gamma)^3} \log \frac{|\mathcal{S}||\mathcal{A}|}{\delta\varepsilon})$, matching the lower bound. However, this work is restricted to $\varepsilon \in (0,1)$, whereas ε can be considered upto the value $1/(1-\gamma)$ for the MDP problems. Recently, this result has been further improved recently by [79] and [4], which considered $\varepsilon \in (0, 1/\sqrt{(1-\gamma)})$ and $\varepsilon \in (0, 1/(1-\gamma))$, respectively.*

Theorem 4 for the robust RL setting also considers ε upto $\mathcal{O}(1/(1-\gamma))$. However, the sample complexity obtained is worse by a factor of $|\mathcal{S}|$ and $1/(1-\gamma)$ when compared to the non-robust setting. These additional terms are appearing in our result due to a covering number argument we used in the proof, which seems necessary for getting a tractable bound. However, it is not clear if this is fundamental to the robust RL problem with TV uncertainty set. We leave this investigation for our future work.

We next consider the chi-square uncertainty set.

Theorem 5 (Chi-square Uncertainty Set)**.** *Consider an RMDP with a Chi-square uncertainty set $\mathcal{P}^{\mathrm{c}}$. Fix $\delta \in (0,1)$ and $\varepsilon \in (0, 16\gamma/(1-\gamma))$, for an absolute constant $c_1 > 1$. Consider the REVI algorithm with $K \geq K_0$ and $N \geq N^{\mathrm{c}}$, where K_0 is as given in* (3.9) *and*

$$N^{\mathrm{c}} = \frac{64\gamma^2(2c_r+1)|\mathcal{S}|}{(1-\gamma)^4\varepsilon^2}\log(\frac{192|\mathcal{S}||\mathcal{A}|\gamma}{(\delta\varepsilon(1-\gamma)^2)}). \tag{3.11}$$

Then, $\|V^* - V^{\pi_K}\| \leq \varepsilon$ with probability at least $1-2\delta$.

Remark 3. *The sample complexity of the algorithm with the chi-square uncertainty set is $\mathcal{O}(\frac{|\mathcal{S}|^2|\mathcal{A}|c_r}{(1-\gamma)^4\varepsilon^2})$. The order of sample complexity remains the same compared to that of the TV uncertainty set given in Theorem 4.*

Finally, we consider the KL uncertainty set.

Theorem 6 (KL Uncertainty Set)**.** *Consider an RMDP with a KL uncertainty set $\mathcal{P}^{\mathrm{kl}}$. Fix $\delta \in (0,1)$ and $\varepsilon \in (0, 1/(1-\gamma))$. Consider the REVI algorithm with $K \geq K_0$ and $N \geq N^{\mathrm{kl}}$, where K_0 is as in* (3.9) *and*

$$N^{\mathrm{kl}} = \frac{8\gamma^2|\mathcal{S}|}{c_r^2(1-\gamma)^4\varepsilon^2}\exp(\frac{2\lambda_{\mathrm{kl}}+4}{\lambda_{\mathrm{kl}}(1-\gamma)})\log(\frac{9|\mathcal{S}||\mathcal{A}|}{\delta\lambda_{\mathrm{kl}}(1-\gamma)}), \tag{3.12}$$

and λ_{kl} is a problem dependent parameter but independent of N^{kl}. Then, $\|V^* - V^{\pi_K}\| \leq \varepsilon$ with probability at least $1-2\delta$.

Remark 4. *The sample complexity with the KL uncertainty set is $\mathcal{O}(\frac{|\mathcal{S}|^2|\mathcal{A}|}{(1-\gamma)^4\varepsilon^2c_r^2}\exp(\frac{1}{(1-\gamma)}))$. We note that [82] also considered the robust RL problem with KL uncertainty set. They provided a sample complexity bound of the form $\mathcal{O}(\frac{C|\mathcal{S}|^2|\mathcal{A}|}{(1-\gamma)^4\varepsilon^2c_r^2})$, However the exponential dependence on $1/(1-\gamma)$ was hidden inside the constant C. In this work, we clearly specify the depends on the factor $1/(1-\gamma)$.*

3.4 Why Do We Need Robust Policies?

In the introduction, we have given a qualitative description about the need for finding a robust policy. In this section, we give a formal argument to show that the worst case performance of a non-robust policy can be arbitrarily bad (as bad as a random policy) when compared to that of a robust policy.

We consider a simple setting with an uncertainty set that contains only two models, i.e., $\mathcal{P} = \{P^o, P'\}$. Let π^* be the optimal robust policy. Following the notation in (3.2), let $\pi^o = \pi_{P^o}$ and $\pi' = \pi_{P'}$ be the non-robust optimal policies when the model is P^o and P', respectively. Assume that nominal model is P^o and we decide to employ the non-robust policy π^o. The worst case performance of π^o is characterized by its robust value function V^{π^o} which is $\min\{V_{\pi^o,P^o}, V_{\pi^o,P'}\}$.

We now state the following result.

Theorem 7 (Robustness Gap)**.** *There exists a robust MDP M with uncertainty set $\mathcal{P} = \{P^o, P'\}$, discount factor $\gamma \in (\gamma_o, 1]$, and state $s_1 \in \mathcal{S}$ such that*

$$V^{\pi^o}(s_1) \leq V^{\pi^*}(s_1) - c/(1-\gamma),$$

where c is a positive constant, π^* is the optimal robust policy, and $\pi^o = \pi_{P^o}$ is the non-robust optimal policy when the model is P^o.

Theorem 7 states that the worst case performance of the non-robust policy π^o is lower than that of the optimal robust policy π^*, and this performance gap is $\Omega(1/(1-\gamma))$. Since $|r(s,a)| \leq 1, \forall (s,a) \in \mathcal{S} \times \mathcal{A}$ by assumption, $\|V_{\pi,P}\| \leq 1/(1-\gamma)$ for any policy π and any model P. Therefore, the difference between the optimal (robust) value function and the (robust) value function of an arbitrary policy cannot be greater than $\mathcal{O}(1/(1-\gamma))$. Thus the worst-case performance of the non-robust policy π^o can be as bad as an arbitrary policy in an order sense.

3.5 Sample Complexity Analysis

In this section we explain the key ideas used in the analysis of the REVI algorithm for obtaining the sample complexity bound for each of the uncertainty sets. Recall that we consider an RMDP M and its empirical estimate version as $\widehat{M}$.

To bound $\|V^* - V^{\pi_K}\|$, we split it into three terms as $\|V^* - V^{\pi_K}\| \leq \|V^* - \widehat{V}^*\| + \|\widehat{V}^* - \widehat{V}^{\pi_K}\| + \|\widehat{V}^{\pi_K} - V^{\pi_K}\|$, and analyze each term separately.

Analyzing the second term, $\|\widehat{V}^* - \widehat{V}^{\pi_K}\|$, is similar to that of non-robust algorithms. Due to the contraction property of the robust Bellman operator, it is straight forward to show that $\|\widehat{V}^* - \widehat{V}^{\pi_{k+1}}\| \leq \gamma\|\widehat{V}^* - \widehat{V}^{\pi_k}\|$ for any k. This exponential convergence, with some additional results from the MDP theory, enables us to get a bound $\|\widehat{V}^* - \widehat{V}^{\pi_K}\| \leq 2\gamma^{K+1}/(1-\gamma)^2$.

The analysis of terms $\|V^* - \widehat{V}^*\|$ and $\|\widehat{V}^{\pi_K} - V^{\pi_K}\|$ are however non-trivial and significantly more challenging compared to the non-robust setting. We will focus on the latter, and the analysis of the former is similar.

For any policy π and for any state s, and denoting $a = \pi(s)$, we have

$$\begin{aligned} V^\pi(s) - \widehat{V}^\pi(s) &= \gamma\sigma_{\mathcal{P}_{s,a}}(V^\pi) - \gamma\sigma_{\widehat{\mathcal{P}}_{s,a}}(\widehat{V}^\pi) \\ &= \gamma(\sigma_{\mathcal{P}_{s,a}}(V^\pi) - \sigma_{\mathcal{P}_{s,a}}(\widehat{V}^\pi)) + \gamma(\sigma_{\mathcal{P}_{s,a}}(\widehat{V}^\pi) - \sigma_{\widehat{\mathcal{P}}_{s,a}}(\widehat{V}^\pi)) \end{aligned} \tag{3.13}$$

To bound the first term in (3.13), we present a result that shows that $\sigma_{\mathcal{P}_{s,a}}$ is 1-Lipschitz in the sup-norm.

Lemma 1. *For any $(s,a) \in \mathcal{S} \times \mathcal{A}$ and for any $V_1, V_2 \in \mathbb{R}^{|\mathcal{S}|}$, we have $|\sigma_{\mathcal{P}_{s,a}}(V_1) - \sigma_{\mathcal{P}_{s,a}}(V_2)| \leq \|V_1 - V_2\|$ and $|\sigma_{\widehat{\mathcal{P}}_{s,a}}(V_1) - \sigma_{\widehat{\mathcal{P}}_{s,a}}(V_2)| \leq \|V_1 - V_2\|$.*

Using the above lemma, the first term in (3.13) will be bounded by $\gamma\|V^\pi - \widehat{V}^\pi\|$ and the discount factor makes this term amenable to getting a closed form bound.

Obtaining a bound for $\sigma_{\mathcal{P}_{s,a}}(\widehat{V}^\pi) - \sigma_{\widehat{\mathcal{P}}_{s,a}}(\widehat{V}^\pi)$ is the most challenging part of our analysis. In the non-robust setting, this will be equivalent to the error term $P^o_{s,a}V - \widehat{P}_{s,a}V$, which is unbiased and can be easily bounded using concentration inequalities. In the robust setting, however, because

of the nonlinear nature of the function $\sigma(\cdot)$, $\mathbb{E}[\sigma_{\widehat{\mathcal{P}}_{s,a}}(\widehat{V}^\pi)] \neq \sigma_{\mathcal{P}_{s,a}}(\widehat{V}^\pi)$. So, using concentration inequalities to get a bound is not immediate. Our strategy is to find appropriate upperbound for this term that is amenable to using concentration inequalities. To that end, we will analyze this term separately for each of the three uncertainty set.

3.5.1 Total variation uncertainty set

We will first get following upperbound:

Lemma 2 (TV uncertainty set)**.** *Let* $\mathcal{V} = \{V \in \mathbb{R}^{|\mathcal{S}|} : \|V\| \leq 1/(1-\gamma)\}$. *For any* $(s,a) \in \mathcal{S} \times \mathcal{A}$ *and for any* $V \in \mathcal{V}$,

$$|\sigma_{\widehat{\mathcal{P}}^{\text{tv}}_{s,a}}(V) - \sigma_{\mathcal{P}^{\text{tv}}_{s,a}}(V)| \leq 2 \max_{\mu \in \mathcal{V}} |\widehat{P}_{s,a}\mu - P^o_{s,a}\mu|. \tag{3.14}$$

While the term $|\widehat{P}_{s,a}\mu - P^o_{s,a}\mu|$ in (3.14) can be upperbounded using the standard Hoeffding's inequality, bounding $\max_{\mu\in\mathcal{V}} |\widehat{P}_{s,a}\mu - P^o_{s,a}\mu|$ is more challenging as it requires a uniform bound. Since μ can take a continuum of values, a simple union bound argument will also not work. We overcome this issue by using a covering number argument and obtain the following bound.

Lemma 3. *Let* $V \in \mathbb{R}^{|\mathcal{S}|}$ *with* $\|V\| \leq 1/(1-\gamma)$. *For any* $\eta, \delta \in (0,1)$,

$$\max_{\mu: 0 \leq \mu \leq V} \max_{s,a} |\widehat{P}_{s,a}\mu - P^o_{s,a}\mu| \leq \frac{1}{1-\gamma}\sqrt{\frac{|\mathcal{S}|}{2N}\log(\frac{12|\mathcal{S}||\mathcal{A}|}{(\delta\eta(1-\gamma))}} + 2\eta,$$

with probability at least $1 - \delta/2$.

We note that this uniform bound adds an additional $\sqrt{|\mathcal{S}|}$ factor compared to the non-robust setting, which results in an additional $|\mathcal{S}|$ in the sample complexity. Combining these, we finally get the following result.

Proposition 4. *Let* $\mathcal{V} = \{V \in \mathbb{R}^{|\mathcal{S}|} : \|V\| \leq 1/(1-\gamma)\}$. *For any* $\eta, \delta \in (0,1)$, *with probability at*

least $1-\delta$*,*

$$\max_{V\in\mathcal{V}}\max_{s,a}|\sigma_{\widehat{\mathcal{P}}^{\mathrm{tv}}_{s,a}}(V)-\sigma_{\mathcal{P}^{\mathrm{tv}}_{s,a}}(V)|\leq C^{\mathrm{tv}}_u(N,\eta,\delta),\ \textit{where,}$$

$$C^{\mathrm{tv}}_u(N,\eta,\delta)=4\eta+\frac{2}{1-\gamma}\sqrt{\frac{|\mathcal{S}|\log(6|\mathcal{S}||\mathcal{A}|/(\delta\eta(1-\gamma)))}{2N}}. \tag{3.15}$$

Tracing back the steps to (3.13), we can get an arbitrary small bound for $\|V^\pi-\widehat{V}^\pi\|$ by selecting N appropriately, as specified in Theorem 4.

3.5.2 Chi-square uncertainty set

We will first get the following upperbound:

Lemma 4 (Chi-square uncertainty set)**.** *For any* $(s,a)\in\mathcal{S}\times\mathcal{A}$ *and for any* $V\in\mathbb{R}^{|\mathcal{S}|}, \|V\|\leq 1/(1-\gamma)$*,*

$$\begin{aligned}&|\sigma_{\widehat{\mathcal{P}}^{\mathrm{c}}_{s,a}}(V)-\sigma_{\mathcal{P}^{\mathrm{c}}_{s,a}}(V)|\leq\\&\max_{\mu:0\leq\mu\leq V}|\sqrt{c_r\mathrm{Var}_{\widehat{P}_{s,a}}(V-\mu)}-\sqrt{c_r\mathrm{Var}_{P^o_{s,a}}(V-\mu)}|\\&+\max_{\mu:0\leq\mu\leq V}|\widehat{P}_{s,a}(V-\mu)-P^o_{s,a}(V-\mu)|.\end{aligned} \tag{3.16}$$

The second term of (3.16) can be bounded using Lemma 3. However, the first term, which involves the square-root of the variance is more challenging. We use a concentration inequality that is applicable for variance to overcome this challenge. Finally, we get the following result.

Proposition 5. *Let* $\mathcal{V}=\{V\in\mathbb{R}^{|\mathcal{S}|}:\|V\|\leq 1/(1-\gamma)\}$*. For any* $\eta,\delta\in(0,1)$*, with probability at least* $(1-\delta)$*,*

$$\max_{V\in\mathcal{V}}\max_{s,a}|\sigma_{\widehat{\mathcal{P}}^{\mathrm{c}}_{s,a}}(V)-\sigma_{\mathcal{P}^{\mathrm{c}}_{s,a}}(V)|\leq C^{\mathrm{c}}_u(N,\eta,\delta),\ \textit{where,}$$

$$C_u^{\mathrm{c}}(N,\eta,\delta) \le \sqrt{\frac{32\eta c_r}{1-\gamma}} + 2\eta + \frac{1}{1-\gamma}\sqrt{\frac{(2c_r+1)|\mathcal{S}|\log(12|\mathcal{S}||\mathcal{A}|/(\delta\eta(1-\gamma)))}{N}}, \tag{3.17}$$

Now, by selecting appropriate N as specified in Theorem 5, we can show the ε-optimality of π_K.

The details on the KL uncertainty set analysis is included in the appendix.

3.6 Experiments

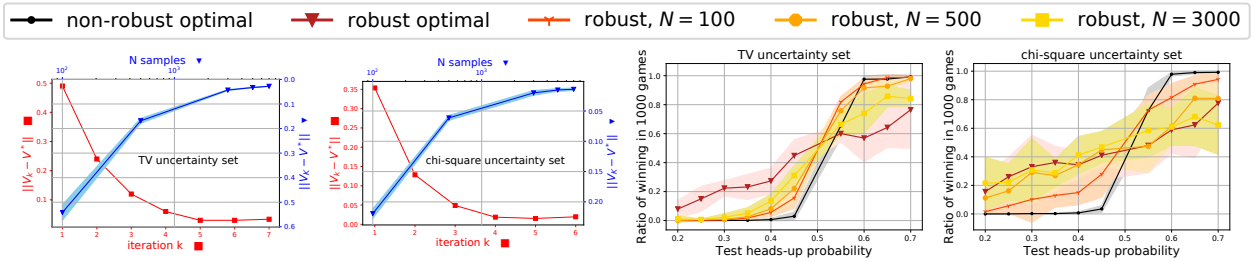

Figure 3.1: *Experiment results for the Gambler's problem.* The first two plots shows the rate of convergence with respect to the number of iterations (k) and the rate of convergence with respect to the number of samples (N) for the TV and chi-square uncertainty set, respectively. The third and fourth plots shows the robustness of the learned policy against changes in the model parameter (heads-up probability). (Reprinted from [2])

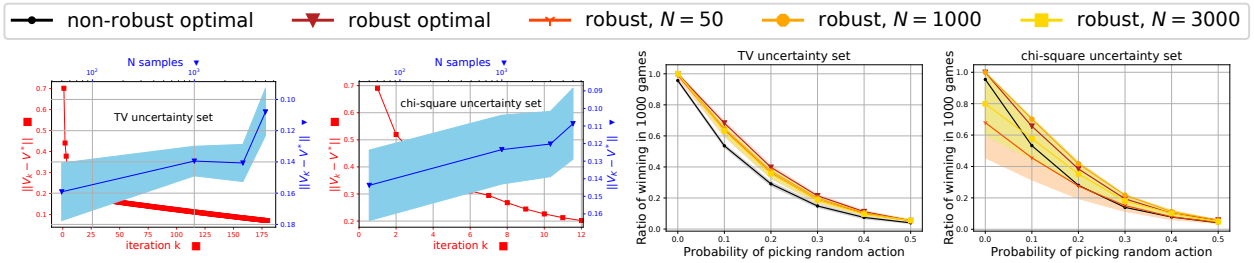

Figure 3.2: *Experiment results for the FrozenLake8x8 environment.* The first two plots shows the rate of convergence with respect to the number of iterations (k) and the rate of convergence with respect to the number of samples (N) for the TV and chi-square uncertainty set, respectively. The third and fourth plots shows the robustness of the learned policy against changes in the model parameter (probability of picking a random action). (Reprinted from [2])

In this section we demonstrate the convergence behavior and robust performance of our REVI

algorithm using numerical experiments. We consider two different settings, namely, the *Gambler's Problem* environment [6, Example 4.3] and *FrozenLake8x8* environment in OpenAI Gym [71]. We also consider the TV uncertainty set and chi-square uncertainty set. We solve the optimization problem $\sigma_{\widehat{\mathcal{P}}}$ and $\sigma_{\mathcal{P}}$ using the Scipy [88] optimization library.

We illustrate the following important characteristics of the REVI algorithm:

(1) Rate of convergence with respect to the number of iterations: To demonstrate this, we plot $\|V_k - V^*\|$ against the iteration number k, where V_k is the value at the kth step of the REVI algorithm with $N = 5000$. We compute V^* using the full knowledge of the uncertainty set for benchmarking the performance of the REVI algorithm.

(2) Rate of convergence with respect to the number of samples: To show this, we plot $\|V_K(N) - V^*\|$ against the number of samples N, where $V_K(N)$ is final value obtained from the REVI algorithm using N samples.

(3) Robustness of the learned policy: To demonstrate this, we plot the number of times the robust policy π_K (obtained from the REVI algorithm) successfully completed the task as a function of the change in an environment parameter. We perform 1000 trials for each environment and each uncertainty set, and plot the fraction of the success.

Gambler's Problem: In gambler's problem, a gambler starts with a random balance in her account and makes bets on a sequence of coin flips, winning her stake with heads and losing with tails, until she wins $\$100$ or loses all money. This problem can be formulated as a chain MDP with states in $\{1, \cdots, 99\}$ and when in state s the available actions are in $\{0, 1, \cdots, \min(s, 100 - s)\}$. The agent is rewarded 1 after reaching a goal and rewarded 0 in every other timestep. The biased coin probability is fixed throughout the game. We denote its heads-up probability as p_h and use 0.6 as a nominal model for training our algorithm. We also fix $c_r = 0.2$ for the chi-square uncertainty set experiments and $c_r = 0.4$ for the TV uncertainty set experiments.

The red curves with square markers in the first two plots in Fig. 3.1 show the rate of convergence with respect to the number of iterations for TV and chi-square uncertainty sets respectively. As expected, convergence is fast due to the contraction property of the robust Bellman operator.

The blue curves with triangle markers in the first two plots in Fig. 3.1 show the rate of convergence with respect to the number of samples for TV and chi-square uncertainty sets. We generated these curves for 10 different seed runs. The bold line depicts the mean of these runs and the error bar is the standard deviation. As expected, the plots show that $V_K(N)$ converges to V^* as N increases.

We then demonstrate the robustness of the approximate robust policy π_K (obtained with $N = 100, 500, 3000$) by evaluating its performance on environments with different values of p_h. We plot the fraction of the wins out of 1000 trails. We also plot the performance the optimal robust policy π^* as a benchmark. The third and fourth plot in Fig. 3.1 show the results with TV and chi-square uncertainty sets respectively. We note that the performance of the non-robust policy decays drastically as we decrease the parameter p_h from its nominal value 0.6. On the other hand, the optimal robust policy performs consistently better under this change in the environment. We also note that $\pi_K(N)$ closely follows the performance of π^* for large enough N.

Frozen Lake environment: *FrozenLake8x8* is a gridworld environment of size 8×8. It consists of some flimsy tiles which makes the agent fall into the water. The agent is rewarded 1 after reaching a goal tile without falling and rewarded 0 in every other timestep. We use the *FrozenLake8x8* environment with default design as our nominal model except that we make the probability of transitioning to a state in the intended direction to be 0.4 (the default value is $1/3$). We also set $c_r = 0.35$ for the chi-square uncertainty set experiments and $c_r = 0.7$ for the TV uncertainty set experiments.

The red curves in the first two plots in Fig. 3.2 show the rate of convergence with respect to the number of iterations for TV and chi-square uncertainty sets respectively. The blue curves in the first two plots in Fig. 3.2 show the rate of convergence with respect to the number of samples for TV and chi-square uncertainty sets respectively. The behavior is similar to the one observed in the case of gambler's problem.

We demonstrate the robustness of the learned policy by evaluating it on FrozenLake test environments with action perturbations. In the 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. In addition, we change the probability of transitioning to a state in the intended direction to be 0.2 for these test environments. We observe that the performance of the robust RL policy is consistently better than the non-robust policy as we introduce model mismatch in terms of the probability of picking random actions. We also note that $\pi_K(N)$ closely follows the performance of π^* for large enough N.

We note that we have included our code for experiments in this GitHub page. We note that we can employ a hyperparameter learning strategy to find the best value of c_r. We demonstrate this on the FrozenLake environment for the TV uncertainty set. We computed the optimal robust policy for each $c_r \in \{0.1, 0.2, \ldots, 1.6\}$. We tested these policies across 1000 games with random action probabilities $\{0.1, 0.2, \ldots, 0.5\}$ on the test environment. We found that the policy for $c_r = 1.2$ has the best winning ratio across all the random action probabilities. We do not exhibit exhaustive experiments on this hyperparameter learning strategy as it is out of scope of the intent of this manuscript.

3.7 Conclusion and Future Work

We presented a model-based robust reinforcement learning algorithm called Robust Empirical Value Iteration algorithm, where we used an approximate robust Bellman updates in the vanilla robust value iteration algorithm. We provided a finite sample performance characterization of the learned policy with respect to the optimal robust policy for three different uncertainty sets, namely, the total variation uncertainty set, the chi-square uncertainty set, and the Kullback-Leibler uncertainty set. We also demonstrated the performance of REVI algorithm on two different environments showcasing its theoretical properties of convergence. We also showcased the REVI algorithm based policy being robust to the changes in the environment as opposed to the non-robust policies.

The goal of this work was to develop the fundamental theoretical results for the finite state space and action space regime. As mentioned earlier, the sub-optimality of the sample complexity of our REVI algorithm in factors $|\mathcal{S}|$ and $1/(1-\gamma)$ needs more investigation and refinements in

the analyses. In the future, we will extend this idea to robust RL with linear and nonlinear function approximation architectures and for more general models in deep RL.

4. ROBUST REINFORCEMENT LEARNING USING OFFLINE DATA*

In this chapter, we focus on developing robust reinforcement learning algorithm for large state space with access to offline historical data.

4.1 Introduction

Reinforcement learning (RL) algorithms often require a large number of data samples to learn a control policy. As a result, training them directly on the real-world systems is expensive and potentially dangerous. This problem is typically overcome by training them on a simulator (online RL) or using a pre-collected offline dataset (offline RL). The offline dataset is usually collected either from a sophisticated simulator of the real-world system or from the historical measurements. The trained RL policy is then deployed assuming that the training environment, the simulator or the offline data, faithfully represents the model of the real-world system. This assumption is often incorrect due to multiple factors such as the approximation errors incurred while modeling, changes in the real-world parameters over time and possible adversarial disturbances in the real-world. For example, the standard simulator settings of the sensor noise, action delay, friction, and mass of a mobile robot can be different from that of the actual real-world robot, in addition to changes in the terrain, weather conditions, lighting, and obstacle densities of the testing environment. Unfortunately, the current RL control policies can fail dramatically when faced with even mild changes in the training and testing environments [30, 31, 32].

The goal in robust RL is to learn a policy that is robust against the model parameter mismatches between the training and testing environments. The robust planning problem is formalized using the framework of Robust Markov Decision Process (RMDP) [33, 34]. Unlike the standard MDP which considers a single model (transition probability function), the RMDP formulation considers a set of models which is called the *uncertainty set*. The goal is to find an optimal robust policy that performs the best under the worst possible model in this uncertainty set. The minimization over

*Reprinted with permission from Kishan Panaganti, Zaiyan Xu, Dileep Kalathil, Mohammad Ghavamzadeh, "Robust Reinforcement Learning using Offline Data." Neural Information Processing Systems. PMLR, 2022.

the uncertainty set makes the robust MDP and robust RL problems significantly more challenging than their non-robust counterparts.

In this work, we study the problem of developing a robust RL algorithm with provably optimal performance for an RMDP with arbitrarily large state spaces, using only offline data with function approximation. Before stating the contributions of our work, we provide a brief overview of the results in offline and robust RL that are directly related to ours. We leave a more thorough discussion on related works to Appendix C.4.

Offline RL: Offline RL considers the problem of learning the optimal policy only using a pre-collected (offline) dataset. Offline RL problem has been addressed extensively in the literature [89, 63, 90, 91, 92, 93, 94]. Many recent works develop deep RL algorithms and heuristics for the offline RL problem, focusing on the algorithmic and empirical aspects [95, 96, 97, 98, 99]. A number of theoretical work focus on analyzing the variations of Fitted Q-Iteration (FQI) algorithm [100, 101], by identifying the necessary and sufficient conditions for the learned policy to be approximately optimal and characterizing the performance in terms of sample complexity [68, 102, 69, 91, 103, 94]. All these works assume that the offline data is generated according to a single model and the goal is to find the optimal policy for the MDP with the same model. In particular, none of these works consider the *offline robust RL problem* where the offline data is generated according to a (training) model which can be different from the one in testing, and the goal is to learn a policy that is robust w.r.t. an uncertainty set.

Robust RL: The RMDP framework was first introduced in [33, 34]. The RMDP problem has been analyzed extensively in the literature [37, 36, 38, 72, 73] providing computationally efficient algorithms, but these works are limited to the planning problem. Robust RL algorithms with provable guarantees have also been proposed [41, 39, 40, 1, 104], but they are limited to tabular or linear function approximation settings and only provide asymptotic convergence guarantees. Robust RL problem has also been addressed using deep RL methods [45, 48, 74, 46, 47]. However, these works do not provide any theoretical guarantees on the performance of the learned policies.

The works that are closest to ours are by [82, 83, 2] that address the robust RL problem in a

tabular setting under the generative model assumption. Due to the generative model assumption, the offline data has the same uniform number of samples corresponding to each and every state-action pair, and tabular setting allows the estimation of the uncertainty set followed by solving the planning problem. Our work is significantly different from these in the following way: (i) we consider a robust RL problem with arbitrary large state space, instead of the small tabular setting, (ii) we consider a true offline RL setting where the state-action pairs are sampled according to an arbitrary distribution, instead of using the generative model assumption, (iii) we focus on a function approximation approach where the goal is to directly learn optimal robust value/policy using function approximation techniques, instead of solving the tabular planning problem with the estimated model. *To the best of our knowledge, this is the first work that addresses the offline robust RL problem with arbitrary large state space using function approximation, with provable guarantees on the performance of the learned policy.*

Offline Robust RL: Challenges and Our Contributions: Offline robust RL is significantly more challenging than its non-robust counterpart mainly because of the following key difficulties. (i) Data generation: The optimal robust policy is computed by taking the infimum over all models in the uncertainty set $\mathcal{P}$. However, generating data according to all models in $\mathcal{P}$ is clearly infeasible. It may only be possible to get the data from a nominal (training) model P^o. *How do we use the data from a nominal model to account for the behavior of all the models in the uncertainty set $\mathcal{P}$?*

(ii) Optimization over the uncertainty set $\mathcal{P}$: The robust Bellman operator (defined in (4.3)) involves a minimization over $\mathcal{P}$, which is a significant computational challenge. Moreover, the uncertainty set $\mathcal{P}$ itself is unknown in the RL setting. *How do we solve the optimization over $\mathcal{P}$?*

(iii) Function approximation: Approximation of the robust Bellman update requires a modified target function which also depends on the approximate solution of the optimization over the uncertainty set. *How do we perform the offline RL update accounting for both approximations?*

As the *key technical contributions* of this work, we first derive a dual reformulation of the robust Bellman operator which replaces the expectation w.r.t. all models in the uncertainty set $\mathcal{P}$

with an expectation only w.r.t. the nominal (training) model P^o. This enables using the offline data generated by P^o for learning, without relying on high variance importance sampling techniques to account for all models in $\mathcal{P}$. Following the same reformulation, we then show that the optimization problem over $\mathcal{P}$ can be further reformulated as functional optimization. We solve this functional optimization problem using empirical risk minimization and obtain performance guarantees using the Rademacher complexity based bounds. We then use the approximate solution obtained from the empirical risk minimization to generate modified target samples that are then used to approximate robust Bellman update through a generalized least squares approach with provably bounded errors. Performing these operations iteratively results in our proposed Robust Fitted Q-Iteration (RFQI) algorithm, for which we prove that its learned policy achieves non-asymptotic and approximately optimal performance guarantees.

Notations: For a set $\mathcal{X}$, we denote its cardinality as $|\mathcal{X}|$. The set of probability distribution over $\mathcal{X}$ is denoted as $\Delta(\mathcal{X})$, and its power set sigma algebra as $\Sigma(\mathcal{X})$. For any $x \in \mathbb{R}$, we denote $\max\{x, 0\}$ as $(x)_+$. For any function $f : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, state-action distribution $\nu \in \Delta(\mathcal{S} \times \mathcal{A})$, and real number $p \geq 1$, the ν-weighted p-norm of f is defined as $\|f\|_{p,\nu} = \mathbb{E}_{s,a\sim\nu}[|f(s,a)|^p]^{1/p}$.

4.2 Preliminaries

A Markov Decision Process (MDP) is a tuple $(\mathcal{S}, \mathcal{A}, r, P, \gamma, d_0)$, 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, $\gamma \in (0, 1)$ is the discount factor, and $d_0 \in \Delta(\mathcal{S})$ is the initial state distribution. The transition probability function $P_{s,a}(s')$ is the probability of transitioning to state s' when action a is taken at state s. In the literature, P is also called the *model* of the MDP. We consider a setting where $|\mathcal{S}|$ and $|\mathcal{A}|$ are finite but can be arbitrarily large. We will also assume that $r(s, a) \in [0, 1]$, for all $(s, a) \in \mathcal{S} \times \mathcal{A}$, without loss of generality. A policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ is a conditional distribution over actions given a state. The value function $V_{\pi,P}$ and the state-action value function $Q_{\pi,P}$ of a policy π for an MDP with model P are defined as

$$V_{\pi,P}(s) = \mathbb{E}_{\pi,P}[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s], \quad Q_{\pi,P}(s, a) = \mathbb{E}_{\pi,P}[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s, a_0 = a],$$

where the expectation is over the randomness induced by the policy π and model P. The optimal value function V_P^* and the optimal policy π_P^* of an MDP with the model P are defined as $V_P^* = \max_\pi V_{\pi,P}$ and $\pi_P^* = \operatorname{argmax}_\pi V_{\pi,P}$. The optimal state-action value function is given by $Q_P^* = \max_\pi Q_{\pi,P}$. The optimal policy can be obtained as $\pi_P^*(s) = \operatorname{argmax}_a Q_P^*(s,a)$. The discounted state-action occupancy of a policy π for an MDP with model P, denoted as $d_{\pi,P} \in \Delta(\mathcal{S} \times \mathcal{A})$, is defined as $d_{\pi,P}(s,a) = (1-\gamma)\mathbb{E}_{\pi,P}[\sum_{t=0}^{\infty} \gamma^t \mathbb{1}(s_t = s, a_t = a)]$.

Robust Markov Decision Process (RMDP): Unlike the standard MDP which considers a single model (transition probability function), the RMDP formulation considers a set of models. We refer to this set as the *uncertainty set* and denote it as $\mathcal{P}$. We consider $\mathcal{P}$ that satisfies the standard (s,a)*-rectangularity condition* [33]. We note that a similar uncertainty set can be considered for the reward function at the expense of additional notations. However, since the analysis will be similar and the sample complexity guarantee will be identical up to a constant factor, without loss of generality, we assume that the reward function is known and deterministic.

We specify an RMDP as $M = (\mathcal{S}, \mathcal{A}, r, \mathcal{P}, \gamma, d_0)$, where the uncertainty set $\mathcal{P}$ is typically defined as

$$\mathcal{P} = \otimes_{(s,a)\in\mathcal{S}\times\mathcal{A}} \mathcal{P}_{s,a}, \quad \text{where } \mathcal{P}_{s,a} = \{P_{s,a} \in \Delta(\mathcal{S}) \,:\, D(P_{s,a}, P^o_{s,a}) \le \rho\}, \tag{4.1}$$

$P^o = (P^o_{s,a}, (s,a) \in \mathcal{S} \times \mathcal{A})$ is the *nominal model*, $D(\cdot,\cdot)$ is a distance metric between two probability distributions, and $\rho > 0$ is the radius of the uncertainty set that indicates the level of robustness. The nominal model P^o can be thought as the model of the training environment. It is either the model of the simulator on which the (online) RL algorithm is trained, or in our setting, it is the model according to which the offline data is generated. The uncertainty set $\mathcal{P}$ (4.1) is the set of all valid transition probability functions (valid testing models) in the neighborhood of the nominal model P^o, which by definition satisfies (s,a)-rectangularity condition [33], where the neighborhood is defined using the distance metric $D(\cdot,\cdot)$ and radius ρ. In this work, we consider the *Total Variation (TV) uncertainty set* defined using the TV distance, i.e.,

$D(P_{s,a}, P^o_{s,a}) = (1/2)\|P_{s,a} - P^o_{s,a}\|_1$.

The RMDP problem is to find the optimal robust policy which maximizes the value against the worst possible model in the uncertainty set $\mathcal{P}$. The *robust value function* V^π corresponding to a policy π and the *optimal robust value function* V^* are defined as [33, 34]

$$V^\pi = \inf_{P \in \mathcal{P}} V_{\pi,P}, \qquad V^* = \sup_{\pi} \inf_{P \in \mathcal{P}} V_{\pi,P}. \tag{4.2}$$

The *optimal robust policy* π^* is such that the robust value function corresponding to it matches the optimal robust value function, i.e., $V^{\pi^*} = V^*$. It is known that there exists a deterministic optimal policy [33] for the RMDP. The *robust Bellman operator* is defined as [33]

$$(TQ)(s,a) = r(s,a) + \gamma \inf_{P_{s,a} \in \mathcal{P}_{s,a}} \mathbb{E}_{s' \sim P_{s,a}}[\max_b Q(s', b)]. \tag{4.3}$$

It is known that T is a contraction mapping in the infinity norm and hence it has a unique fixed point Q^* with $V^*(s) = \max_a Q^*(s,a)$ and $\pi^*(s) = \text{argmax}_a\, Q^*(s,a)$ [33]. The ***Robust Q-Iteration (RQI)*** can now be defined using the robust Bellman operator as $Q_{k+1} = TQ_k$. Since T is a contraction, it follows that $Q_k \to Q^*$. So, RQI can be used to compute (solving the planning problem) Q^* and π^* in the tabular setting with a known $\mathcal{P}$. Due to the optimization over the uncertainty set $\mathcal{P}_{s,a}$ for each (s,a) pair, solving the planning problem in RMDP using RQI is much more computationally intensive than solving it in MDP using Q-Iteration.

Offline RL: Offline RL considers the problem of learning the optimal policy of an MDP when the algorithm does not have direct access to the environment and cannot generate data samples in an online manner. For learning the optimal policy π^*_P of an MDP with model P, the algorithm will only have access to an offline dataset $\mathcal{D}_P = \{(s_i, a_i, r_i, s'_i)\}_{i=1}^N$, where $(s_i, a_i) \sim \mu$, $\mu \in \Delta(\mathcal{S} \times \mathcal{A})$ is some distribution, and $s'_i \sim P_{s_i,a_i}$. ***Fitted Q-Iteration (FQI)*** is a popular offline RL approach which is amenable to theoretical analysis while achieving impressive empirical performance. In addition to the dataset $\mathcal{D}_P$, FQI uses a function class $\mathcal{F} = \{f : \mathcal{S} \times \mathcal{A} \to [0, 1/(1-\gamma)]\}$ to approximate Q^*_P. The typical FQI update is given by $f_{k+1} = \text{argmin}_{f \in \mathcal{F}} \sum_{i=1}^N (r(s_i, a_i) +$

$\gamma \max_b f_k(s_i', b) - f(s_i, a_i))^2$, which aims to approximate the non-robust Bellman update using offline data with function approximation. Under suitable assumptions, it is possible to obtain provable performance guarantees for FQI [105, 91, 103].

4.3 Offline Robust Reinforcement Learning

The goal of an offline robust RL algorithm is to learn the optimal robust policy π^* using a pre-collected offline dataset $\mathcal{D}$. The data is typically generated according to a nominal (training) model P^o, i.e., $\mathcal{D} = \{(s_i, a_i, r_i, s_i')\}_{i=1}^N$, where $(s_i, a_i) \sim \mu, \mu \in \Delta(\mathcal{S} \times \mathcal{A})$ is some data generating distribution, and $s_i' \sim P^o_{s_i,a_i}$. The uncertainty set $\mathcal{P}$ is defined around this nominal model P^o as given in (4.1) w.r.t. the total variation distance metric. We emphasize that the learning algorithm does not know the nominal model P^o as it has only access to $\mathcal{D}$, and hence it also does not know $\mathcal{P}$. Moreover, the learning algorithm does not have data generated according to any other models in $\mathcal{P}$ and has to rely only on $\mathcal{D}$ to account for the behavior w.r.t. all models in $\mathcal{P}$.

Learning policies for RL problems with large state-action spaces is computationally intractable. RL algorithms typically overcome this issue by using function approximation. In this paper, we consider two function classes $\mathcal{F} = \{f : \mathcal{S} \times \mathcal{A} \to [0, 1/(1-\gamma)]\}$ and $\mathcal{G} = \{g : \mathcal{S} \times \mathcal{A} \to [0, 1/(1-\gamma)]\}$. We use $\mathcal{F}$ to approximate Q^* and $\mathcal{G}$ to approximate the dual variable functions which we will introduce in the next section. For simplicity, we will first assume that these function classes are finite but exponentially large, and we will use the standard log-cardinality to characterize the sample complexity results, as given in Theorem 8. We note that, at the cost of additional notations and analysis, infinite function classes can also be considered where the log-cardinalities are replaced by the appropriate notions of covering number.

Similar to the non-robust offline RL, we make the following standard assumptions about the data generating distribution μ and the representation power of $\mathcal{F}$.

Assumption 4 (Concentratability)**.** *There exists a finite constant $C > 0$ such that for any $\nu \in \{d_{\pi,P^o} \mid$ any policy $\pi\} \subseteq \Delta(\mathcal{S} \times \mathcal{A})$, we have $\|\nu/\mu\|_\infty \leq \sqrt{C}$.*

Assumption 4 states that the ratio of the distribution ν and the data generating distribution μ,

$\nu(s,a)/\mu(s,a)$, is uniformly bounded. This assumption is widely used in the offline RL literature [67, 106, 91, 55, 94] in many different forms. We borrow this assumption from [91], where they used it for non-robust offline RL. In particular, we note that the distribution ν is in the collection of discounted state-action occupancies on model P^o alone for the robust RL.

Assumption 5 (Approximate completeness)**.** *Let $\mu \in \Delta(\mathcal{S} \times \mathcal{A})$ be the data distribution. Then,* $\sup_{f \in \mathcal{F}} \inf_{f' \in \mathcal{F}} \|f' - Tf\|_{2,\mu}^2 \leq \varepsilon_{\mathrm{c}}$.

Assumption 5 states that the function class $\mathcal{F}$ is approximately closed under the robust Bellman operator T. This assumption has also been widely used in the offline RL literature [106, 91, 55, 94].

One of the most important properties that the function class $\mathcal{F}$ should have is that there must exist a function $f' \in \mathcal{F}$ which well-approximates Q^*. This assumption is typically called *approximate realizability* in the offline RL literature. This is typically formalized by assuming $\inf_{f \in \mathcal{F}} \|f - Tf\|_{2,\mu}^2 \leq \varepsilon_{\mathrm{r}}$ [91]. It is known that the approximate completeness assumption and the concentratability assumption imply the realizability assumption [91, 94].

4.4 Robust Fitted Q-Iteration: Algorithm and Main Results

In this section, we give a step-by-step approach to overcome the challenges of the offline robust RL outlined in Section 4.1. We then combine these intermediate steps to obtain our proposed RFQI algorithm. We then present our main result about the performance guarantee of the RFQI algorithm, followed by a brief description about the proof approach.

4.4.1 Dual Reformulation of Robust Bellman Operator

One key challenge in directly using the standard definition of the optimal robust value function given in (4.2) or of the robust Bellman operator given in (4.3) for developing and analyzing robust RL algorithms is that both involve computing an expectation w.r.t. each model $P \in \mathcal{P}$. Given that the data is generated only according to the nominal model P^o, estimating these expectation values is really challenging. We show that we can overcome this difficulty through the dual reformulation of the robust Bellman operator, as given below.

Proposition 6. *Let M be an RMDP with the uncertainty set $\mathcal{P}$ specified by* (4.1) *using the total variation distance $D(P_{s,a}, P^o_{s,a}) = (1/2)\|P_{s,a} - P^o_{s,a}\|_1$. Then, for any $Q : \mathcal{S} \times \mathcal{A} \to [0, 1/(1-\gamma)]$, the robust Bellman operator T given in* (4.3) *can be equivalently written as*

$$(TQ)(s,a) = r(s,a) - \gamma \inf_{\eta \in [0, \frac{1}{(1-\gamma)}]} (\mathbb{E}_{s' \sim P^o_{s,a}}[(\eta - V(s'))_+] - \eta + \rho(\eta - \inf_{s''} V(s''))_+), \quad (4.4)$$

where $V(s) = \max_{a \in \mathcal{A}} Q(s,a)$. Moreover, the inner optimization problem in (4.4) *is convex in η.*

Note that in (4.4), the expectation is now only w.r.t. the nominal model P^o, which opens up the possibility of using empirical estimates obtained from the data generated according to P^o. This avoids the need to use importance sampling based techniques to account for all models in $\mathcal{P}$, which often have high variance, and thus, are not desirable.

While (4.4) provides a form that is amenable to estimation using offline data, it involves finding $\inf_{s''} V(s'')$. Though this computation is straightforward in a tabular setting, it is infeasible in a function approximation setting. In order to overcome this issue, we make the following assumption.

Assumption 6 (Fail-state)**.** *The RMDP M has a 'fail-state' s_f, such that $r(s_f, a) = 0$ and $P_{s_f,a}(s_f) = 1,\ \forall a \in \mathcal{A},\ \forall P \in \mathcal{P}$.*

We note that this is not a very restrictive assumption because such a 'fail-state' is quite natural in most simulated or real-world systems. For example, a state where a robot collapses and is not able to get up, either in a simulation environment like MuJoCo or in real-world setting, is such a fail state.

Assumption 6 immediately implies that $V_{\pi,P}(s_f) = 0,\ \forall P \in \mathcal{P}$, and hence $V^*(s_f) = 0$ and $Q^*(s_f, a) = 0,\ \forall a \in \mathcal{A}$. It is also straightforward to see that $Q_{k+1}(s_f, a) = 0,\ \forall a \in \mathcal{A}$, where Q_k's are the RQI iterates given by the robust Bellman update $Q_{k+1} = TQ_k$ with the initialization $Q_0 = 0$. By the contraction property of T, we have $Q_k \to Q^*$. So, under Assumption 6, without loss of generality, we can always keep $Q_k(s_f, a) = 0,\ \forall a \in \mathcal{A}$ and for all k in RQI (and later in RFQI). So, in the light of the above description, for the rest of the paper we will use the robust

Bellman operator T by setting $\inf_{s''} V(s'') = 0$. In particular, for any function $f : \mathcal{S} \times \mathcal{A} \to [0, 1/(1-\gamma)]$ with $f(s_f, a) = 0$, the robust Bellman operator T is now given by

$$(Tf)(s,a) = r(s,a) - \gamma \inf_{\eta \in [0, \frac{1}{((1-\gamma))}]} (\mathbb{E}_{s' \sim P^o_{s,a}}[(\eta - \max_{a'} f(s', a'))_+] - (1-\rho)\eta). \tag{4.5}$$

4.4.2 Approximately Solving the Dual Optimization using Empirical Risk Minimization

Another key challenge in directly using the standard definition of the optimal robust value function given in (4.2) or of the robust Bellman operator given in (4.3) for developing and analyzing robust RL algorithms is that both involve an optimization over $\mathcal{P}$. The dual reformulation given in (4.5) partially overcomes this challenge also, as the optimization over $\mathcal{P}$ is now replaced by a convex optimization over a scalar $\eta \in [0, 2/(\rho(1-\gamma))]$. However, this still requires solving an optimization for each $(s,a) \in \mathcal{S} \times \mathcal{A}$, which is clearly infeasible even for moderately sized state-action spaces, not to mention the function approximation setting. Our key idea to overcome this difficulty is to reformulate this as a functional optimization problem instead of solving it as multiple scalar optimization problems. This functional optimization method will make it amenable to approximately solving the dual problem using an empirical risk minimization approach with offline data.

Consider the probability (measure) space $(\mathcal{S} \times \mathcal{A}, \Sigma(\mathcal{S} \times \mathcal{A}), \mu)$ and let $L^1(\mathcal{S} \times \mathcal{A}, \Sigma(\mathcal{S} \times \mathcal{A}), \mu)$ be the set of all absolutely integrable functions defined on this space.* In other words, L^1 is the set of all functions $g : \mathcal{S} \times \mathcal{A} \to \mathcal{C} \subset \mathbb{R}$, such that $\|g\|_{1,\mu}$ is finite. We set $\mathcal{C} = [0, 1/(1-\gamma)]$, anticipating the solution of the dual optimization problem (4.5). We also note μ is the data generating distribution which is a σ-finite measure.

For any given function $f : \mathcal{S} \times \mathcal{A} \to [0, 1/(1-\gamma)]$, we define the loss function $L_{\text{dual}}(\cdot; f)$ as

$$L_{\text{dual}}(g; f) = \mathbb{E}_{s,a \sim \mu}[\mathbb{E}_{s' \sim P^o_{s,a}}[(g(s,a) - \max_{a'} f(s', a'))_+] - (1-\rho) g(s,a)], \quad \forall g \in L^1. \tag{4.6}$$

*In the following, we will simply denote $L^1(\mathcal{S} \times \mathcal{A}, \Sigma(\mathcal{S} \times \mathcal{A}), \mu)$ as L^1 for conciseness.

In the following lemma, we show that the scalar optimization over η for each (s, a) pair in (4.5) can be replaced by a single functional optimization w.r.t. the loss function L_{dual}.

Lemma 5. *Let L_{dual} be the loss function defined in (4.6). Then, for any function $f : \mathcal{S} \times \mathcal{A} \to [0, 1/(1-\gamma)]$, we have*

$$\inf_{g \in L^1} L_{\text{dual}}(g; f) = \mathbb{E}_{s,a\sim\mu}\Big[\inf_{\eta \in [0, \frac{1}{(1-\gamma)}]} \Big(\mathbb{E}_{s'\sim P^o_{s,a}}\big[\big(\eta - \max_{a'} f(s', a')\big)_+\big] - (1-\rho)\eta\Big)\Big]. \tag{4.7}$$

Note that the RHS of (4.7) has minimization over η for each (s, a) pair and minimization is inside the expectation $\mathbb{E}_{s,a\sim\mu}[\cdot]$. However, the LHS of (4.7) has a single functional minimization over $g \in L^1$ and this minimization is outside the expectation. For interchanging the expectation and minimization, and for moving from point-wise optimization to functional optimization, we use the result from [107, Theorem 14.60], along with the fact that L^1 is a decomposable space. We also note that this result has been used in many recent works on distributionally robust optimization [108, 109] (see Appendix C.1 for more details).

We can now define the empirical loss function $\widehat{L}_{\text{dual}}$ corresponding to the true loss L_{dual} as

$$\widehat{L}_{\text{dual}}(g; f) = \frac{1}{N}\sum_{i=1}^{N} (g(s_i, a_i) - \max_{a'} f(s'_i, a'))_+ - (1-\rho) g(s_i, a_i). \tag{4.8}$$

Now, for any given f, we can find an approximately optimal dual function through the *empirical risk minimization* approach as $\inf_{g\in L^1} \widehat{L}_{\text{dual}}(g; f)$.

As we mentioned in Section 4.3, our offline robust RL algorithm is given an input function class $\mathcal{G} = \{g : \mathcal{S} \times \mathcal{A} \to [0, 1/(1-\gamma)]\}$ to approximate the dual variable functions. So, in the empirical risk minimization, instead of taking the infimum over all the functions in L^1, we can only take the infimum over all the functions in $\mathcal{G}$. For this to be meaningful, $\mathcal{G}$ should have sufficient representation power. In particular, the result in Lemma 5 should hold approximately even if we replace the infimum over L^1 with infimum over $\mathcal{G}$. One can see that this is similar to the realizability requirement for the function class $\mathcal{F}$ as described in Section 4.3. We formalize the representation power of $\mathcal{G}$ in the following assumption.

Assumption 7 (Approximate dual realizability)**.** *For all $f \in \mathcal{F}$, there exists a uniform constant ε_{dual} such that* $\inf_{g\in\mathcal{G}} L_{\mathrm{dual}}(g; f) - \inf_{g\in L^1} L_{\mathrm{dual}}(g; f) \leq \varepsilon_{dual}$

Using the above assumption, for any given $f \in \mathcal{F}$, we can find an approximately optimal dual function $\widehat{g}_f \in \mathcal{G}$ through the *empirical risk minimization* approach as $\widehat{g}_f = \operatorname{argmin}_{g\in\mathcal{G}} \widehat{L}_{\mathrm{dual}}(g; f)$.

In order to characterize the performance of this approach, consider the operator T_g for any $g \in \mathcal{G}$ as

$$(T_g f)(s, a) = r(s, a) - \gamma(\mathbb{E}_{s'\sim P^o_{s,a}}[(g(s, a) - \max_{a'} f(s', a'))_+] - (1 - \rho)g(s, a)), \tag{4.9}$$

for all $f \in \mathcal{F}$ and $(s, a) \in \mathcal{S} \times \mathcal{A}$. We will show in Lemma 28 in Appendix C.3 that the error $\sup_{f\in\mathcal{F}} \|Tf - T_{\widehat{g}_f} f\|_{1,\mu}$ is $\mathcal{O}(\log(|\mathcal{F}|/\delta)/\sqrt{N})$ with probability at least $1 - \delta$.

4.4.3 Robust Fitted Q-iteration

The intuitive idea behind our robust fitted Q-iteration (RFQI) algorithm is to approximate the exact RQI update step $Q_{k+1} = TQ_k$ with function approximation using offline data. The exact RQI step requires updating each (s, a)-pair separately, which is not scalable to large state-action spaces. So, this is replaced by the function approximation as $Q_{k+1} = \operatorname{argmin}_{f\in\mathcal{F}} \|TQ_k - f\|^2_{2,\mu}$. It is still infeasible to perform this update as it requires to exactly compute the expectation (w.r.t. P^o and μ) and to solve the dual problem accurately. We overcome these issues by replacing both these exact computations with empirical estimates using the offline data. We note that this intuitive idea is similar to that of the FQI algorithm in the non-robust case. However, RFQI has unique challenges due to the nature of the robust Bellman operator T and the presence of the dual optimization problem within T.

Given a dataset $\mathcal{D}$, we also follow the standard non-robust offline RL choice of least-squares residual minimization [91, 94, 55]. Define the empirical loss of f given f' (which represents the

Q-function from the last iteration) and dual variable function g as

$$\widehat{L}_{\mathrm{RFQI}}(f; f', g) = \frac{1}{N}\sum_{i=1}^{N}\Big(\begin{array}{r} r(s,a) + \gamma\big(-(g(s_i,a_i) - \max_{a'} f'(s_i', a'))_+ \\ + (1-\rho) g(s_i, a_i)\big) - f(s_i, a_i) \end{array} \Big)^2. \tag{4.10}$$

The correct dual variable function to be used in (4.10) is the optimal dual variable $g^*_{f'} = \mathrm{argmin}_{g\in\mathcal{G}}$ $L_{\mathrm{dual}}(g; f')$ corresponding to the last iterate f', which we will approximate it by $\widehat{g}_{f'} = \mathrm{argmin}_{g\in\mathcal{G}}$ $\widehat{L}_{\mathrm{dual}}(g; f')$. The RFQI update is then obtained as $\mathrm{argmin}_{f\in\mathcal{F}} \widehat{L}_{\mathrm{RFQI}}(f; f', \widehat{g}_{f'})$.

Summarizing the individual steps described above, we formally give our RFQI algorithm below.

Algorithm 3 Robust Fitted Q-Iteration (RFQI) Algorithm

1: **Input:** Offline dataset $\mathcal{D} = (s_i, a_i, r_i, s_i')_{i=1}^N$, function classes $\mathcal{F}$ and $\mathcal{G}$.

2: **Initialize:** $Q_0 \equiv 0 \in \mathcal{F}$.

3: **for** $k = 0, \cdots, K-1$ **do**

4: **Dual variable function optimization:** Compute the dual variable function corresponding to Q_k through empirical risk minimization as $g_k = \widehat{g}_{Q_k} = \mathrm{argmin}_{g\in\mathcal{G}} \widehat{L}_{\mathrm{dual}}(g; Q_k)$ (see (4.8)).

5: **Robust Q-update:** Compute the next iterate Q_{k+1} through least-squares regression as $Q_{k+1} = \mathrm{argmin}_{Q\in\mathcal{F}} \widehat{L}_{\mathrm{RFQI}}(Q; Q_k, g_k)$ (see (4.10)).

6: **end for**

7: **Output:** $\pi_K = \mathrm{argmax}_a Q_K(s,a)$

Now we state our main theoretical result on the performance of the RFQI algorithm.

Theorem 8. *Let Assumptions 4-7 hold. Let π_K be the output of the RFQI algorithm after K iterations. Denote $J^\pi = \mathbb{E}_{s\sim d_0}[V^\pi(s)]$ where d_0 is initial state distribution. Then, for any $\delta \in$*

$(0,1)$, *with probability at least* $1-2\delta$, *we have*

$$J^{\pi^*} - J^{\pi_K} \leq \frac{\gamma^K}{(1-\gamma)^2} + \frac{\sqrt{C}(\sqrt{6\varepsilon_c} + \gamma\varepsilon_{dual})}{(1-\gamma)^2} + \frac{10}{(1-\gamma)^3}\sqrt{\frac{18C\log(2|\mathcal{F}||\mathcal{G}|/\delta)}{N}}.$$

Remark 5. *Theorem 8 states that the RFQI algorithm can achieve approximate optimality. To see this, note that with* $K \geq \mathcal{O}(\frac{1}{\log(1/\gamma)}\log(\frac{1}{\varepsilon(1-\gamma)}))$*, and neglecting the second term corresponding to (inevitable) approximation errors* ε_c *and* ε_{dual}*, we get* $J^{\pi^*} - J^{\pi_K} \leq \varepsilon/(1-\gamma)$ *with probability greater than* $1-2\delta$ *for any* $\varepsilon, \delta \in (0,1)$*, as long as the number of samples* $N \geq \mathcal{O}(\frac{1}{\varepsilon^2(1-\gamma)^4}\log\frac{|\mathcal{F}||\mathcal{G}|}{\delta})$*. So, the above theorem can also be interpreted as a* ***sample complexity*** *result.*

Remark 6. *The known sample complexity of robust-RL in the tabular setting is* $\widetilde{\mathcal{O}}(\frac{|\mathcal{S}|^2|\mathcal{A}|}{\varepsilon^2(1-\gamma)^4})$ *[83, 2]. Considering* $\widetilde{\mathcal{O}}(\log(|\mathcal{F}||\mathcal{G}|))$ *to be* $\widetilde{\mathcal{O}}(|\mathcal{S}||\mathcal{A}|)$*, we can recover the same bound as in the tabular setting (we save* $|\mathcal{S}|$ *due to the use of Bernstein inequality).*

Remark 7. *Under similar Bellman completeness and concentratability assumptions, RFQI sample complexity is comparable to that of a non-robust offline RL algorithm, that is,* $\mathcal{O}(\frac{1}{\varepsilon^2(1-\gamma)^4}\log\frac{|\mathcal{F}|}{\delta})$ *[91]. As a consequence of robustness, we have* ρ^{-2} *and* $\log(|\mathcal{G}|)$ *factors in our bound.*

4.4.4 Proof Sketch

Here we briefly explain the key ideas used in the analysis of RFQI for obtaining the optimality gap bound in Theorem 8. The complete proof is provided in Appendix C.3.

Step 1: To bound $J^{\pi^*} - J^{\pi_K}$, we connect it to the error $\|Q^{\pi^*} - Q_K\|_{1,\nu}$ for any state-action distribution ν. While the similar step follows almost immediately using the well-known performance lemma in the analysis of non-robust FQI, such a result is not known in the robust RL setting. So, we derive the basic inequalities to get a recursive form and to obtain the bound $J^{\pi^*} - J^{\pi_K} \leq 2\|Q^{\pi^*} - Q_K\|_{1,\nu}/(1-\gamma)$ (see (C.12) and the steps before in Appendix C.3).

Step 2: To bound $\|Q^{\pi^*} - Q_K\|_{1,\nu}$ for any state-action distribution ν such that $\|\nu/\mu\|_\infty \leq \sqrt{C}$, we decompose it to get a recursion, with approximation terms based on the least-squares regres-

sion and empirical risk minimization. Recall that $\widehat{g}_f$ is the dual variable function from the algorithm for state-action value function $f \in \mathcal{F}$. Denote $\widehat{f}_g$ as the least squares solution from the algorithm for the state-action value function $f \in \mathcal{F}$ and dual variable function $g \in \mathcal{G}$, i.e., $\widehat{f}_g = \text{argmin}_{Q \in \mathcal{F}} \widehat{L}_{\text{RFQI}}(Q; f, g)$. By recursive use of the obtained inequality (C.13) (see Appendix C.3) and using uniform bound, we get

$$\|Q^{\pi^*} - Q_K\|_{1,\nu} \leq \frac{\gamma^K}{1-\gamma} + \frac{\sqrt{C}}{1-\gamma} \sup_{f \in \mathcal{F}} \|Tf - T_{\widehat{g}_f} f\|_{1,\mu} + \frac{\sqrt{C}}{1-\gamma} \sup_{f \in \mathcal{F}} \sup_{g \in \mathcal{G}} \|T_g f - \widehat{f}_g\|_{2,\mu}.$$

Step 3: We recognize that $\sup_{f \in \mathcal{F}} \|Tf - T_{\widehat{g}_f} f\|_{1,\mu}$ is an empirical risk minimization error term. Using Rademacher complexity based bounds, we show in Lemma 28 that this error is $\mathcal{O}(\log(|\mathcal{F}|/\delta)/\sqrt{N})$ with high probability.

Step 4: Similarly, we also recognize that $\sup_{f \in \mathcal{F}} \sup_{g \in \mathcal{G}} \|T_g f - \widehat{f}_g\|_{2,\mu}$ is a least-squares regression error term. We also show that this error is $\mathcal{O}(\log(|\mathcal{F}||\mathcal{G}|/\delta)/\sqrt{N})$ with high probability. We adapt the generalized least squares regression result to accommodate the modified target functions resulting from the robust Bellman operator to obtain this bound (see Lemma 29).

The proof is complete after combining steps 1-4 above.

4.5 Experiments

Here, we demonstrate the robust performance of our RFQI algorithm by evaluating it on *Cartpole* and *Hopper* environments in OpenAI Gym [71]. In all the figures shown, the quantity in the vertical axis is averaged over 20 different seeded runs depicted by the thick line and the band around it is the ± 0.5 standard deviation. We provide our code in a **github repository** https://github.com/zaiyan-x/RFQI containing instructions to reproduce all results in this paper. We also provide more detailed description of the experiments and more results on additional experiments in Appendix C.5.

For the *Cartpole*, we compare RFQI algorithm against the non-robust RL algorithms FQI and DQN, and the soft-robust RL algorithm proposed in [48]. We test the robustness of the algorithms by changing the parameter *force_mag* (to model external force disturbance), and also by introduc-

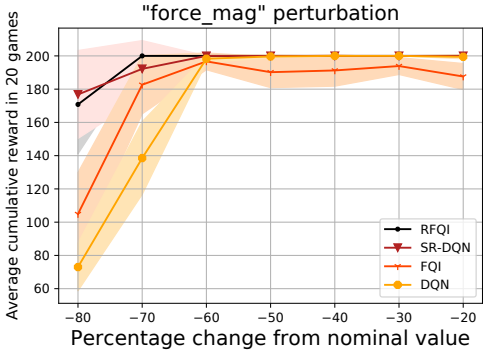

Figure 4.1: CartPole: Performance of RFQI algorithm with *force-mag* sim2real parameter. (Reprinted from [3])

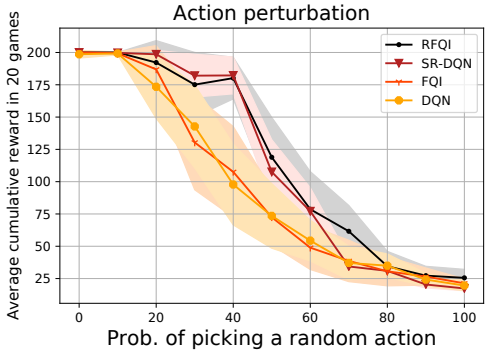

Figure 4.2: CartPole: Performance of RFQI algorithm with random action sim2real parameter. (Reprinted from [3])

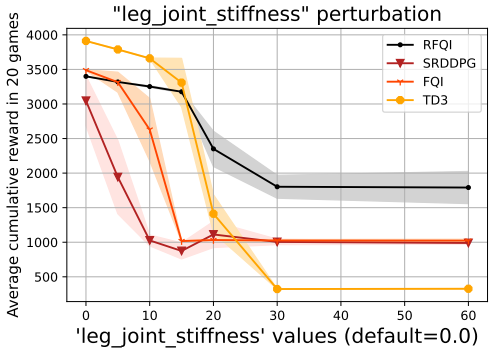

Figure 4.3: Hopper: Performance of RFQI algorithm with *leg_joint_stiffness* sim2real parameter. (Reprinted from [3])

ing action perturbations (to model actuator noise). Fig. 4.1 and Fig. 4.2 shows superior robust performance of RFQI compared to the non-robust FQI and DQN. The RFQI performance is similar to that of soft-robust DQN. We note that soft-robust RL algorithm (here soft-robust DQN) is an online deep RL algorithm (and not an offline RL algorithm) and has no provable performance guarantee. Moreover, soft-robust RL algorithm requires generating online data according a number of models in the uncertainty set, whereas RFQI only requires offline data according to a single nominal training model.

For the *Hopper*, we compare RFQI algorithm against the non-robust RL algorithms FQI and

TD3 [110], and the soft-robust RL (here soft-robust DDPG) algorithm proposed in [48]. We test the robustness of the algorithms by changing the parameter *leg_joint_stiffness*. Fig. 4.3 shows the superior performance of our RFQI algorithm against the non-robust algorithms and soft-robust DDPG algorithm. The average episodic reward of RFQI remains almost the same initially, and later decays much less and gracefully when compared to the non-robust FQI and TD3.

4.6 Conclusion

In this work, we presented a novel robust RL algorithm called Robust Fitted Q-Iteration algorithm with provably optimal performance for an RMDP with arbitrarily large state space, using only offline data with function approximation. We also demonstrated the superior performance of the proposed algorithm on standard benchmark problems.

One limitation of our present work is that, we considered only the uncertainty set defined with respect to the total variation distance. In future work, we will consider uncertainty sets defined with respect to other f-divergences such as KL-divergence and Chi-square divergence. Finding a lower bound for the sample complexity and relaxing the assumptions used are also important and challenging problems.

5. IMPROVING BEHAVIORAL CLONING WITH DISTRIBUTIONALLY ROBUST OPTIMIZATION*

In this chapter, we focus on leveraging a distributionally robust optimization tool in imitation learning problems to alleviate covariate shift and system model perturbations.

5.1 Introduction

A child, a dog, or even a reptile is capable of learning through imitation [111]. Such intuitive way of learning naturally extends from animal's survival instincts to solving potentially complicated control tasks. Hence, it serves as the primary philosophy underlying most, if not all, methods in Imitation Learning, a very fundamental reinforcement learning (RL) setting in which the goal is to learn a control policy exclusively from expert demonstrations. However simple and fundamental the idea of Imitation Learning may sound, it is by no means the easy fix for everything. In fact, although it is quite clear its strength is the simplicity, its drawbacks are equally straightforward: an imitator merely mimics, and once it encounters anything unseen in the expert demonstration, it hardly knows how to cope with the new challenges. Mistakes compound in the decision horizon, making the learner never recover from the moment it deviates from the expert trajectories.

Imitation Learning: Learning through imitation can be traced back to as early as [112]. Imitation learning assumes access to only expert demonstrations. This has given rise to the most natural approach behavior cloning [113], which is a supervised learning method, and learns by simply minimizing differences between the actions of the learners and those of the experts. In this approach, once the imitator enters an uncharted territory, its mistakes compound such that it never recovers after since. Named covariate shift, this issue has been long studied. [114] studied the behavior cloning approach and characterized a tight bound on sub-optimality gap of order $O(\varepsilon H^2)$. Most of works in imitation learning then try to improve this bound with additional assumptions. [115] proposed DAGGER and showed that the above bound on the sub-optimality gap can be improved

*Reprinted with permission from Kishan Panaganti, Zaiyan Xu, Dileep Kalathil, Mohammad Ghavamzadeh, "Improving Behavioral Cloning with Distributionally Robust Optimization." Preprint, 2023.

to $O(\varepsilon u H)$, where u is the cost of taking different action than the expert at one step and following the expert's suggestion afterward, by querying the expert and interacting with the environment. In worse cases, u can be as large as H. DRIL [116] needs environment interactions but no expert query, and achieves a sub-optimality gap bound that is linear in H. These works are closes to ours as we also aim to mitigate the covariate shift issue both theoretically and empirically. However, they are also different from our work in that we try to mitigate the covariate shift issue by only bootstrapping around the expert demonstrations and not asking for additional offline data or the ability to query. In this aspect, we are very close to the classical behavior cloning. A recent work [117] showed that the lower bound $\Omega(\varepsilon H)$ on sub-optimality gap for the known model P^o setting. Access to the environment essentially allows limited queries and does not have the full power of the known model setting. Still, the ability to interact with the environment helps to mitigate the covariate shift issue. For example, GAIL [118] uses a discriminator network to distinguish expert states from those visited by learner's policy. [119] provides a game theoretic framework to mitigate covariate shift that naturally competes with noisy expert policies.

Inverse RL (IRL): This is a framework where an agent learns the underlying true reward function of an expert and uses it in the usual RL algorithm to produce a policy that imitates an expert. Notable works include [120, 51, 121, 122, 52]. Although this is not the setting we consider, there are works trying to mitigate the covariate shift issue in the imitation learning by using inverse RL methodologies. [123] proposed MIMIC-MD which estimates the expert trajectory distribution and gets a tight sub-optimality bound, but their algorithm is not practically implementable as is. [124] proposed MILO which uses an additional offline dataset to estimate environment dynamics and has great empirical performance when trained with extremely limited expert demonstration where BC fails to produce any working policy. [125] proposed an IRL algorithm to learn cost functions that are robust in noisy systems. These are all inherently adversarial approaches which are using critics to disturb the underlying systems or to distinguish and pick good reward representations.

Robust RL: The framework of the robust Markov decision process (RMDP) [33, 34] addresses the problem of learning a policy that is robust against model mismatches between the training and

testing environments. This is the goal of distributionally robust reinforcement learning (DR-RL). Robust RL finds applications in many real-world evolving systems where there is always a gap to the simulator model. Deploying naive RL policies [126] can be catastrophic when there is such obvious differences in the models. The RMDP problem is well-studied. [37, 36, 38, 72, 73] have investigated various types of uncertainty sets and sought tractable methods to solve RMDP. [127, 128, 35] study the sample complexity of model-based robust RL algorithms in a tabular setting using a generative model, which is a strong oracle enabling learners to query arbitrary transitions. [1] developed a model-free online robust RL algorithm with linear function approximation to tackle potentially infinite state spaces. [104] proposed an online robust Q-learning with R-contamination uncertainty set. Many of the optimization techniques and analyses were developed in the context of supervised learning [109, 129, 130, 131, 132, 133].

Main Contributions: We summarize our contributions in this paper as follows and refer to the relevant sections:

(i) We propose Distributionally Robust Behavioral Cloning (DR-BC) algorithm, a novel imitation learning algorithm that utilizes DRO methodology. Despite suffering from the covariate shift issue, the BC method is computationally efficient, which makes any other imitation learning algorithm falls short on. Contrarily, IL optimization mitigates the covariate shift issue but needs possibly infinite computational access. Our proposed method cleverly bridges these two methods shining light upon their positive aspects while suppressing bad characteristics. We discuss this in Section 5.2.

(ii) We introduce the problem of robust imitation learning for the model parameter mismatches. In this work, we consider mismatch in system transition dynamics. Robust learning in RL is studied widely but not in imitation learning. We only know of [134], but it is in the IRL framework for making fair comparison. Critical real-world applications like power systems, healthcare, self-driving automobiles, etc have guidance from expert across diverse scenarios [135, 136, 137]. We propose a robust imitation learning algorithm which is closely related to our DR-BC algorithm from (i) to address the model mismatch. We also give its theoretical guarantees. We discuss further in Section 5.3.

(iii) We make significant theoretical contributions in this paper. As per our knowledge, we are the first to use the DRO technique in imitation learning. Although we provide intuitive remarks for the DR-BC method, DRO alone is not the key. We had to bring together many different techniques from literature in RL on policy gradient methods, statistical theory, class of instances where covariate shift issue is mitigated and robust for model changes, to finally showcase the prowess of the DR-BC sub-optimality result. We provide detailed proofs for our results in Appendices.
(iv) We perform extensive simulations on three notable continuous-action OpenAI [71] Gym MuJoCo [138] environments. We show that the learned DR-BC policy is able to mitigate the covariate shift issue. We also demonstrate that the DR-BC policy is robust against model perturbations in that when BC has catastrophic drop in performance, DR-BC weathers model mismatches for much severer model perturbations.

5.2 Imitation Learning

In this section, we formally introduce the imitation learning problem using the MDP framework. We then discuss the issue of covariate shift that the BC algorithm suffers from. We also discuss the optimality of the original imitation learning optimization method by [115] and its drawbacks. We then propose the DR-BC algorithm to address the covariate shift issue in imitation learning using the DRO technique and discuss its theoretical guarantees.

5.2.1 Problem Formulation

The goal of Imitation Learning (IL) in sequential decision-making [116, 117] is to *imitate* an expert's policy using only demonstrations generated by its interactions with an environment. More formally, consider an infinite-horizon Markov decision process (MDP), denoted by the tuple $\{\mathcal{S}, \mathcal{A}, P^o, \gamma, r, \mu\}$, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces, $P^o : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition dynamics (*model*) of the environment, γ is a discount factor, μ is the initial state distribution, and $r : \mathcal{S} \times \mathcal{A} \to [0, 1]$ is the true reward function (unknown to the learner). In this paper, we consider a system with finite actions and a large state space. A stochastic policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ maps states to distributions over actions. For any policy π, the value function of an

initial state $s_0 \sim \mu$ is given by $V_\pi(s_0) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | a_t \sim \pi(\cdot|s_t), s_{t+1} \sim P^o_{s_t,a_t}]$. We denote $V_\pi = \mathbb{E}_{s_0 \sim \mu} V_\pi(s_0)$ and drop the explicit dependence on μ going forward for notation simplicity.

For any policy π, denote $d^\pi_{P^o} \in \Delta(\mathcal{S})$ as the state distribution of π under the evaluation of model P^o with initial state picked from μ. In this section, we simply denote such state-distributions as d^π and make explicit dependence on P^o where brevity is needed. Formally, let $\Pr_t(s|\pi, s_0 \sim \mu)$ be the probability of visiting state $s \in \mathcal{S}$ at time t following policy π on model P^o starting at initial state $s_0 \sim \mu$. Then, the state distribution of π is $d^\pi(s) = (1-\gamma) \sum_{t=0}^{\infty} \gamma^t \Pr_t(s|\pi, s_0 \sim \mu, P^o)$. We can now rewrite the value of π as $V_\pi = \mathbb{E}_{s \sim d^\pi, a \sim \pi}[r(s,a)]/(1-\gamma)$.

In the vanilla IL setting, the true reward function r is unknown to the learner. We instead have the dataset generated by rolling an expert policy (which is unknown to the learner) specified by $\pi_e : \mathcal{S} \to \Delta(\mathcal{A})$. Concisely speaking, we have an expert dataset in the form of i.i.d. tuples $\mathcal{D}_e = \{s_i, a_i\}_{i=1}^N$ sampled from state distribution $d^{\pi_e}_{P^o}$. We let $\pi_e \in \Pi$, where Π is the class of stochastic policies such that $V_{\pi_e} \geq V_\pi$ for all $\pi \in \Pi$. **The IL problem**: The goal of an IL algorithm is to output a policy $\widehat{\pi}$ that imitates the expert policy π_e by satisfying $V_{\widehat{\pi}} \approx V_{\pi_e}$.

Now, we assume a specific structure for the expert policy to be used for the imitation learning problem. Towards this, we first define the *entropy regularized* value function. For any policy π and $\tau > 0$, $\tilde{V}_\pi = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \tilde{r}(s_t, a_t) | a_t \sim \pi(\cdot|s_t), s_{t+1} \sim P^o_{s_t,a_t}]$, where $\tilde{r}(s,a) = r(s,a) - \tau \log \pi(a|s)$. The optimal policy, for any $\tau > 0$, is given by $\pi^*_\tau = \operatorname{argmax}_{\pi \in \Pi} \tilde{V}_\pi$. Our main motivation using the *entropy regularized* class of MDPs is its usefulness in improving the regret bounds on factor H in online policy gradient algorithms [139, 140]. We also let $\tau' = \tau \log(|\mathcal{A}|)$ for presenting our results. We now make the following assumption for our results in Section 5.2.

Assumption 8. $\pi_e = \pi^*_\tau$ *for an* unknown *small* $\tau \in (0, \varepsilon_\tau]$.

This may appear restrictive but there are many real-world applications (see Section 5.1) where experts are near-optimal.

5.2.2 Covariate Shift Issue

We first formalize one of the earliest IL algorithms, behavioral cloning (BC), which is a supervised learning method to solve the IL problem. The BC algorithm finds a policy π_{bc} by solving the following optimization problem:

$$\operatorname*{argmin}_{\pi} L_{\mathrm{bc}}(\pi) = \mathbb{E}_{s\sim d^{\pi_e}}[l(\pi_e(\cdot \mid s), \pi(\cdot \mid s))], \tag{5.1}$$

where the surrogate loss function $l(\pi_e(\cdot|s), \pi(\cdot|s))$ measures how far the learner policy π is with respect to the expert action for the states visited by the expert. Examples of the loss function l comprise of 0-1 loss (described by $\mathbb{E}_{a\sim\pi(\cdot|s)}\mathbb{1}(a \neq \pi_e(s))$ for deterministic expert policies), total variation loss (described by $D_{\mathrm{TV}}(\pi_e(\cdot|s), \pi(\cdot|s)) = 0.5\left\|\pi_e(\cdot|s) - \pi(\cdot|s)\right\|_1$), KL loss (described by $D_{\mathrm{KL}}(\pi(\cdot|s)\|\pi_e(\cdot|s)) = \sum_a \pi(a|s)\log(\pi(a|s)/\pi_e(a|s))$ with π absolutely continuous to π_e), and many more such quantifiers. We remark that the BC algorithm can be solved approximately using the expert dataset $\mathcal{D}_e$ since its optimization problem Eq. (5.1) only depends on the expert state distribution d^{π_e}. So we note that the BC algorithm does not need to know the model P^o, but when given access to P^o, it does not improve. We discuss this further in the following.

We assume we know the model P^o. Hence we now have access to sampling possibly *large* data from the expert state distribution d^{π_e} to calculate the loss $L_{\mathrm{bc}}(\pi_{\mathrm{bc}})$ up to some small error. BC now has the following sub-optimal result.

Theorem 9 ([114])**.** *Assume* $L_{\mathrm{bc}}(\pi_{\mathrm{bc}}) = \varepsilon_{\mathrm{bc}}$*. Then we have* $V_{\pi_e} - V_{\pi_{\mathrm{bc}}} \leq \varepsilon_{\mathrm{bc}}/(1-\gamma)^2$.

Proof. This follows from [114, Theorem 2.1] since $\varepsilon_{\mathrm{bc}}$ is an upper bound on the 0-1 loss of π_{bc}. □

This bound is tight as mentioned in [114], that is, there exist imitation learning examples with $V_{\pi_{\mathrm{bc}}} = V_{\pi_e}(1-\varepsilon_{\mathrm{bc}}H) + \varepsilon_{\mathrm{bc}}H^2$, where $H = 1/(1-\gamma)$ is an effective horizon [106]. We also provide such an instance in Theorem 12. [117, Theorem 5.1(b)] provides the lower-bound $\Omega(\varepsilon H)$ for the known model P^o setting. The tightness of the BC sub-optimality result suggests that it is

one factor of H away from optimality. This is the consequence of the covariate shift or the data distribution shift issue [116, 117, 106].

Consider the following original imitation learning [115] optimization problem.

$$\pi_{\mathrm{il}} = \operatorname*{argmin}_{\pi} L_{\mathrm{il}}(\pi) = \mathbb{E}_{s\sim d^{\pi}}[l(\pi_e(\cdot \mid s), \pi(\cdot \mid s))]. \tag{5.2}$$

We again assume we know the model P^o. We also assume we can query the expert for its actions in any state. Hence we now have access to sampling possibly *large* data from the state distributions d^{π} of all learner policies π to calculate the loss $L_{\mathrm{il}}(\pi_{\mathrm{il}})$ up to some small error. The imitation learning objective Eq. (5.2) now has the following sub-optimal result for the $D_{\mathrm{KL}}(\pi(\cdot|s)\|\pi_e(\cdot|s))$ surrogate loss function l.

Theorem 10. *Fix $\tau > 0$. Assume $L_{\mathrm{il}}(\pi_{\mathrm{il}}) = \varepsilon_{\mathrm{il}}$. Then we have $V_{\pi_e} - V_{\pi_{\mathrm{il}}} \leq \tau' \varepsilon_{\mathrm{il}} + \varepsilon_{\tau}/(1-\gamma)\log(|\mathcal{A}|)$.*

This indeed mitigates the covariate shift issue. This result is actually an improvement from the sub-optimality bound $\mathcal{O}(\varepsilon_{\mathrm{il}}/(1-\gamma)^2)$ in [115]. We provide a proof of this theorem in Section D.2.1. When we know the model P^o, solving the imitation learning objective Eq. (5.2) suffices for tight sub-optimality guarantee. So, why is this not good enough? We remark that to solve the imitation learning objective Eq. (5.2), we need to generate samples from d^{π} *for all policies* π. Clearly, this demands a very high computational capability. It is computationally expensive for solving large-scale problems. Even knowing the model P^o, it is still computationally intractable to compute d^{π} for MDPs with large state-action spaces. Another major drawback of the imitation learning objective Eq. (5.2) is that we need expert supervision to get its actions at any state. This is an expensive requirement in real-world systems where experts are generally human agents and are rarely available. This is also an unrealistic requirement in simulators when we do not have the underlying true reward signals. The best one can do is to solve the imitation learning problem for some heuristic reward function given as a black box model.

This motivates us to propose a new imitation learning algorithm in the following. In summary, we want to avoid both the expensive computations and expensive expert access of IL optimization

Eq. (5.2). We also want to mitigate the covariate shift issue of the BC algorithm. But we want to capture the positive aspects of IL optimization and BC algorithm. Namely, the covariate shift mitigation by the IL optimization Theorem 10 and the inexpensive computation of the BC algorithm Eq. (5.1). In the following, we build this middle ground between the two by the methodology of distributionally robust optimization.

5.2.3 Distributionally Robust Behavioral Cloning

We propose a principled adversarial approach by the methodology of distributionally robust optimization (DRO) to solve the imitation learning problem. DRO is now a well-established area [109, 141, 132], whose formulation is identical to that in the classical RMDP [33, 34] in DR-RL. The distributionally robust behavioral cloning algorithm solves the following optimization problem getting the policy π_{drbc}:

$$\operatorname*{argmin}_{\pi} \max_{\pi' \,:\, D(d^{\pi'}, d^{\pi_e}) \leq \rho_{\mathrm{c}}} \mathbb{E}_{s \sim d^{\pi'}_{Po}} [l(\pi_e(\cdot|s), \pi(\cdot|s))], \tag{5.3}$$

where $D(d^{\pi'}, d^{\pi_e})$ is a distance measure between two probability distributions (e.g., total variation, chi-square, Kullback-Liebler (f-divergences in general), Wasserstein), ρ_{c} is the *covariate shift radius parameter* which is a problem-dependent constant, and $l(\pi_e(\cdot|s), \pi(\cdot|s))$ is the surrogate loss function that we discussed in the previous section. We note that the DR-BC policy π_{drbc} depends on ρ_{c} but simply choose to make it inherent for notation simplicity.

We define the class of policies parameterized by ρ_{c} as $\Pi_{\rho_{\mathrm{c}}} = \{\pi : D(d^{\pi}, d^{\pi_e}) \leq \rho_{\mathrm{c}}\}$. The DR-BC algorithm Eq. (5.3) finds π_{drbc} for the IL problem by minimizing an observed surrogate loss between its actions and the actions of an expert policy under the adversarial state distribution for policy in class $\Pi_{\rho_{\mathrm{c}}}$ which acts as a worse-case distribution. We define the loss function $L_{\mathrm{drbc}}(\pi, \rho_{\mathrm{c}}) = \max_{\pi' \in \Pi_{\rho_{\mathrm{c}}}} \mathbb{E}_{s \sim d^{\pi'}} [l(\pi_e(\cdot|s), \pi(\cdot|s))]$ for any policy π and ρ_{c}. But we immediately notice that to solve the inner optimization in Eq. (5.3) we need access to all the state distributions around the expert's state distribution. Even knowing the model P^o, this is computationally intractable. Moreover, we would also need the capability of querying an expert for actions for

various states chosen by such state distributions. The IL optimization Eq. (5.2) faces similar challenges that we discussed in Section 5.2.2. We now discuss how we circumvent this challenge using the DRO methodology [129, 109].

We restrict to the total-variation distance D_{TV} for the measure D in this paper and leave other measures for future work. Towards this, we specialize the class of policies parameterized by ρ_{c} as $\Pi_{\rho_{\mathrm{c}}} = \{\pi \in \Pi : D_{\mathrm{TV}}(d^{\pi}, d^{\pi_e}) \leq \rho_{\mathrm{c}}\}$. For D_{TV}, we can naturally restrict $\rho_{\mathrm{c}} \in (0, 1]$. We now have the following result that provides a dual reformulation for the inner maximization in Eq. (5.3) as a consequence of the DRO methodology.

Proposition 7 (Informal)**.** *Fix the underlying model P^o. For all $\pi \in \Pi$ and $\rho_{\mathrm{c}} > 0$,*

$$\begin{aligned}&\max_{\pi' \in \Pi_{\rho_{\mathrm{c}}}} \mathbb{E}_{s \sim d^{\pi'}_{P^o}}[l(\pi_e(\cdot|s), \pi(\cdot|s))] \\ &\qquad = \min_{\eta \in \mathbb{R}} \mathbb{E}_{s \sim d^{\pi_e}_{P^o}}[\tilde{f}(l(\pi_e(\cdot|s), \pi(\cdot|s)), \rho_{\mathrm{c}}, \eta)],\end{aligned}$$

where the structure of the function $\tilde{f}$ depends on D_{TV}.

We give formal statement and proof in Section D.2. We give our DR-BC algorithm that only requires an expert dataset $\mathcal{D}_e$ generated according to model P^o in Algorithm 4 based on Proposition 7. The DRO technique in Proposition 7 transforms the inner maximization in Eq. (5.3) to an unconstrained scalar variables convex optimization problem. We remark that this new optimization problem due to the dual reformulation only depends on the expert's state distribution. This enables us to use the expert dataset to solve the DR-BC objective Eq. (5.3). We emphasize that we need access to all the state distributions to solve the inner optimization in Eq. (5.3) directly which is computationally intractable for large-scale problems. Now we are overcoming this challenge through this dual reformulation result. We note that we provide the practical implementable algorithm and more details for DR-BC in Section D.4.

Algorithm 4 Distributionally Robust Behavioral Cloning

1: **Input:** Expert dataset $\mathcal{D}_e = (s_i, a_i)_{i=1}^N$ according to model P^o, covariate shift radius parameter $\rho_{\rm c}$.

2: **Initialize:** Policy π_θ parameterized by θ.

3: Calculate the empirical loss for $L_{\rm drbc}(\pi_\theta, \rho_{\rm c})$:

$$\min_{\lambda>0, \eta\in\mathbb{R}} (1/N) \sum_{(s,a)\in\mathcal{D}_e} \tilde{f}(l(a, \pi_\theta(s)), \lambda, \rho_{\rm c}, \eta). \tag{5.4}$$

4: $\theta \leftarrow \operatorname{argmin}_\theta L_{\rm drbc}(\pi_\theta, \rho_{\rm c})$.

5: **Output policy:** $\widehat{\pi}_{\rm drbc} = \pi_\theta$

We emphasize that our DR-BC objective Eq. (5.3) addresses the covariate shift issue in a principled manner compared to other adversarial methods mentioned in Section 5.1. Intuitively, the inner maximization in the DR-BC objective Eq. (5.3) implicitly perturbs the expert's state distribution, up to the radius parameter $\rho_{\rm c}$. Then it minimizes the loss between the learner and expert actions on the states that the worse perturbed state distribution chooses. We also provide a theoretical DR-BC algorithm in Algorithm 4 that only requires an expert dataset $\mathcal{D}_e$ by incorporating the DRO methodology. We remark that our DR-BC algorithm is as computationally inexpensive as the BC algorithm in terms of not needing access to the learner's state distribution, which the IL optimization needs. We note that Step 3 in the DR-BC algorithm is a scalar variable convex optimization problem. Assuming we have a computationally efficient method to solve Step 3 in the DR-BC algorithm, we are as computationally efficient as the BC algorithm.

We assume the model P^o is known. We now turn to our *main question* to address the covariate shift issue: *Can we give an order optimal sub-optimality bound for the DR-BC algorithm?* That is, can we achieve $V_{\pi_e} - V_{\pi_{\rm drbc}} \leq \mathcal{O}(\varepsilon H)$? Yes, we do achieve it as discussed below.

Before giving the result, we consider a class of DR-BC policies as follows. Let $L_{\rm drbc}(\pi_{\rm drbc}, \rho_{\rm c}) = \varepsilon_{\rm drbc}(\rho_{\rm c}) > 0$ be a small optimization error for all $\rho_{\rm c} \in (0, 1]$. We then define a feasibility set $\mathcal{U} = \{x \in (0, 1] : \gamma^2 \varepsilon_{\rm drbc}(x) \leq x(1-\gamma)^2\}$ which is a collection of all DR-BC policies that satisfy

the aforementioned constraint. We now give the following sub-optimality result for π_{drbc} Eq. (5.3) with loss function l as the $D_{\mathrm{KL}}(\pi(\cdot|s)\|\pi_e(\cdot|s))$ divergence.

Theorem 11 (DR-BC sub-optimality bound)**.** *For any $\rho_{\mathrm{c}} \in \mathcal{U}$, DR-BC policy π_{drbc} satisfies*

$$V_{\pi_e} - V_{\pi_{\mathrm{drbc}}} \leq \frac{1}{1-\gamma} \cdot \frac{\tau' \varepsilon_{\mathrm{drbc}}(\rho_{\mathrm{c}}) + \varepsilon_\tau}{\log(|\mathcal{A}|)}.$$

Remark 8. *This result $V_{\pi_e} - V_{\pi_{\mathrm{drbc}}} \leq \mathcal{O}(\varepsilon_{\mathrm{drbc}}(\rho_{\mathrm{c}})H)$ is order optimal which mitigates the covariate shift issue. In other words, the DR-BC algorithm recovers the positive aspect of IL optimization discussed in Section 5.2.2. We note that $\varepsilon_{\mathrm{drbc}}(\rho_{\mathrm{c}})$ increases with ρ_{c} and finding the closed-form for $\varepsilon_{\mathrm{drbc}}(\rho_{\mathrm{c}})$ is problem dependent. This suggests characterizing the feasibility set $\mathcal{U}$ is problem dependent as well. That being said, we provide examples where DR-BC algorithm achieves a tight sub-optimality bound in Theorem 12. MILO [124] also achieves optimal $\mathcal{O}(\varepsilon H)$ sub-optimality bound under the known model P^o. Although a direct comparison between these two methods is unfair since MILO is an IRL approach that relies on decoding the true reward function and our DR-BC is an imitation learning algorithm that learns to mimic expert actions directly.*

We now show that there are examples in which the BC algorithm suffers with quadratic horizon dependence whereas DR-BC is able to recover the linear horizon dependence.

Theorem 12 (Covariate Shift mitigation)**.** *Consider a class of MDPs $\{\mathcal{S}, \mathcal{A}, P^o, \gamma, r\}$ and corresponding MDP dependent covariate shift parameter $\rho_{\mathrm{c}} > 0$. There exists some arbitrarily small $\varepsilon > 0$ with $L_{\mathrm{bc}}(\pi_{\mathrm{bc}}) \leq \varepsilon$ and all $L_{\mathrm{drbc}}(\pi_{\mathrm{drbc}}, \rho_{\mathrm{c}}) \leq \varepsilon$. There exists expert policy π_e and initial state $s_0 \in \mathcal{S}$ such that we have $V_{\pi_{\mathrm{bc}}}(s_0) \leq V_{\pi_e}(s_0) - \varepsilon\gamma/(1-\gamma)^2$ and $V_{\pi_e}(s_0) - \varepsilon/(1-\gamma) = V_{\pi_{\mathrm{drbc}}}(s_0)$.*

We also present the approximation result for the sub-optimality of $\widehat{\pi}_{\mathrm{drbc}}$ returned by Algorithm 4 that uses the expert dataset $\mathcal{D}_e$. We consider $\mathrm{e}_{\min}$, the minimum non-zero probability value in π_e, as a problem dependent constant. We again consider $L_{\mathrm{drbc}}(\pi_{\mathrm{drbc}}, \sqrt{\rho_{\mathrm{c}}}) = \varepsilon_{\mathrm{drbc}}(\rho_{\mathrm{c}}) > 0$ be a small optimization error for all ρ_{c}.

Theorem 13 (Approximate DR-BC sub-optimality bound)**.** *Let the data dependent feasibility set be* $\mathcal{U}_N = \{x \in (0,1] : \gamma^2 \widehat{\varepsilon}_{\mathrm{drbc}}(x) \leq x(1-\gamma)^2\}$ *where* $\widehat{\varepsilon}_{\mathrm{drbc}}(\rho_{\mathrm{c}}) = \varepsilon_{\mathrm{drbc}}(\rho_{\mathrm{c}}) + \widetilde{\mathcal{O}}(\rho_{\mathrm{c}}\sqrt{\log(1/\delta)/(\mathrm{e}_{\min}N)})$. *Then, for any* $\rho_{\mathrm{c}} \in \mathcal{U}_N$*, policy* $\widehat{\pi}_{\mathrm{drbc}}$ *satisfies* $V_{\pi_e} - V_{\widehat{\pi}_{\mathrm{drbc}}} \leq \tau' \widehat{\varepsilon}_{\mathrm{drbc}}(\rho_{\mathrm{c}}) + \varepsilon_\tau/(1-\gamma)\log(|\mathcal{A}|)$*, with probability at least* $1-\delta$*.*

Remark 9. *We note that* $\widetilde{\mathcal{O}}(\cdot)$ *is order optimal up to a logarithmic term on* N *and its exact form is available in Section D.2. We know that the feasibility set* $\mathcal{U}_N$ *is problem dependent. This immediately poses the question whether the DR-BC policy is able to mitigate the covariate shift even with considerably small expert dataset sizes. We indeed showcase this with our practical version of the DR-BC algorithm provided in Sections D.4 and 5.4. In particular, Section D.4.2 has details on expert data sizes.*

We end this section with this remark for when the model P^o is not known. Suppose we have access to large offline data from non-expert policies. We use it to estimate the model P^o as $\widehat{P^o}$. We then use it in Proposition 7, also in DR-BC algorithm, by quantifying the state-distribution error between $d^{\pi_e}_{\widehat{P^o}} \neq \hat{d}^{\pi_e}_{P^o}$. Here $\hat{d}^{\pi_e}_{P^o}$ is the empirical state distribution of the expert dataset $\mathcal{D}_e$. We have already illustrated the statistical error between $\hat{d}^{\pi_e}_{P^o} \neq d^{\pi_e}_{P^o}$ in Theorem 13. From Lemma 35, the statistical error between $d^{\pi_e}_{\widehat{P^o}} \neq d^{\pi_e}_{P^o}$ amounts for an approximation error of $\|\widehat{P^o} - P^o\|_1 \leq \mathcal{O}(\varepsilon_{\mathrm{approx}}H)$, which is widely available in the literature [102, 142, 106, 50, 143, 3]. Putting these things together, we get an additional error $\mathcal{O}(\varepsilon_{\mathrm{approx}}H^2)$ for the DR-BC sub-optimality Theorem 13. This also matches the lower bound $\Omega(\varepsilon H^2)$ in [117] for the unknown model P^o setting with interaction. We skip presenting this result due to these existing analyses from the literature and a straightforward extension of techniques already presented in this paper. We instead introduce the robust imitation learning problem in the next section.

5.3 Robust Imitation Learning

In this section, we formally introduce the imitation learning problem addressing parameter mismatches (w.r.t transition dynamics) between the real-world and simulator models. We then propose a robust imitation learning algorithm for this problem that is closely related to DR-BC

algorithm in Algorithm 4. We end this section by discussing why we need to study robust imitation learning.

5.3.1 Problem Formulation

We use the RMDP framework [33, 34] subsuming the MDP discussions in Section 5.2.1. Consider an RMDP tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, r, \mu\}$ where $\gamma \in [0.5, 1)$ and the *uncertainty set* $\mathcal{P}$ is defined as

$$\begin{aligned} &\mathcal{P} = \otimes_{(s,a)\in\mathcal{S}\times\mathcal{A}} \mathcal{P}_{s,a} \quad \text{with} \\ &\mathcal{P}_{s,a} = \{P_{s,a} \in \Delta(\mathcal{S}) \ : \ D_{\mathrm{TV}}(P_{s,a}, P^o_{s,a}) \leq \rho'_{\mathrm{r}}\}, \end{aligned} \tag{5.5}$$

where $P^o = (P^o_{s,a}, (s,a) \in \mathcal{S}\times\mathcal{A})$ is the simulator model and $\rho'_{\mathrm{r}} \in (0, (1-\gamma)/\gamma]$ is the radius of the uncertainty set indicating the level of robustness. We assume the real-world model belongs to this uncertainty set $\mathcal{P}$.

From the RMDP literature [33, 83, 3, 35], we introduce the *robust* value function as $V^{\mathrm{rob}}_\pi(s) = \sum_a \pi(a|s) Q^{\mathrm{rob}}_\pi(s,a)$ and the corresponding *robust* Q-value function as $Q^{\mathrm{rob}}_\pi(s,a) = r(s,a) + \gamma \inf_{P_{s,a}\in\mathcal{P}_{s,a}} P^\top_{s,a} V^{\mathrm{rob}}_\pi$ for policy π. We do not introduce the entropy regularization here. Instead for Section 5.3, similar to [114], we consider some expert policy $\pi_e \in \Pi$ that need not be the non-robust optimal policy as in Assumption 8. **The robust IL problem**: A robust IL policy $\widehat{\pi}$ must satisfy $V^{\mathrm{rob}}_{\widehat{\pi}} \approx V^{\mathrm{rob}}_{\pi_e}$. We have provided real-world applications that motivate this problem formulation in Section 5.1. We make the same data assumption as in Section 5.2, that an expert dataset $\mathcal{D}_e = \{s_i, a_i\}_{i=1}^N$ is sampled i.i.d. from $d^{\pi_e}_{P^o}$.

5.3.2 Robust Against Model Mismatch

We let the *robustness radius parameter* ρ_{r} be $\gamma\rho'_{\mathrm{r}}/(1-\gamma) \in (0, 1]$. We define uncertainty set $\mathcal{M} = \{P \in \mathcal{P} : D_{\mathrm{TV}}(d^{\pi_e}_P, d^{\pi_e}_{P^o}) \leq \rho_{\mathrm{r}}\}$. It is straightforward from its definition and Lemma 35 that $\mathcal{M} = \mathcal{P}$. With slight misuse, we reuse notations $L_{\mathrm{drbc}}(\pi)$ and π_{drbc} from Section 5.2 here. We

define the model mismatch distributionally robust behavioral cloning loss for any policy π as

$$L_{\mathrm{drbc}}(\pi) = \max_{P \in \mathcal{M}} \mathbb{E}_{s \sim d_P^{\pi_e}}[D_{\mathrm{KL}}(\pi(\cdot|s) \| \pi_e(\cdot|s))]. \tag{5.6}$$

Now the model mismatch distributionally robust behavioral cloning policy is $\pi_{\mathrm{drbc}} = \operatorname{argmin}_{\pi \in \Pi} L_{\mathrm{drbc}}(\pi)$. This immediately poses a challenge that we cannot directly solve this optimization problem since we will need samples from all the models in $\mathcal{M}$ (assuming having access to all models in $\mathcal{M}$ is unrealistic). So, motivated from the DR-RL literature [83, 3, 35], we now state a dual reformulation result to overcome this challenge (similar to Proposition 7) here.

Proposition 8 (Informal)**.** *For a fixed expert policy $\pi_e \in \Pi$, we have, for all $\pi \in \Pi$ and $\rho_{\mathrm{r}} \in (0, 1]$,*

$$\begin{aligned}
&\max_{P \in \mathcal{M}} \mathbb{E}_{s \sim d_P^{\pi_e}}[D_{\mathrm{KL}}(\pi(\cdot|s) \| \pi_e(\cdot|s))] \\
&\quad = \min_{\eta \in \mathbb{R}} \mathbb{E}_{s \sim d_{P^o}^{\pi_e}}[\tilde{f}(D_{\mathrm{KL}}(\pi(\cdot|s) \| \pi_e(\cdot|s)), \rho_{\mathrm{r}}, \eta)].
\end{aligned}$$

We provide a formal statement and a proof in Section D.3. We do not assume that the model P^o is known. Now it immediately follows from Proposition 8 that the DR-BC algorithm (Algorithm 4) can be used to address the robust IL problem albeit with a different tuning robustness parameter ρ_{r}. We now give the sub-optimality guarantee of model mismatch DR-BC policy.

Theorem 14 (Model mismatch DR-BC sub-optimality bound)**.** *Assume small optimization error $L_{\mathrm{drbc}}(\pi_{\mathrm{drbc}}) = \varepsilon_{\mathrm{drbc}}(\rho_{\mathrm{r}})$. We have $V_{\pi_e}^{\mathrm{rob}} - V_{\pi_{\mathrm{drbc}}}^{\mathrm{rob}} \leq 2\sqrt{\varepsilon_{\mathrm{drbc}}(\rho_{\mathrm{r}})}/(1-\gamma)^2$.*

Note that approximate results similar to that in Theorem 13 hold for model mismatch DR-BC with similar analysis.

Remark 10. *We have an $\mathcal{O}(\varepsilon_{\mathrm{drbc}}(\rho_{\mathrm{r}})H^2)$ sub-optimality bound. When the robustness parameter $\rho_{\mathrm{r}} = 0$, we recover the non-robust BC algorithm and its quadratic horizon dependence [114]. This sub-optimality bound is in fact tight $\Omega(\varepsilon H^2)$ [117] under the unknown model P^o setting. It is not immediately clear if the DR-BC method can avoid the covariate issue as in Section 5.2 when model*

P^o is known. One reason is that the dual reformulation results in Propositions 7 and 8 hold only when either the model or the policy, respectively, is fixed.

5.3.3 Need for Robust Imitation Learning

In this section, we formally show that the IL policy can be arbitrarily bad (as bad as a random policy) and the DR-BC policy recovers a robust performance under model parameter mismatch. We emphasize that the DR-BC policy Eq. (5.6) despite suffering from covariate issue is still able to mitigate the changes in the model parameters showcasing robust performance.

We consider a simple setting with $\mathcal{P} = \{P^o, P'\}$ where P^o is the simulator model and P' is the perturbed model. We give the following result.

Theorem 15 (Robustness Gap)**.** *There exists an uncertainty set $\mathcal{P} = \{P^o, P'\}$, expert policy π_e for nominal model P^o, discount factor $\gamma \in (\gamma_o, 1]$, and initial state $s_0 \in \mathcal{S}$ such that $V_{\pi_{\mathrm{il}},P'}(s_0) \leq \max_\pi V_{\pi,P'}(s_0) - c/(1-\gamma)$, where c is a positive constant, $V_{\pi,P'}$ is the value of any policy π for model P', and π_{il} is the imitation learning policy. Furthermore, for a class of uncertainty sets $\mathcal{P}$, there exists some arbitrarily small $\varepsilon > 0$ with $L_{\mathrm{drbc}}(\pi_{\mathrm{drbc}}) \leq \varepsilon$ with $\rho_{\mathrm{r}} \in (0, 1]$ and $L_{\mathrm{bc}}(\pi_{\mathrm{bc}}) \leq \varepsilon$, such that $V^{\mathrm{rob}}_{\pi_{\mathrm{bc}}}(s_0) \leq V^{\mathrm{rob}}_{\pi_e}(s_0) - c/(1-\gamma)$ and $V^{\mathrm{rob}}_{\pi_e}(s_0) - \varepsilon/(1-\gamma) \leq V^{\mathrm{rob}}_{\pi_{\mathrm{drbc}}}(s_0)$.*

Remark 11. *The imitation learning policy π_{il} deployed on a perturbed model is bad with a performance gap $\Omega(1/(1-\gamma))$. Since $|r(s,a)| \leq 1$ uniformly by assumption, $\|V_{\pi,P}\| \leq 1/(1-\gamma)$ for any policy π and any model P. Therefore, the difference between the optimal/expert value function and the value function of an arbitrary policy cannot be greater than $\mathcal{O}(1/(1-\gamma))$. Thus the performance of an imitation learning policy π_{il} can be as bad as an arbitrary policy in an order sense. Furthermore, there also exists instances where the DR-BC policy is able to recover the robust value of the expert policy whereas the BC policy is as bad as an arbitrary policy in an order sense with a similar argument mentioned above for robust value functions.*

5.4 Experiments

We aim to answer the following questions: (1) Does the DR-BC algorithm mitigate the covariate shift? (2) When model mismatches are present, is the DR-BC algorithm robust compared to

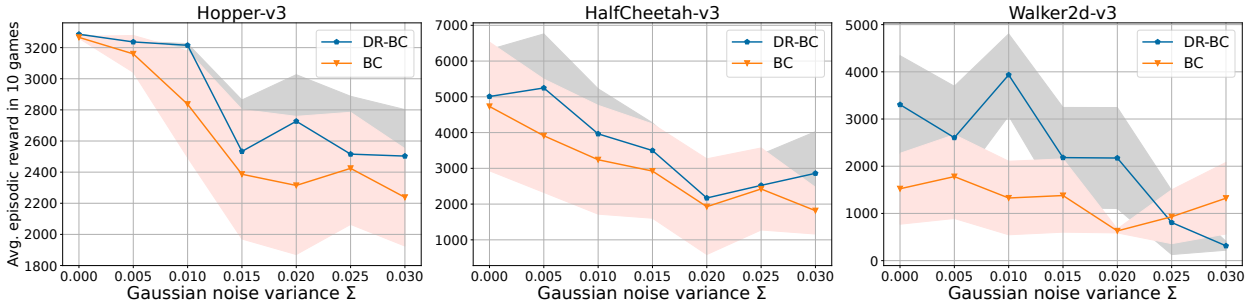

Figure 5.1: *Mitigation of covariate shift.* Average episodic reward on 10 differently seeded episodes. In every decision step, a random Gaussian vector $g \sim \mathcal{N}(0, \Sigma I)$ is added to the action of the BC and DR-BC agents.

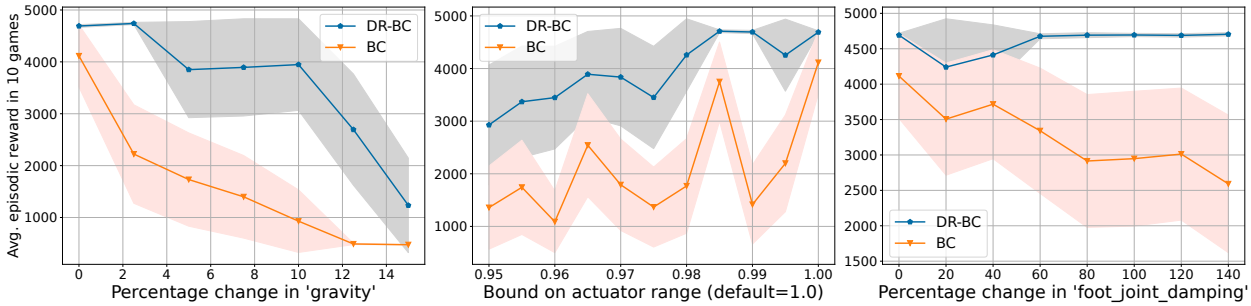

Figure 5.2: `Walker2d-v3` *perturbation results.* Average episodic reward on 10 differently seeded episodes. From left to right, the perturbations are in: *'gravity'*, *'actuator_ctrlrange'* of all joints, and *'foot_joint_damping'* of both foot joints.

the non-robust BC algorithm?

We consider three OpenAI Gym [71] environments simulated with MuJoCo [138]: `Hopper-v3`, `HalfCheetah-v3`, and `Walker2d-v3`. We train both the BC and DR-BC algorithms on the expert data generated by the pre-trained TD3 [110] policies from the RL Baselines3 Zoo repositories [144]. [124] pointed out that BC is very effective at imitating the expert when given large number of samples. Hence, like [116, 124], we give both BC and DR-BC relatively low number of expert trajectories. See Appendix D.4.2 for the selections of the expert datasizes for different environments. Here we point out that since the Gym MuJoCo control tasks have the episode length capped at 1000 steps, we roughly equate 1000 data samples to 1 expert trajectory.

We provide access to our code, more detailed explanation of the experiments, and more simu-

lation results in Section D.4.

5.4.1 Why Is BC the Only Fair Comparison?

As summarized in Section 5.1, most of the Imitation Learning works add additional assumptions in order to improve on the covariate shift issue. If the learner is given the access to a simulator or outright the transition model P^o itself, then one can estimate the state visitation distribution $d^{\pi}_{P^o}$ however well it wants, or one can simply use the model to precisely characterize the state visitation distribution. Once the state visitaion distribution is in hand, the covariate shift issue can be reliably mitigated using imitation learning objective (see Theorem 10). Our goal is to use **only** the expert demonstrations for mitigating the covariate shift issue as well as learning robustness against model perturbations.

5.4.2 Fighting the Covariate Shift

As mentioned in Section 5.2.3, Eq. (5.3) implicitly perturbs the expert's state distribution $d^{\pi_e}_{P^o}$. Theorem 13 suggests that with a high probability, by performing Eq. (5.3), we can capture the learner's state visitation distribution $d^{\pi}_{P^o}$ in the uncertainty set. And we know that the proximity to $d^{\pi}_{P^o}$ gets us closer to the imitation learning objective and mitigates the covaraite shift issue. Now let's move to the empirical side. If we have captured $d^{\pi_e}_{P^o}$ many times using the uncertainty set (based on Theorem 13) during optimization steps in training, then the slightly perturbed learned policy $\pi + g$, where $g \sim \mathcal{N}(0, \Sigma I)$ is a small Gaussian noise, should have its state visitation distribution $d^{\pi+g}_{P^o}$ close to the one of the expert's which is $d^{\pi_e}_{P^o}$. It's well-known that the occupancy measure has a one-to-one correspondence with a policy (see Lemma 32). This suggests that the learner $\pi + g$ should be somewhat "robust" against small action perturbation. In Fig. 5.1, when a small Gaussian noise is added to the action, it shows that the DR-BC maintains its performance better than the BC does, which translates to being close to the expert policy based on our rationale above.

5.4.3 Test For Robustness

We test the robustness of the BC and DR-BC algorithms on perturbed environments. Here we include the simulation results on perturbed `Walker2d-v3`. The simulation results on perturbed `Hopper-v3` and `HalfCheetah-v3` as well as more on `Walker2d` can be found in Appendix D.4. We perturb `Walker2d-v3` by changing the model parameter *'gravity'*, *'actuator_ctrlrange'* of all six joints, and *'joint_damping'* of both left and right foot joints. Fig. 5.2 shows that DR-BC is tenacious under model perturbations. For example, in the middle figure, when the value of *'actuator_ctrlrange'* decreases from the nominal value 1.0, the agent exerts less force on all six joints on the walker. Such perturbation greatly undermines the controllability of the object. A non-robust policy such as BC cannot withstand such mismatch between the training and testing environments. Meanwhile, the DR-BC agent fights on and refuses to drop in performance for much wider range of perturbations.

Thus DR-BC is able to mitigating the covariate shift issue as well as be robust against model perturbations.

5.5 Conclusion

In this paper, we present a novel approach for Imitation Learning problem, Distributionally Robust Behavioral Cloning (DR-BC) algorithm. Our proposed DR-BC algorithm utilizes the distributionally robust optimization (DRO) technique for BC to efficiently solve the covariate shift problem in IL and also to address robustness for the changes in the real-world parameters. We have shown through both theoretical and practical analysis that DR-BC can effectively and computation efficiently combat both the covariate shift issue and model perturbations.

While in this paper we only consider the total variation distance for the inner maximization, future work will explore using other types measures such as KL-divergence and Chi-square divergence. The same applies for the loss function considered in this work which is limited to KL-divergence. Another limitation in this paper is the restriction of the expert policy for mitigating covariate shift issue that we want to address in future. We also plan to work on the scenario

where the model is not known in large-scale problems using general function approximations. An interesting practical direction could be to use DR-BC algorithm to fine-tune the policy network in online IL algorithms like GAIL which generate more diverse and realistic examples.

6. OFFLINE REINFORCEMENT LEARNING USING DISTRIBUTIONALLY ROBUST REINFORCEMENT LEARNING*

In this chapter, we propose a novel Model(transition dynamics) Pessimistic Q-Iteration (MPQI) algorithm for finite state-action space setting and Linear-MDP Model-Pessimistic Q-Iteration (LMM-PQI) algorithm for a linear architecture setting to solve the offline reinforcement learning problem.

6.1 Introduction

Offline reinforcement learning (RL) has gained attention due to its potential to overcome the limitations of online data collection in various real-world applications. While RL has demonstrated impressive performance in gaming scenarios with abundant training data, obtaining real-time data in domains like clinical trials and online advertising is challenging and resource-intensive. To address this issue, leveraging previously collected historical data has emerged as a promising approach. This data-driven approach has been explored in diverse domains, including robotics [145, 146], autonomous driving [147, 148], and healthcare [149, 150]. Offline RL, also known as batch RL, refers to the subfield that focuses on utilizing historical data without additional exploration of the environment.

The primary objective of offline RL is to learn an optimal policy only using offline/historical data collected in the environment of interest. Furthermore, the goal of offline RL algorithms is to achieve the desired statistical accuracy to learn an optimal policy while minimizing the number of required offline data samples. Offline RL problem has been addressed extensively in the literature [89, 63, 90, 91, 92, 93, 94]. Many recent works also develop deep RL algorithms and heuristics for the offline RL problem, focusing on the algorithmic and empirical aspects [95, 96, 97, 98, 99]. Many theoretical work focus on analyzing the variations of reward pessimistic-bonus based algorithm [151, 152], by identifying the necessary and sufficient conditions (data coverage assumption) for the learned policy to be approximately optimal and characterizing the performance in terms of

*Reprinted with permission from Kishan Panaganti, Zaiyan Xu, Dileep Kalathil, Mohammad Ghavamzadeh, "Offline Reinforcement Learning using Distributionally Robust Reinforcement Learning." Preprint, 2023.

sample complexity [151, 152, 153]. In this work, we focus on benchmarking our results in tabular setting [154, 152, 155, 153, 156] and linear architecture (in particular, *linear MDP* [50]) setting [151, 5, 156] for the offine RL problem.

Before stating the contributions of our work, we provide a brief overview of the results in offline RL that are directly related to ours in Table 6.1. We mainly make comparisons with the state-of-the-art pessimistic reward bonus type value iteration algorithms and with the recent work which proposed transition dynamics model pessimism algorithm [156].

Algorithm	Algorithm-type	Data coverage assumption	Suboptimality
Lower Bound [153]	-	single-policy, clipped ℓ_∞	$O\left(\sqrt{\frac{SC_{\pi^*}}{(1-\gamma)^3 N}}\right)$
[152]	Reward -pessimism	single-policy, ℓ_∞	$O\left(\sqrt{\frac{SC^+_{\pi^*}}{(1-\gamma)^5 N}}\right)$
[153]	Reward -pessimism	single-policy, clipped ℓ_∞	$O\left(\sqrt{\frac{SC_{\pi^*}}{(1-\gamma)^3 N}}\right)$
[156]	Oracle-Model -pessimism	single-policy, ℓ_∞	$O\left(\sqrt{\frac{S^2AC^+_{\pi^*}}{(1-\gamma)^4 N}}\right)$
MPQI **(this work)**	VI-Model -pessimism	single-policy, clipped ℓ_∞	$O\left(\sqrt{\frac{SC^+_{\pi^*}}{(1-\gamma)^4 N}}\right)$

Table 6.1: Comparison of provable offline RL algorithms in the tabular setting. Here the algorithm-type column is describing the type of algorithm that is proposed and analyzed for solving offline RL problem. The data coverage assumption is based on the constants $C^+_{\pi^*}$ and C_{π^*} being small and bounded. Single-policy (with some given comparator policy π^*) concentrability ℓ_∞ and clipped ℓ_∞ is $C^+_{\pi^*} = \max_{s,a}(\frac{d^{\pi^*}(s,a)}{\mu(s,a)})$ and $C_{\pi^*} = \max_{s,a}(\frac{\min\{d^{\pi^*}(s,a), 1/|\mathcal{S}|\}}{\mu(s,a)})$ respectively, where $d^{\pi^*}(s,a)$ and $\mu(s,a)$ are both discounted occupancy measures corresponding to comparator policy π^* and data generating policy. Finally, the suboptimality column is the statistical bounds for the offline RL objective Eq. (6.1). We make these more formal in further sections from Section 6.2.

We also extend our results to provide offline RL guarantee for large state space setting and compare with existing results related to ours in Table 6.2. We mainly focus on the special linear

architecture of the problem where we assume the true transition dynamics is a *linear MDP*. We provide more details in further sections regarding this linear architecture. We compare our Linear-MDP Model-Pessimistic Q-Iteration (LMMPQI) algorithm with the state-of-the-art pessimistic reward bonus type value iteration algorithms and the transition dynamics model pessimism algorithm [156] under the linear architecture of *linear MDP* true transition dynamics.

Algorithm	Algorithm-type	Data coverage assumption	Suboptimality
Lower Bound [5, Theorem 3.5]	-	$C^o_{\pi^*,\varphi} < \infty$	$\widetilde{\mathcal{O}}\left(\sqrt{\frac{dC^o_{\pi^*,\varphi}}{(1-\gamma)^3 N}}\right)$
[5]	Reward -pessimism	$C^o_{\pi^*,\varphi} < \infty$	$\widetilde{\mathcal{O}}\left(\sqrt{\frac{dC^o_{\pi^*,\varphi}}{(1-\gamma)^3 N}}\right)$
[156]	Oracle-Model -pessimism	$C_{\pi^*,\varphi} < \infty$, $\inf_{s,a,s'} P^o_{s,a}(s') > 0$	$\widetilde{\mathcal{O}}\left(\sqrt{\frac{d^2 C_{\pi^*,\varphi}}{(1-\gamma)^4 N}}\right)$
LMMPQI **(this work)**	VI-Model -pessimism	$C_{\pi^*,\varphi} < \infty$	$\widetilde{\mathcal{O}}\left(\sqrt{\frac{dC_{\pi^*,\varphi}}{(1-\gamma)^4 N}}\right)$

Table 6.2: Comparison of provable offline RL algorithms in the linear MDP setting. Here the algorithm-type column is describing the type of algorithm that is proposed and analyzed for solving offline RL problem. The data coverage assumption is based on the constants $C^o_{\pi^*,\varphi}$ and $C_{\pi^*,\varphi}$ being small and bounded. Single-policy (with some given comparator policy π^*) concentrability feature coverage is defined as $C_{\pi^*,\varphi} = \max_{x\in\mathbb{R}^d} (x^\top \Sigma_{d^{\pi^*}} x)/(x^\top \Sigma_\mu x)$, where $\Sigma_{d^{\pi^*}}$ and Σ_μ are both feature correlation matrices that depend on discounted occupancy measures corresponding to comparator policy π^* and data generating policy. That is, $\Sigma_{d^{\pi^*}} = \mathbb{E}_{s,a\sim d^{\pi^*}}[\varphi(s,a)\varphi(s,a)^\top]$ and $\Sigma_\mu = \mathbb{E}_{s,a\sim\mu}[\varphi(s,a)\varphi(s,a)^\top]$ with d-dimensional feature vectors $\varphi(s,a) \in \mathbb{R}^d$. Likewise, the concentrability in [5] is $C^o_{\pi^*,\varphi} = \mathbb{E}_{s,a\sim d^{\pi^*}}[\varphi(s,a)^\top \Lambda^{-1} \varphi(s,a)]$ where $\Lambda = \mathbb{E}_{s,a\sim\mu}\varphi(s,a)\varphi(s,a)^\top$. Finally, the suboptimality column is the statistical bounds for the offline RL objective Eq. (6.1). We make these notations more formal from Section 6.2. Here we only note the algebraic relation $C_{\pi^*,\varphi} < C^o_{\pi^*,\varphi}$ which we formally show in Lemma 10.

Connection of offline RL to robust RL: The robust Markov decision process (RMDP) framework was first introduced in [33, 34]. The RMDP problem has been analyzed extensively in the literature [37, 36, 38, 72, 73] providing computationally efficient algorithms, but these works are limited to the planning problem. The learning methodologies to solve the RMDP problems termed

Distributionally Robust RL (DRRL) algorithms with provable guarantees have also been proposed [41, 39, 40, 104] including our line of works presented in this thesis [1, 143, 3, 2, 35]. We emphasize in offline RL we are only given access to offline data and cannot collect more data from the environment for the purposes of exploration that is crucial in RL. Thus to penalize the algorithm for visiting unseen areas in the offline data, we have reward pessimistic-bonus based algorithms [151, 152] proposed. Recently, [156] proposes another algorithm which is penalizing the algorithm by being pessimistic with the transition dynamics model instead of a perturbing reward. They show initial guarantees for an unimplementable algorithm as described in Table 6.1. We note that DRRL also focuses on learning the optimal policy for the worse pessimistic model. This important observation motivates us to solve the offline RL problem using the robust MDP framework.

Contributions of our work: (i) We propose novel implementable algorithm based on DRRL to solve the offline RL problem in the finite state-action space. We give good suboptimal guarantees almost matching the lower bound (see Table 6.1), that is, the suboptimality guarantee is one horizon factor ($\mathcal{O}(1/(1-\gamma))$) away in terms of sample complexity N from the state-of-the-art lower bound. (ii) We also extend and propose novel implementable algorithm to solve the offline RL problem for large a state space setting using a linear architecture. We also give better suboptimal guarantee (see Table 6.2), that is, the suboptimality guarantee is one horizon factor ($\mathcal{O}(1/(1-\gamma))$) away in terms of sample complexity N from the state-of-the-art lower bound. We remark that variance-based analyses techniques in literature [154, 153, 5] can be used to close the gap in terms of horizon and postpone this improvement to our future works.

Notations: For a set $\mathcal{X}$, we denote its cardinality as $|\mathcal{X}|$. The set of probability distribution over $\mathcal{X}$ is denoted as $\Delta(\mathcal{X})$. We define total variation distance between two distributions p and q as $D_{\mathrm{TV}}(p,q) = 0.5\left\|p-q\right\|_1$. For any vector x and a positive semidefinite matrix A, $\|x\|_A = \sqrt{x^\top Ax}$. Let $\mathrm{tr}(\cdot)$ denote the trace operator.

6.2 Problem Formulation and Preliminaries

As in the offline RL literature [151, 152, 153], we consider Markov decision processes (MDPs) to formulate this problem. We consider an MDP tuple $(\mathcal{S}, \mathcal{A}, r, P^o, \gamma, \rho)$ where $\mathcal{S}$ is a finite state

space, $\mathcal{A}$ is a finite set of actions, $r : \mathcal{S} \times \mathcal{A} \to [0, 1]$ is a stochastic reward function, $P^o : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is a probability transition function (model) describing an environment, γ is a discount factor, and ρ is the starting state distribution. A stationary (stochastic) policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ specifies a distribution over actions in each state. Each policy $\pi \in \Pi$ induces a discounted occupancy density over state-action pairs $d^\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$ defined as $d^\pi(s, a) = (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t P_t(s_t = s, a_t = a; \pi)$, where $P_t(s_t = s, a_t = a; \pi)$ denotes the visitation probability of state-action pair (s, a) at time step t, starting at $s_0 \sim \rho(\cdot)$ and following π on the model P^o. For simplicity, we denote $P_t(s_t = s, a_t = a; \pi)$ by $d_t^\pi(s, a)$.

The value function of a policy π is $V^\pi(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s, a_t \sim \pi(\cdot|s_t), s_{t+1} \sim P^o_{s_t,a_t},\ \forall\, t \geq 0\right]$ starting at state $s \in \mathcal{S}$ where $r_t \sim r(s_t, a_t)$. Similarly, we define Q function of a policy

$$Q^\pi(s, a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s, a_0 = a, a_t \sim \pi(\cdot|s_t), s_{t+1} \sim P^o_{s_t,a_t},\ \forall\, t > 0\right].$$

We study offline RL where we assume access only to a historical and fixed dataset of interactions $\mathcal{D} = \{(s_i, a_i, r_i, s_i')\}_{i=1}^N$, where $r_i \sim r(s_i, a_i)$, $s_i' \sim P^o_{s_i,a_i}$. We assume that (s_i, a_i) pairs are generated i.i.d. according to a data distribution $\mu \in \Delta(\mathcal{S} \times \mathcal{A})$. Furthermore, without loss of generality, we assume known rewards, that is $r_i = r(s_i, a_i)$. The *goal of offline RL* is to learn a *good* policy $\widehat{\pi}$ based on the offline data $\mathcal{D}$. More formally, the goal is to characterize the statistical gap

$$\mathbb{E}_{s_0 \sim d_0}[V^{\pi^*}(s_0) - \mathbb{E}_{\mathcal{D}}[V^{\widehat{\pi}}(s_0)]], \tag{6.1}$$

where $d_0 = \rho$ is the starting state distribution and the randomness for $\mathbb{E}_{\mathcal{D}}$ is from offline data $\mathcal{D}$.

6.3 Model-Pessimistic Q-Iteration (MPQI)

In this section, we first propose our MPQI algorithm to solve offline RL problem in the tabular setting (finite state-action space). We also provide its theoretical guarantees and give complete proof.

6.3.1 MPQI Algorithm

In this section, we first mention the required details that help us to propose our MPQI algorithm in Algorithm 5. We denote $N(s,a), N(s,a,s')$ as the number of samples for (s,a)-pair and (s,a,s')-tuple in the offline data $\mathcal{D}$. Now, the maximum likelihood estimate for the model $P^o_{s,a}(s')$ for any s,a,s' is given by

$$\widehat{P}^o_{s,a}(s') = \frac{N(s,a,s')}{N(s,a)} \cdot \mathbb{1}\{N(s,a) \geq 1\} + \frac{1}{|\mathcal{S}|} \cdot \mathbb{1}\{N(s,a) = 0\}.$$

We now construct $\widehat{\mathcal{P}}$ with small data-dependent radius as follows:

$$\widehat{\mathcal{P}} = \bigotimes_{s,a} \widehat{\mathcal{P}}_{s,a} \quad \text{where}$$

$$\widehat{\mathcal{P}}_{s,a} = \{P \in \Delta(\mathcal{S}) : D_{\mathrm{TV}}(P, \widehat{P}^o_{s,a}) \leq \min\{1, \sqrt{\frac{|\mathcal{S}| \log(2|\mathcal{S} \times \mathcal{A}|/\delta)}{8N(s,a)}} 1(N(s,a) \geq 1)\}\}.$$

This set $\widehat{\mathcal{P}}$ referred to as uncertainty set in robust RL literature satisfies SA-rectangularity [33, 1, 143, 2, 3, 35]. *This is an important construction which enables the usage of robust Bellman equation [33].* We would like to point out that the uncertainty set construction in [156] does not satisfy SA-rectangularity. Letting Q be any Q-function, we define the robust Bellman operator $\widehat{T}$ for the uncertainty set $\widehat{\mathcal{P}}$ as follows for every s,a-pair:

$$(\widehat{T}Q)(s,a) = r(s,a) + \gamma \inf_{P_{s,a} \in \widehat{\mathcal{P}}_{s,a}} \mathbb{E}_{s' \sim P_{s,a}}[\max_b Q(s',b)].$$

We remark on the efficient computation of the above robust Bellman equation here. We use our practical algorithms from the Chapters 3 and 4 to get an implementable algorithm for the Algorithm 5.

Algorithm 5 Model-Pessimistic Q-Iteration (MPQI) Algorithm

1: **Input:** Offline data $\mathcal{D} = (s_i, a_i, r_i, s_i')_{i=1}^N$

2: **Initialize:** $Q_0 \equiv 0$

3: **for** $k = 0, \cdots, K-1$ **do**

4: **Pessimistic Q-update:** Compute the next iterate $Q_{k+1} = \widehat{T} Q_k$

5: **end for**

6: **Output:** $\pi_K = \operatorname{argmax}_a Q_K(s, a)$

6.3.2 Results and Proofs

Here is a useful concentration inequality for our proofs.

Lemma 6 (Hoeffding's inequality [157, see Theorem 2.8])**.** *Let $X_1, \ldots, X_n$ be independent random variables such that X_i takes its values in $[a_i, b_i]$ almost surely for all $i \leq n$. Let*

$$S = \sum_{i=1}^{n} (X_i - \mathbb{E}[X_i]).$$

Then for every $t > 0$,

$$\mathbb{P}(S \geq t) \leq \exp\left(-\frac{2t^2}{\sum_{i=1}^n (b_i - a_i)^2}\right).$$

Furthermore, if $X_1, \ldots, X_n$ are a sequence of independent, identically distributed random variables with mean μ. Let $\overline{X}_n = \frac{1}{n}\sum_{i=1}^n X_i$. Suppose that $X_i \in [a, b]$, $\forall i$. Then for all $t > 0$

$$\mathbb{P}\left(\left|\overline{X}_n - \mu\right| \geq t\right) \leq 2\exp\left(-\frac{2nt^2}{(b-a)^2}\right).$$

We first make the observation that the true model P^o lies in the uncertainty set $\widehat{\mathcal{P}}$ with high probability. Intuitively, the empirical estimator $\widehat{P}^o$ of P^o are statistically closer which is dependent on the number of samples. We make this observation and intuition formal in the lemma below.

Lemma 7. *We have $P^o \in \widehat{\mathcal{P}}$ with probability at least $1 - \delta$.*

Proof. We start with the fact that $D_{\mathrm{TV}}(p,q) \leq 1$ for any distributions p, q. For the case $N(s,a) < 1$, i.e., $N(s,a) = 0$, it is trivial that $P^o_{s,a} \in \widehat{\mathcal{P}}_{s,a}$, almost surely, since $\widehat{\mathcal{P}}_{s,a} = \Delta(\mathcal{S})$.

From Hoeffding's inequality (Lemma 6), we have $\left\| P^o_{s,a} - \widehat{P}_{s,a} \right\|_1 \leq \sqrt{|\mathcal{S}| \log(2/\delta)/(2N(s,a))}$ for any s, a pair with probability at least $1 - \delta/(|\mathcal{S} \times \mathcal{A}|)$. Thus $\otimes_{s,a} P^o_{s,a} \in \otimes_{s,a} \widehat{\mathcal{P}}_{s,a}$ holds with probability at least $1 - \delta$. □

We are now ready to present our main result of this chapter. With the above result, we now provide the offline RL suboptimality guarantee below.

Theorem 16. *Let π_K be the MPQI policy after K iterations. With probability at least $1-\delta$ it holds that*

$$\mathbb{E}_{s_0 \sim d_0}[V^{\pi^*}(s_0) - \mathbb{E}_{\mathcal{D}}[V^{\pi_K}(s_0)]] \leq \frac{4\gamma\sqrt{C^+_{\pi^*}}}{(1-\gamma)^2}\sqrt{\frac{2|\mathcal{S}|\log(2|\mathcal{S}\times\mathcal{A}|/\delta)}{N}} + \frac{2\gamma^{K+1}}{(1-\gamma)^2}.$$

Proof. We first make important definitions that will be useful for our analyses. We denote the value function of policy π for the transition dynamics model P as V^π_P. We now denote the robust value function [2, 35, 3] for uncertainty set $\widehat{\mathcal{P}}$ as $V^\pi_{\widehat{\mathcal{P}}} = \min_{P \in \widehat{\mathcal{P}}} V^\pi_P$ and its optimal robust policy as $\widehat{\pi}^* = \operatorname{argmax}_\pi V^\pi_{\widehat{\mathcal{P}}}$. We let $Q^\pi_{\widehat{\mathcal{P}}}$ be its corresponding robust Q-function. From robust RL [2, 35, 3] we can write the following robust Bellman equation: $Q^\pi_{\widehat{\mathcal{P}}}(s,a) = r(s,a) + \gamma \min_{P_{s,a} \in \widehat{\mathcal{P}}_{s,a}} \mathbb{E}_{s' \sim P_{s,a}}(V^\pi_{\widehat{\mathcal{P}}}(s'))$. To make it notationally easy, we write $V^{\pi^*}(d^\pi)$ as $V^{\pi^*}_{P^o}(d^\pi_{P^o})$ making the dependence on the model P^o explicit.

We now start analyzing offline RL suboptimality as:

$$\begin{aligned}
\mathbb{E}_{s_0 \sim d_0}[V^{\pi^*}_{P^o}(s_0) - V^{\pi_K}_{P^o}(s_0)] &= \mathbb{E}_{s_0 \sim d_0}[V^{\pi^*}_{P^o}(s_0) - V^{\pi_K}_{\widehat{\mathcal{P}}}(s_0) + V^{\pi_K}_{\widehat{\mathcal{P}}}(s_0) - V^{\pi_K}_{P^o}(s_0)] \\
&\overset{(a)}{\leq} \mathbb{E}_{s_0 \sim d_0}[V^{\pi^*}_{P^o}(s_0) - V^{\pi_K}_{\widehat{\mathcal{P}}}(s_0)] \\
&= \mathbb{E}_{s_0 \sim d_0}[V^{\pi^*}_{P^o}(s_0) - V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s_0) + V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s_0) - V^{\pi_K}_{\widehat{\mathcal{P}}}(s_0)] \\
&\leq \mathbb{E}_{s_0 \sim d_0}[V^{\pi^*}_{P^o}(s_0) - V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s_0)] + \left\| V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}} - V^{\pi_K}_{\widehat{\mathcal{P}}} \right\|_\infty \\
&\overset{(b)}{\leq} \mathbb{E}_{s_0 \sim d_0}[V^{\pi^*}_{P^o}(s_0) - V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s_0)] + \frac{2\gamma^{K+1}}{(1-\gamma)^2}, \qquad (6.2)
\end{aligned}$$

where (a) follows from Lemma 7 and definition of robust value function $V^{\pi_K}_{\widehat{\mathcal{P}}}(s_0)$ and (b) follows from robust amplification lemma [2, Lemma 10, eq.(28)]. For the rest of the analysis, we focus on analyzing $\mathbb{E}_{s_0\sim d_0}[V^{\pi^*}_{P^o}(s_0) - V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s_0)]$.

Observe that,

$$
\begin{aligned}
&\mathbb{E}_{s_0\sim d_0}[V^{\pi^*}_{P^o}(s_0) - V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s_0)] = \mathbb{E}_{s_0\sim d_0}[Q^{\pi^*}_{P^o}(s_0,\pi^*(s_0)) - Q^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s_0,\widehat{\pi}^*(s_0))] \\
&\overset{(c)}{\leq} \mathbb{E}_{s_0\sim d_0}[Q^{\pi^*}_{P^o}(s_0,\pi^*(s_0)) - Q^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s_0,\pi^*(s_0))] \\
&\overset{(d)}{=} \mathbb{E}_{s_0\sim d_0}[r(s_0,\pi^*(s_0)) + \gamma\mathbb{E}_{s'\sim P^o_{s_0,\pi^*(s_0)}}(V^{\pi^*}_{P^o}(s')) \\
&\qquad\qquad - r(s_0,\pi^*(s_0)) - \gamma \min_{P_{s_0,\pi^*(s_0)}\in\widehat{\mathcal{P}}_{s_0,\pi^*(s_0)}} \mathbb{E}_{s'\sim P_{s_0,\pi^*(s_0)}}(V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s'))] \\
&= \mathbb{E}_{s_0\sim d_0}[\gamma\mathbb{E}_{s'\sim P^o_{s_0,\pi^*(s_0)}}(V^{\pi^*}_{P^o}(s')) - \gamma\mathbb{E}_{s'\sim P^o_{s_0,\pi^*(s_0)}}(V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s'))] + \\
&\qquad\qquad \mathbb{E}_{s_0\sim d_0}[\gamma\mathbb{E}_{s'\sim P^o_{s_0,\pi^*(s_0)}}(V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s'))] - \gamma \min_{P_{s_0,\pi^*(s_0)}\in\widehat{\mathcal{P}}_{s_0,\pi^*(s_0)}} \mathbb{E}_{s'\sim P_{s_0,\pi^*(s_0)}}(V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s'))] \\
&= \mathbb{E}_{s_0\sim d_0}[\gamma\mathbb{E}_{s'\sim P^o_{s_0,\pi^*(s_0)}}(V^{\pi^*}_{P^o}(s') - V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s'))] + \\
&\qquad\qquad \underbrace{\mathbb{E}_{s_0\sim d_0}[\gamma\mathbb{E}_{s'\sim P^o_{s_0,\pi^*(s_0)}}(V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s'))] - \gamma \min_{P_{s_0,\pi^*(s_0)}\in\widehat{\mathcal{P}}_{s_0,\pi^*(s_0)}} \mathbb{E}_{s'\sim P_{s_0,\pi^*(s_0)}}(V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s'))]}_{(I)}, \qquad (6.3)
\end{aligned}
$$

where (c) follows since $\widehat{\pi}^*$ is optimal robust policy of $V^{\pi}_{\widehat{\mathcal{P}}}$ and (d) follows from classical and robust Bellman equations.

Analyzing (I) in Eq. (6.3) for any $P_{s_0,\pi^*(s_0)} \in \widehat{\mathcal{P}}_{s_0,\pi^*(s_0)}$:

$$
\begin{aligned}
(I) &= \mathbb{E}_{s_0\sim d_0}[\gamma\mathbb{E}_{s'\sim P^o_{s_0,\pi^*(s_0)}}(V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s')) - \gamma\mathbb{E}_{s'\sim \widehat{P}^o_{s_0,\pi^*(s_0)}}(V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s')) \\
&\qquad\qquad \gamma\mathbb{E}_{s'\sim \widehat{P}^o_{s_0,\pi^*(s_0)}}(V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s')) - \gamma\mathbb{E}_{s'\sim P_{s_0,\pi^*(s_0)}}(V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s'))] \\
&\overset{(g)}{\leq} \frac{\gamma}{1-\gamma}\min\{1, \sqrt{\frac{\log(2|\mathcal{S}\times\mathcal{A}|/\delta)}{8N(s_0,\pi^*(s_0))}}1(N(s_0,\pi^*(s_0))\geq 1)\} + \\
&\qquad\qquad \gamma\mathbb{E}_{s_0\sim d_0}[\mathbb{E}_{s'\sim \widehat{P}^o_{s_0,\pi^*(s_0)}}(V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s')) - \gamma\mathbb{E}_{s'\sim P_{s_0,\pi^*(s_0)}}(V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s'))] \\
&\overset{(h)}{\leq} \frac{2\gamma}{1-\gamma}\sqrt{\frac{\log(2|\mathcal{S}\times\mathcal{A}|/\delta)}{8N(s_0,\pi^*(s_0))}}1(N(s_0,\pi^*(s_0))\geq 1), \qquad (6.4)
\end{aligned}
$$

where (g), holds with probability at least $1-\delta$, follows from Holder's inequality and by Lemma 7, and (h) by Holder's inequality and the definition of uncertainty set $\widehat{\mathcal{P}}$.

Substituting Eq. (6.4) back in Eq. (6.3) and via recursion we get,

$$\begin{aligned}\mathbb{E}_{s_0\sim d_0}[V^{\pi^*}_{P^o}(s_0) - V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s_0)] &\leq \frac{2\gamma}{1-\gamma}\sum_{t=0}^{\infty}\gamma^t\mathbb{E}_{s\sim d^{\pi^*}_{P^o,t}}[\sqrt{\frac{\log(2|\mathcal{S}\times\mathcal{A}|/\delta)}{8N(s,\pi^*(s))}}1(N(s,\pi^*(s))\geq 1)] \\ &= \frac{2\gamma}{(1-\gamma)^2}\mathbb{E}_{s\sim d^{\pi^*}_{P^o}}[\sqrt{\frac{\log(2|\mathcal{S}\times\mathcal{A}|/\delta)}{8N(s,\pi^*(s))}}1(N(s,\pi^*(s))\geq 1)],\end{aligned}$$

where last equality follows by the definition of state-distribution $d^{\pi^*}_{P^o} = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t d^{\pi^*}_{P^o,t}$. Now, putting this back in Eq. (6.2), the offline RL guarantee becomes:

$$\begin{aligned}&\mathbb{E}_{\mathcal{D}}[\mathbb{E}_{s_0\sim d_0}[V^{\pi^*}_{P^o}(s_0) - V^{\pi_K}_{P^o}(s_0)]] \leq \frac{2\gamma^{K+1}}{(1-\gamma)^2} + \\ &\frac{2\gamma}{(1-\gamma)^2}\mathbb{E}_{s\sim d^{\pi^*}_{P^o}}[\mathbb{E}_{\mathcal{D}}[\sqrt{\frac{|\mathcal{S}|\log(2|\mathcal{S}\times\mathcal{A}|/\delta)}{8N(s,\pi^*(s))}}1(N(s,\pi^*(s))\geq 1)]] \\ &\overset{(i)}{\leq} \frac{2\gamma^{K+1}}{(1-\gamma)^2} + \frac{2\gamma}{(1-\gamma)^2}\mathbb{E}_{s\sim d^{\pi^*}_{P^o}}[\sqrt{\frac{16|\mathcal{S}|\log(2|\mathcal{S}\times\mathcal{A}|/\delta)}{8N\mu(s,\pi^*(s))}}] \\ &\overset{(j)}{\leq} \frac{2\gamma^{K+1}}{(1-\gamma)^2} + \frac{2\gamma}{(1-\gamma)^2}\mathbb{E}_{s\sim d^{\pi^*}_{P^o}}[\sqrt{\frac{C^+_{\pi^*}16|\mathcal{S}|\log(2|\mathcal{S}\times\mathcal{A}|/\delta)}{8Nd^{\pi^*}_{P^o}(s,\pi^*(s))}}] \\ &\overset{(k)}{\leq} \frac{2\gamma^{K+1}}{(1-\gamma)^2} + \frac{4\gamma\sqrt{C^+_{\pi^*}}}{(1-\gamma)^2}\sqrt{\frac{2|\mathcal{S}|\log(2|\mathcal{S}\times\mathcal{A}|/\delta)}{N}},\end{aligned}$$

where (i) follows from [152, Lemma 14], (j) by recalling the definition of single-policy clipped concentrability with comparator policy π^*, that is,

$$C^+_{\pi^*} = \max_{s,a}\frac{d^{\pi^*}_{P^o}(s,a)}{\mu(s,a)},$$

and (k) by Cauchy-Schwarz inequality and recognizing $d^{\pi^*}_{P^o}(\cdot,\pi^*(\cdot))$ as a probability distribution. This completes the proof of this main theorem. □

6.4 Linear-MDP Model-Pessimistic Q-Iteration (LMMPQI)

In this section, we first propose our LMMPQI algorithm to solve offline RL problem in the linear representation setting with large state space and finite actions. We also provide its theoretical guarantees and give complete proof.

6.4.1 LMMPQI Algorithm

In this section, we first mention the required details that help us to propose our LMMPQI algorithm in Algorithm 6.

We now define the linear architecture called *linear MDP* used in RL literature [50, 151, 5] for handling large state space setting.

Definition 1 (Linear MDP [50])**.** *We say an MDP $M = (\mathcal{S}, \mathcal{A}, r, P, \gamma)$ is a linear MDP with a known feature map $\varphi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$, if there exists d unknown (signed) measures $\rho = (\rho_1(\cdot), \ldots, \rho_d(\cdot))$ over $\mathcal{S}$ and an unknown vector $\theta \in \mathbb{R}^d$, such that for any $(s, a) \in \mathcal{S} \times \mathcal{A}$, we have*

$$P_{s,a} = \langle \varphi(s,a), \rho(\cdot) \rangle, \;\; r(s,a) = \langle \varphi(s,a), \theta \rangle. \tag{6.5}$$

We now state the linear architecture assumption which we use to propose the algorithm and give the offline RL suboptimality guarantee Eq. (6.1).

Assumption 9 (Linear MDP assumption)**.** *1. $M = (\mathcal{S}, \mathcal{A}, r, P^o, \gamma)$ is linear MDP with ρ^o, φ. Wolog we assume r is known.*

2. $\|\varphi(s,a)\| \leq 1$, for all $(s,a) \in \mathcal{S} \times \mathcal{A}$ and $\varphi_{\min} = \min_{s,a:\varphi(s,a)>0} \varphi(s,a)$. Wolog we assume $\Lambda = \mathbb{E}_{s,a\sim\mu} \varphi(s,a)\varphi(s,a)^\top$ is positive semi-definite (psd) matrix.

Let us define the following useful matrix notations. With some abuse of notation, let P^o be an $|\mathcal{S}||\mathcal{A}| \times |\mathcal{S}|$ matrix for the transition dynamics $P^o_{s,a}(\cdot)$, Φ be an $|\mathcal{S}||\mathcal{A}| \times d$ matrix for the feature vector $\varphi(s,a) \in \mathbb{R}^d$, and ρ^o be a $d \times |\mathcal{S}|$ matrix for the functionals $\rho^o(\cdot)$. We consider the following

multi-variate linear regression problem to estimate the model P^o.

$$\widehat{P}^o = \arg\min_{\Phi\rho} \sum_{i=1}^{N} (\varphi(s_i, a_i)^\top \rho(s_i') - \varphi(s_i, a_i)^\top \rho^o(s_i'))^2.$$

Furthermore, we note by the structure of the offline dataset $\mathcal{D}$, we know $\delta(s_i')$ is an unbiased estimate of $P^o_{s_i,a_i}(s_i') = \varphi(s_i, a_i)^\top \rho^o(s_i')$ which can be used as the target in the above linear regression problem. Following [106, Section 8.3], there is a closed-form solution to the above linear regression problem as follows. That is, the linear MDP model-based estimate $\widehat{P}^o$ can be written as:

$$\widehat{P}^o_{s,a}(s') = \varphi(s, a)^\top \widehat{\rho}(s') \quad \text{where}$$
$$\widehat{\rho}(s') = \frac{1}{N} \sum_{t=1}^{N} \Lambda_N^{-1} \varphi(s_t, a_t) 1(s' = s_t'), \quad \Lambda_N = \frac{\lambda}{N} I + \frac{1}{N} \sum_{t=1}^{N} \varphi(s_t, a_t) \varphi(s_t, a_t)^\top.$$

Fix $\delta \in (0, 1)$. We now define $\widehat{\mathcal{M}}$ with small data-dependent radius as follows:

$$\widehat{\mathcal{M}} = \bigotimes_{i \in [d]} \widehat{\mathcal{M}}_i \quad \text{where} \quad \widehat{\mathcal{M}}_i = \{\rho_i \in \Delta(\mathcal{S}) : \mathrm{d}_{\mathcal{V}}(\rho_i, \widehat{\rho}_i) \leq \widetilde{\mathcal{O}}\left(\frac{\log(N/\delta)}{\varphi_{\min}(1-\gamma)\sqrt{N}}\right)\}.$$

Here $\mathrm{d}_{\mathcal{V}}$ is integral probability metric (IPM) defined as $\mathrm{d}_{\mathcal{V}}(p, q) = \sup_{V \in \mathcal{V}} |\int_s (p(s) - q(s)) V(s) \mathrm{d}s|$ with $\mathcal{V} = \{V \in \mathbb{R}^{|\mathcal{S}|} : \|V\|_\infty \leq 1/(1-\gamma)\}$, and $\widetilde{\mathcal{O}}$ involves logarithmic dependence on parameters d, $1/(1-\gamma)$, and N which we ignore for brevity. Furthermore, we define $\widehat{\mathcal{P}}$ as follows:

$$\widehat{\mathcal{P}} = \{ \left(\sum_{i \in [d]} \varphi_i(s, a) \rho_i(s') \right)_{sas'} : \rho_i \in \widehat{\mathcal{M}}_i, \ \forall i \in [d]\}.$$

This set $\widehat{\mathcal{P}}$ referred to as uncertainty set in robust RL literature satisfies d-rectangularity [158]. We note that all models in the set $\widehat{\mathcal{P}}$ are linear MDP models. *This is an important construction which enables the usage of robust Bellman equation [33] or specifically [158, Eq.(6)].* We would like to point out that the uncertainty set construction in [156] does not satisfy d-rectangularity. Letting Q be any Q-function, we define the robust Bellman operator $\widehat{L}$ for the uncertainty set $\widehat{\mathcal{P}}$ as follows

for every s, a-pair:

$$(\widehat{L}Q)(s,a) = r(s,a) + \gamma \inf_{P_{s,a} \in \widehat{\mathcal{P}}_{s,a}} \mathbb{E}_{s' \sim P_{s,a}}[\max_b Q(s',b)].$$

By construction of the linear MDP models set $\widehat{\mathcal{P}}$, the robust Bellman operator $\widehat{L}$ further simplifies to the following

$$(\widehat{L}Q)(s,a) = r(s,a) + \gamma \sum_{i \in [d]} \varphi_i(s,a) \min_{\rho_i \in \widehat{\mathcal{M}}_i} \mathbb{E}_{s' \sim \rho_i}(\max_b Q(s',b)). \tag{6.6}$$

We remark on the efficient computation of Eq. (6.6) here. Firstly we note that we use distributionally robust optimization tools directly on the optimization over the sets $\widehat{\mathcal{M}}_i$ for all $i \in [d]$. Now with the known metric relation $\mathrm{d}_{\mathcal{V}}(p,q) = D_{\mathrm{TV}}(p,q)/(1-\gamma)$ [159, Theorem 2], we use our practical algorithms from the Chapters 2 and 4 to get an implementable algorithm for the Algorithm 6.

Algorithm 6 Linear-MDP Model-Pessimistic Q-Iteration (MPQI) Algorithm

1: **Input:** Offline data $\mathcal{D} = (s_i, a_i, r_i, s_i')_{i=1}^N$
2: **Initialize:** $Q_0 \equiv 0$, **Evaluate:** Λ_N with $\lambda = N^{3/4}$.
3: **for** $k = 0, \cdots, K-1$ **do**
4: **Linear-MDP Model-Pessimistic Q-update:**
Compute the next iterate $Q_{k+1} = \widehat{L}Q_k$ Eq. (6.6)
5: **end for**
6: **Output:** $\pi_K = \mathrm{argmax}_a \, Q_K(s,a)$

6.4.2 Results and Proofs

In the following, we always use $c > 0$ for a small universal constant whose exact value might be changing.

We first make similar observation as in Lemma 7 that the true model functionals ρ^o lies in the uncertainty set $\widehat{\mathcal{M}}$ with high probability. We make this formal in the lemma below.

Lemma 8. *We have $\rho^o \in \widehat{\mathcal{M}}$ with probability at least $1-\delta$.*

Proof. We start with the following algebraic inequality,

$$\frac{1}{N}\sum_{t=1}^{N}\sup_{V\in\mathcal{V}}|\int_{\mathcal{S}}(P^o_{s_t,a_t}-\widehat{P}_{s_t,a_t})V(ds')| = \frac{1}{N}\sum_{t=1}^{N}\sum_{i=1}^{d}\varphi_i(s_t,a_t)\cdot \mathrm{d}_{\mathcal{V}}(\rho_i^o,\widehat{\rho}_i)$$
$$\geq \sum_{i=1}^{d}\varphi_{\min}\cdot \mathrm{d}_{\mathcal{V}}(\rho_i^o,\widehat{\rho}_i) \geq \varphi_{\min} d\cdot \max_{i\in[d]}\mathrm{d}_{\mathcal{V}}(\rho_i^o,\widehat{\rho}_i).$$

From [106, Lemma 8.7], we have $|\int_{\mathcal{S}}(P^o_{s_t,a_t}-\widehat{P}_{s_t,a_t})V(ds')| \leq \widetilde{\mathcal{O}}\left(d\log(N/\delta)/((1-\gamma)\sqrt{N})\right)$ with probability at least $1-\delta$ for any t, V uniformly. Thus we have a high probability event that $\rho_i^o \in \widehat{\mathcal{M}}_i$ for any $i\in[d]$ simultaneously with probability at least $1-\delta$. □

Before presenting our main result here is an important high probability result.

Lemma 9. *Let $\lambda = N^{3/4}$. For all s, a simultaneously, with probability at least $1-\delta$ we have $\mathbb{E}_{\mathcal{D}}\|\varphi(s,a)\|_{\Lambda_N^{-1}} \leq \|\varphi(s,a)\|_{\Lambda^{-1}} + \frac{d\sqrt[4]{c\log(d/\delta)}}{\sqrt[8]{N}}$ where $\Lambda_N = \frac{\lambda}{N}I + \frac{1}{N}\sum_{t=1}^{N}\varphi(s_t,a_t)\varphi(s_t,a_t)^\top$ and $\Lambda = \mathbb{E}_{s,a\sim\mu}\varphi(s,a)\varphi(s,a)^\top$.*

Proof. Using the fact $\sqrt{a+b}\leq\sqrt{a}+\sqrt{b}$ and for any realization of $\mathcal{D}$, we only need to show that $\|\varphi(s,a)\|^2_{\Lambda_N^{-1}} - \|\varphi(s,a)\|^2_{\Lambda^{-1}} \leq d^2\frac{\sqrt{c\log(d/\delta)}}{N^{1/4}}$. Furthermore, we have $\|\varphi(s,a)\|^2_{\Lambda_N^{-1}} - \|\varphi(s,a)\|^2_{\Lambda^{-1}} = \varphi(s,a)^\top(\Lambda_N^{-1}-\Lambda^{-1})\varphi(s,a) = \varphi(s,a)^\top\Lambda_N^{-1}(\Lambda-\Lambda_N)\Lambda^{-1}\varphi(s,a)$.

For psd matrices we have $\mathrm{tr}(AB)\leq\mathrm{tr}(A)\mathrm{tr}(B)$. We further have

$$\varphi(s,a)^\top\Lambda_N^{-1}\Lambda^{-1}\varphi(s,a) = \mathrm{tr}(\varphi(s,a)\varphi(s,a)^\top\Lambda_N^{-1}\Lambda^{-1}) \leq d\mathrm{tr}(\Lambda_N^{-1}\Lambda^{-1}) = d\mathrm{tr}(\Lambda^{-1}\Lambda_N^{-1})$$
$$\overset{(a)}{\leq} d^2\sup_{x\in\mathbb{R}^d}\frac{x^\top\Lambda x}{x^\top\Lambda_N x} \leq d^2\sup_{x\in\mathbb{R}^d}\frac{\|x\|_2^2}{(\lambda/N)\|x\|_2^2} = d^2N^{1/4},$$

where (a) follows from [156, Lemma 15] and last equality follows because $\lambda = N^{3/4}$.

From the feature vector properties, we have $\mathbb{E}_{s,a\sim\mu}[\varphi(s,a)\varphi(s,a)^\top] \leq 1_{d\times d}$. So, from Hoeffding's inequality Lemma 6, we get $\Lambda - \Lambda_N \leq \sqrt{\frac{c\log(1/\delta)}{N}} 1_{d\times d}$ with probability at least $1-\delta/d^2$. Now combining the above, we get the required result. □

We are now ready to present our main result of this chapter. With the above result, we now provide the offline RL suboptimality guarantee below.

Theorem 17. *Let Assumption 9 hold. Let π_K be the LMMPQI algorithm (Algorithm 6) policy after K iterations. With probability at least $1-\delta$ it holds that*

$$\mathbb{E}_{s_0\sim d_0}[V^{\pi^*}(s_0) - \mathbb{E}_{\mathcal{D}}[V^{\pi_K}(s_0)]] \leq \frac{2\gamma^{K+1}}{(1-\gamma)^2} + \frac{2\gamma}{(1-\gamma)^2}\widetilde{\mathcal{O}}\left(\sqrt{\frac{\log(dN/\delta)}{N}}\right)\max\{\sqrt{dC_{\pi^*,\varphi}}, \frac{d}{\varphi_{\min}}\} + \frac{d}{(1-\gamma)^2}\widetilde{\mathcal{O}}\left(\frac{(\log(d/\delta))^{3/4}}{N^{5/8}}\right).$$

Proof. We first recall our analyses of Theorem 16. We denote the value function of policy π for the transition dynamics model P as V_P^π. We now denote the robust value function [2, 35, 3] for uncertainty set $\widehat{\mathcal{P}}$ as $V_{\widehat{\mathcal{P}}}^\pi = \min_{P\in\widehat{\mathcal{P}}} V_P^\pi$ and its optimal robust policy as $\widehat{\pi}^* = \operatorname{argmax}_\pi V_{\widehat{\mathcal{P}}}^\pi$. We let $Q_{\widehat{\mathcal{P}}}^\pi$ be its corresponding robust Q-function. From robust RL [2, 35, 3] we can write the following robust Bellman equation: $Q_{\widehat{\mathcal{P}}}^\pi(s,a) = r(s,a) + \gamma \min_{P_{s,a}\in\widehat{\mathcal{P}}_{s,a}} \mathbb{E}_{s'\sim P_{s,a}}(V_{\widehat{\mathcal{P}}}^\pi(s'))$. To make it notationally easy, we write V^{π^*} (d^π) as $V_{P^o}^{\pi^*}$ ($d_{P^o}^\pi$) making the dependence on the model P^o explicit.

We again recall Eq. (6.2):

$$\mathbb{E}_{s_0\sim d_0}[V_{P^o}^{\pi^*}(s_0) - V_{P^o}^{\pi_K}(s_0)] \leq \mathbb{E}_{s_0\sim d_0}[V_{P^o}^{\pi^*}(s_0) - V_{\widehat{\mathcal{P}}}^{\widehat{\pi}^*}(s_0)] + \frac{2\gamma^{K+1}}{(1-\gamma)^2}. \tag{6.7}$$

Further recalling Eq. (6.3) we know,

$$\mathbb{E}_{s_0\sim d_0}[V_{P^o}^{\pi^*}(s_0) - V_{\widehat{\mathcal{P}}}^{\widehat{\pi}^*}(s_0)] \leq \mathbb{E}_{s_0\sim d_0}[\gamma\mathbb{E}_{s'\sim P^o_{s_0,\pi^*(s_0)}}(V_{P^o}^{\pi^*}(s') - V_{\widehat{\mathcal{P}}}^{\widehat{\pi}^*}(s'))] + \underbrace{\mathbb{E}_{s_0\sim d_0}[\gamma\mathbb{E}_{s'\sim P^o_{s_0,\pi^*(s_0)}}(V_{\widehat{\mathcal{P}}}^{\widehat{\pi}^*}(s'))] - \gamma \min_{P_{s_0,\pi^*(s_0)}\in\widehat{\mathcal{P}}_{s_0,\pi^*(s_0)}} \mathbb{E}_{s'\sim P_{s_0,\pi^*(s_0)}}(V_{\widehat{\mathcal{P}}}^{\widehat{\pi}^*}(s'))]}_{(I)}. \tag{6.8}$$

Analyzing (I) in Eq. (6.8) for any $P \in \widehat{\mathcal{P}}$:

$$
\begin{aligned}
(I) &= \mathbb{E}_{s_0\sim d_0}[\gamma \mathbb{E}_{s'\sim P^o_{s_0,\pi^*(s_0)}}(V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s')) - \gamma \mathbb{E}_{s'\sim \widehat{P}^o_{s_0,\pi^*(s_0)}}(V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s')) \\
&\qquad + \gamma \mathbb{E}_{s'\sim \widehat{P}^o_{s_0,\pi^*(s_0)}}(V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s')) - \gamma \mathbb{E}_{s'\sim P_{s_0,\pi^*(s_0)}}(V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s'))] \\
&\overset{(g)}{\leq} \frac{\gamma}{1-\gamma}\|\varphi(s_0,\pi^*(s_0))\|_{\Lambda_N^{-1}}\widetilde{\mathcal{O}}\left(\sqrt{\frac{\log(d/\delta)}{N}}\right) + \\
&\qquad \gamma \mathbb{E}_{s_0\sim d_0}[\mathbb{E}_{s'\sim \widehat{P}^o_{s_0,\pi^*(s_0)}}(\widehat{V}^{\widehat{\pi}^*}(s')) - \mathbb{E}_{s'\sim P_{s_0,\pi^*(s_0)}}(\widehat{V}^{\widehat{\pi}^*}(s'))] \\
&\overset{(h)}{\leq} \frac{\gamma}{1-\gamma}\|\varphi(s_0,\pi^*(s_0))\|_{\Lambda_N^{-1}}\widetilde{\mathcal{O}}\left(\sqrt{\frac{\log(d/\delta)}{N}}\right) + \sum_{i\in[d]}\varphi_i(s_0,\pi^*(s_0))\mathrm{d}_{\mathcal{V}}(\widehat{\rho}_i,\rho_i) \\
&\overset{(h')}{\leq} \frac{\gamma}{1-\gamma}\|\varphi(s_0,\pi^*(s_0))\|_{\Lambda_N^{-1}}\widetilde{\mathcal{O}}\left(\sqrt{\frac{\log(d/\delta)}{N}}\right) + \widetilde{\mathcal{O}}\left(\frac{d\log(N/\delta)}{\varphi_{\min}(1-\gamma)\sqrt{N}}\right), \qquad (6.9)
\end{aligned}
$$

where (g) holds with probability at least $1-\delta$, which follows from [106, Lemma 8.7], and $(h), (h')$ follows by the definition of set $\widehat{\mathcal{P}}$.

Substituting Eq. (6.9) back in Eq. (6.8) and via recursion we get,

$$
\begin{aligned}
&\mathbb{E}_{s_0\sim d_0}[V^{\pi^*}_{P^o}(s_0) - V^{\widehat{\pi}^*}_{\widehat{\mathcal{P}}}(s_0)] \\
&\leq \frac{2\gamma}{1-\gamma}\sum_{t=0}^{\infty}\gamma^t \mathbb{E}_{s\sim d^{\pi^*}_{P^o,t}}[\|\varphi(s,\pi^*(s))\|_{\Lambda_N^{-1}}\widetilde{\mathcal{O}}\left(\sqrt{\frac{\log(d/\delta)}{N}}\right) + \frac{d\widetilde{\mathcal{O}}(\log(N/\delta))}{\varphi_{\min}\sqrt{N}}] \\
&= \frac{2\gamma d\widetilde{\mathcal{O}}(\log(N/\delta))}{\varphi_{\min}(1-\gamma)^2\sqrt{N}} + \frac{2\gamma}{(1-\gamma)^2}\widetilde{\mathcal{O}}\left(\sqrt{\frac{\log(d/\delta)}{N}}\right)\mathbb{E}_{s\sim d^{\pi^*}_{P^o}}[\|\varphi(s,\pi^*(s))\|_{\Lambda_N^{-1}}],
\end{aligned}
$$

where last equality follows by the definition of state-distribution $d^{\pi^*}_{P^o} = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t d^{\pi^*}_{P^o,t}$. Now, putting this back in Eq. (6.7), the offline RL guarantee becomes:

$$
\begin{aligned}
&\mathbb{E}_{\mathcal{D}}[\mathbb{E}_{s_0\sim d_0}[V^{\pi^*}_{P^o}(s_0) - V^{\pi_K}_{P^o}(s_0)]] \leq \frac{2\gamma^{K+1}}{(1-\gamma)^2} + \frac{2\gamma d\widetilde{\mathcal{O}}(\log(N/\delta))}{\varphi_{\min}(1-\gamma)^2\sqrt{N}} + \\
&\frac{2\gamma}{(1-\gamma)^2}\widetilde{\mathcal{O}}\left(\sqrt{\frac{\log(d/\delta)}{N}}\right)\mathbb{E}_{s\sim d^{\pi^*}_{P^o}}[\mathbb{E}_{\mathcal{D}}[\|\varphi(s,\pi^*(s))\|_{\Lambda_N^{-1}}]] \\
&\overset{(i)}{\leq} \frac{2\gamma^{K+1}}{(1-\gamma)^2} + \frac{2\gamma d\widetilde{\mathcal{O}}(\log(N/\delta))}{\varphi_{\min}(1-\gamma)^2\sqrt{N}} +
\end{aligned}
$$

$$
\begin{aligned}
&\frac{2\gamma}{(1-\gamma)^2}\widetilde{\mathcal{O}}\left(\sqrt{\frac{\log(d/\delta)}{N}}\right)(\mathbb{E}_{s\sim d^{\pi^*}_{Po}}[\|\varphi(s,\pi^*(s))\|_{\Lambda^{-1}}]+\frac{d\sqrt[4]{c\log(d/\delta)}}{\sqrt[8]{N}})\\
&\overset{(j)}{\leq}\frac{2\gamma^{K+1}}{(1-\gamma)^2}+\frac{2\gamma d\widetilde{\mathcal{O}}(\log(N/\delta))}{\varphi_{\min}(1-\gamma)^2\sqrt{N}}+\\
&\frac{2\gamma}{(1-\gamma)^2}\widetilde{\mathcal{O}}\left(\sqrt{\frac{\log(d/\delta)}{N}}\right)(\sqrt{dC_{\pi^*,\varphi}}+\frac{d\sqrt[4]{c\log(d/\delta)}}{\sqrt[8]{N}})\\
&\leq\frac{2\gamma^{K+1}}{(1-\gamma)^2}+\frac{2\gamma}{(1-\gamma)^2}\widetilde{\mathcal{O}}\left(\sqrt{\frac{\log(dN/\delta)}{N}}\right)\max\{\sqrt{dC_{\pi^*,\varphi}},\frac{d}{\varphi_{\min}}\}\\
&+\frac{d}{(1-\gamma)^2}\widetilde{\mathcal{O}}\left(\frac{(\log(d/\delta))^{3/4}}{N^{5/8}}\right),
\end{aligned}
$$

where (i) follows from Lemma 9 with probability at least $1-\delta$, (j) follows by similar analysis in [156, Section C.3] and single-policy concentrability with comparator policy π^*:

$$
C_{\pi^*,\varphi}=\max_{x\in\mathbb{R}^d}\frac{x^\top\Sigma_{d^{\pi^*}_{Po}}x}{x^\top\Sigma_\mu x}.
$$

This completes the proof of this main theorem. □

We remark that wolog we scale the feature vector space such that it satisfies $\varphi_{\min}\geq\sqrt{d/C_{\pi^*,\varphi}}$. With this, the offline RL suboptimality guarantee for the linear architecture is $\widetilde{\mathcal{O}}\left(\sqrt{\frac{dC_{\pi^*,\varphi}}{(1-\gamma)^4N}}\right)$ as mentioned in Table 6.2.

We show the relation between feature concentrability $C^o_{\pi^*,\varphi}$ [5] and $C_{\pi^*,\varphi}$ to complete the comparisons provided in Table 6.2.

Lemma 10. $C_{\pi^*,\varphi}\leq C^o_{\pi^*,\varphi}$.

Proof. Let us define matrix $M=\Lambda^{-1/2}\Sigma_{d^{\pi^*}}\Lambda^{-1/2}$. From definition, we immediately see that $C_{\pi^*,\varphi}$ can be rewritten as $\max_{x\in\mathbb{R}^d}(x^\top Mx)/(\|x\|_2^2)$. So, from linear algebra, it is straightforward that $C_{\pi^*,\varphi}$ is the maximum eigenvalue of M. Again from definition, we have $C^o_{\pi^*,\varphi}=\mathbb{E}_{s,a\sim d^{\pi^*}}[\varphi(s,a)^\top\Lambda^{-1}\varphi(s,a)]=\mathrm{tr}(\mathbb{E}_{s,a\sim d^{\pi^*}}[\Lambda^{-1/2}\varphi(s,a)\varphi(s,a)^\top\Lambda^{-1/2}])$
$=\mathrm{tr}(\Lambda^{-1/2}\mathbb{E}_{s,a\sim d^{\pi^*}}[\varphi(s,a)\varphi(s,a)^\top]\Lambda^{-1/2})$. Thus $C^o_{\pi^*,\varphi}$ is the trace of M. Now, since the trace of M is the sum of eigenvalues of M, the lemma quickly follows. □

6.5 Conclusion

In this work, we presented a novel offline RL algorithm called Model-Pessimism Q-Iteration algorithm with provably optimal performance in finite/large state space and finite action space using only offline data. We also provided performance comparisons in terms of provable offline RL suboptimality guarantees with state-of-the-art algorithms.

One limitation of our present work is that there is a gap of one factor of $1/(1-\gamma)$ from the lower bound for our MPQI/LMMPQI's suboptimality guarantee. In future work, we will tighten this bound. More important future direction is to provide offline RL algorithms based on MPQI that can handle large state space and action space under function approximation setting.

7. CONCLUSION

In this final chapter of this dissertation, we conclude with the summary of this thesis and also discuss future research directions in line with what was discussed thus far.

In Chapter 2, we have developed an online model-free reinforcement learning algorithm, called RLSPI, to learn control policies that can handle parameter uncertainties in large state spaces. Unlike previous empirical approaches, our algorithm provides theoretical guarantees for the performance of the learned policy. It is the first model-free reinforcement learning algorithm with function approximation designed specifically for acquiring an optimal robust policy. We have also conducted empirical evaluations on standard benchmark RL problems to assess the effectiveness of our RLSPI algorithm. This was our first adventure and contribution in the robust RL research area.

In Chapter 3, we propose the Robust Empirical Value Iteration (REVI) algorithm, which is a model-based approach for robust reinforcement learning. Our algorithm approximates the robust Bellman updates in robust dynamic programming. We specifically investigate four distinct uncertainty sets - total variation, chi-square, Kullback-Leibler, - and provide sample complexity results for the learned policy compared to the optimal robust policy. To highlight the theoretical aspects of our REVI algorithm, we showcase its performance in the “Gambler’s Problem" and “Frozen Lake“ environments. Overall, our paper presents a novel approach for robust reinforcement learning and provides insights into its theoretical properties and empirical performance. We also improve our theoretical results in our recent work [35].

In Chapter 4, we introduced a new robust reinforcement learning (RL) algorithm known as the Robust Fitted Q-Iteration algorithm. This algorithm offers provably optimal performance for a Reinforcement MDP (RMDP) with a large state space, utilizing only offline data and function approximation. Additionally, we showcased the remarkable performance of our proposed algorithm by conducting experiments on well-known benchmark problems. This work demonstrates the effectiveness of our algorithm in tackling robust RL tasks and contributes to the advancement

of RL algorithms in challenging environments.

In the preliminary dissertation, we introduced the problem of imitation learning and proposing a new algorithm using the principled approach of distributionally robust optimization. In Chapter 5, we introduce a new algorithm called Distributionally Robust Behavioral Cloning (DR-BC) as a novel approach to the Imitation Learning (IL) problem. Our proposed DR-BC algorithm leverages distributionally robust optimization (DRO) techniques to address the covariate shift problem in IL and enhance robustness against changes in real-world parameters. Through rigorous theoretical analysis and practical evaluations, we demonstrate that DR-BC effectively mitigates the covariate shift issue and model perturbations while being computationally efficient. This work presents a valuable contribution to the field of IL by providing an innovative solution that tackles both the covariate shift problem and robustness challenges.

In our final chapter Chapter 6, we introduced a novel offline RL algorithm known as the Model-Pessimistic Q-Iteration algorithm. This algorithm offers provably optimal performance in environments with finite states and action spaces by leveraging solely offline data. We also provided performance comparisons, specifically focusing on provable suboptimality guarantees in offline RL, with state-of-the-art algorithms. We also extended our setting to linear architecture to support for large state spaces leading to our Linear-MDP Model-Pessimistic Q-Iteration algorithm. This research contributes to the field by presenting an innovative algorithm that achieves optimal performance hoping for future extensions to large scale problems under general large function architectures.

Finally, as a potential future direction where the contributions of this thesis will be impactful, we hope to utilize policy gradient methods for robust RL. Due to its simplicity in a model-free environment, scalability to large/continuous state and action spaces, and applicability to any differentiable policy parameterization, the policy gradient method [160, 161, 162], which models and optimizes the policy directly, has been widely used in RL. The development of a policy gradient method for robust RL with demonstrable resilience to model uncertainty and optimality guarantee still remains substantially open in the robust RL literature. Towards this, we hope to motivate this

future direction based on the current research in this dissertation.

As final comments to end this chapter and dissertation, we have worked on various domains of RL like robust RL, offline RL, imitation learning with the distributionally robust optimization or robust MDP techniques working towards making real world impact by RL algorithms and also giving theoretical support to make the ideas provably concrete.

REFERENCES

[1] K. Panaganti and D. Kalathil, "Robust reinforcement learning using least squares policy iteration with provable performance guarantees," in *International Conference on Machine Learning (ICML)*, 2021, pp. 511–520. xi, 2, 23, 24, 26, 44, 62, 82, 84, 182

[2] ——, "Sample complexity of robust reinforcement learning with a generative model," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2022, pp. 9582–9602. [Online]. Available: https://proceedings.mlr.press/v151/panaganti22a.html xi, xiii, 2, 25, 38, 44, 56, 82, 84, 86, 87, 93, 182

[3] K. Panaganti, Z. Xu, D. Kalathil, and M. Ghavamzadeh, "Robust reinforcement learning using offline data," *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. xi, xii, 3, 58, 71, 72, 73, 82, 84, 86, 93, 194

[4] G. Li, Y. Wei, Y. Chi, Y. Gu, and Y. Chen, "Breaking the sample size barrier in model-based reinforcement learning with a generative model," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 12 861–12 872. xiii, 25, 32, 181

[5] M. Yin, Y. Duan, M. Wang, and Y.-X. Wang, "Near-optimal offline reinforcement learning with linear representation: Leveraging variance information with pessimism," *arXiv preprint arXiv:2203.05804*, 2022. xiii, 80, 81, 82, 89, 95

[6] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018. 1, 39, 199, 204

[7] S. Meyn, *Control Systems and Reinforcement Learning*. Cambridge University Press, 2022. 1

[8] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4. 1

[9] M. Ahn, H. Zhu, K. Hartikainen, H. Ponte, A. Gupta, S. Levine, and V. Kumar, "ROBEL: RObotics BEnchmarks for Learning with low-cost robots," in *Conference on Robot Learning (CoRL)*, 2019. 1

[10] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, T. Lillicrap, and D. Silver, "Mastering Atari, Go, chess and shogi by planning with a learned model," *Nature*, vol. 588, no. 7839, pp. 604–609, 2020. 1

[11] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," *arXiv preprint arXiv:1912.01603*, 2019. 1

[12] "IBM Watson Health." [Online]. Available: https://www.ibm.com/watson-health 1

[13] "Berkeley Artificial Intelligence Research." [Online]. Available: https://bair.berkeley.edu/software.html 1

[14] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, 2017. 1

[15] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, 2021. 1

[16] B. Singh, R. Kumar, and V. P. Singh, "Reinforcement learning in robotic applications: a comprehensive survey," *Artificial Intelligence Review*, pp. 1–46, 2021. 1

[17] F. Berkenkamp, M. Turchetta, A. Schoellig, and A. Krause, "Safe model-based reinforcement learning with stability guarantees," *Advances in neural information processing systems*, vol. 30, 2017. 1

[18] B. Lütjens, M. Everett, and J. P. How, "Certified adversarial robustness for deep reinforcement learning," in *Conference on Robot Learning*. PMLR, 2020, pp. 1328–1337. 1

[19] G. Dulac-Arnold, N. Levine, D. J. Mankowitz, J. Li, C. Paduraru, S. Gowal, and T. Hester, "Challenges of real-world reinforcement learning: definitions, benchmarks and analysis," *Machine Learning*, vol. 110, no. 9, pp. 2419–2468, 2021. 1

[20] F. L. Da Silva, P. Hernandez-Leal, B. Kartal, and M. E. Taylor, "Uncertainty-aware action advising for deep reinforcement learning agents," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 04, 2020, pp. 5792–5799. 1

[21] D. Ghosh, J. Rahme, A. Kumar, A. Zhang, R. P. Adams, and S. Levine, "Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability," *Advances in Neural Information Processing Systems*, vol. 34, pp. 25 502–25 515, 2021. 1

[22] O. Lockwood and M. Si, "A review of uncertainty for deep reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 18, no. 1, 2022, pp. 155–162. 1

[23] E. Salvato, G. Fenu, E. Medvet, and F. A. Pellegrino, "Crossing the reality gap: A survey on sim-to-real transferability of robot controllers in reinforcement learning," *IEEE Access*, vol. 9, pp. 153 171–153 187, 2021. 1, 2

[24] D. Ho, K. Rao, Z. Xu, E. Jang, M. Khansari, and Y. Bai, "Retinagan: An object-aware approach to sim-to-real transfer," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 10 920–10 926. 1

[25] "Closing the Reality Gap in Sim2Real transfer for robotics." [Online]. Available: https://sim2real.github.io/ 1, 2

[26] Y. Shukla and J. Sinpov, "A framework for curriculum schema transfer from low-fidelity to high-fidelity environments." 1

[27] M. Navardi, P. Dixit, T. Manjunath, N. R. Waytowich, T. Mohsenin, and T. Oates, "Toward real-world implementation of deep reinforcement learning for vision-based autonomous drone navigation with mission," *UMBC Student Collection*, 2022. 1

[28] "Closing the Simulation-to-Reality gap for Deep Robotic Learning." [Online]. Available: https://ai.googleblog.com/2017/10/closing-simulation-to-reality-gap-for.html 1, 2

[29] "Closing the Sim2Real Gap with NVIDIA Isaac sim and NVIDIA Isaac replicator." [Online]. Available: https://developer.nvidia.com/blog/closing-the-sim2real-gap-with-nvidia-isaac-sim-and-nvidia-isaac-replicator/ 1

[30] N. Sünderhauf, O. Brock, W. Scheirer, R. Hadsell, D. Fox, J. Leitner, B. Upcroft, P. Abbeel, W. Burgard, M. Milford *et al.*, "The limits and potentials of deep learning for robotics," *The International journal of robotics research*, vol. 37, no. 4-5, pp. 405–420, 2018. 2, 43

[31] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 2017, pp. 23–30. 2, 43

[32] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 3803–3810. 2, 43

[33] G. N. Iyengar, "Robust dynamic programming," *Mathematics of Operations Research*, vol. 30, no. 2, pp. 257–280, 2005. 2, 5, 7, 9, 10, 24, 28, 30, 43, 44, 47, 48, 61, 67, 72, 81, 84, 90, 151, 153, 155, 160, 178, 181, 182

[34] A. Nilim and L. El Ghaoui, "Robust control of Markov decision processes with uncertain transition matrices," *Operations Research*, vol. 53, no. 5, pp. 780–798, 2005. 2, 5, 7, 9, 24, 30, 43, 44, 48, 61, 67, 72, 81, 161, 166, 181

[35] Z. Xu*, K. Panaganti*, and D. Kalathil, "Improved sample complexity bounds for distributionally robust reinforcement learning," in *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*. Conference on Artificial Intelligence and Statistics, 2023. 3, 62, 72, 73, 82, 84, 86, 93, 97

[36] W. Wiesemann, D. Kuhn, and B. Rustem, "Robust Markov decision processes," *Mathematics of Operations Research*, vol. 38, no. 1, pp. 153–183, 2013. 5, 7, 24, 44, 62, 81, 182

[37] H. Xu and S. Mannor, "Distributionally robust Markov decision processes," in *Advances in Neural Information Processing Systems*, 2010, pp. 2505–2513. 5, 7, 24, 44, 62, 81, 182

[38] P. Yu and H. Xu, "Distributionally robust counterpart in Markov decision processes," *IEEE Transactions on Automatic Control*, vol. 61, no. 9, pp. 2538–2543, 2015. 5, 24, 44, 62, 81

[39] A. Tamar, S. Mannor, and H. Xu, "Scaling up robust mdps using function approximation," in *International Conference on Machine Learning*, 2014, pp. 181–189. 5, 7, 11, 12, 26, 44, 82, 121, 182

[40] A. Roy, H. Xu, and S. Pokutta, "Reinforcement learning under model mismatch," in *Advances in Neural Information Processing Systems*, 2017, pp. 3043–3052. 5, 7, 9, 24, 26, 44, 82, 123, 141, 144, 182

[41] S. H. Lim, H. Xu, and S. Mannor, "Reinforcement learning in robust Markov decision processes," in *Advances in Neural Information Processing Systems*, 2013, pp. 701–709. 5, 26, 44, 82

[42] M. G. Lagoudakis and R. Parr, "Least-squares policy iteration," *Journal of Machine Learning Research*, vol. 4, pp. 1107–1149, 2003. 6, 17, 20, 141

[43] L. Yang and M. Wang, "Reinforcement learning in feature space: Matrix bandit, kernels, and regret bound," in *International Conference on Machine Learning*. PMLR, 2020, pp. 10 746–10 756. 6

[44] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning." in *ICLR (Poster)*, 2016. 6, 20

[45] L. Pinto, J. Davidson, R. Sukthankar, and A. Gupta, "Robust adversarial reinforcement learning," in *International Conference on Machine Learning*, 2017, pp. 2817–2826. 6, 24, 27, 44, 182

[46] D. J. Mankowitz, N. Levine, R. Jeong, A. Abdolmaleki, J. T. Springenberg, Y. Shi, J. Kay, T. Hester, T. Mann, and M. Riedmiller, "Robust reinforcement learning for continuous con-

trol with model misspecification," in *International Conference on Learning Representations*, 2020. 6, 7, 24, 27, 44, 182

[47] H. Zhang, H. Chen, D. S. Boning, and C.-J. Hsieh, "Robust reinforcement learning on state observations with learned optimal adversary," in *International Conference on Learning Representations*, 2020. 6, 24, 44

[48] E. Derman, D. J. Mankowitz, T. A. Mann, and S. Mannor, "Soft-robust actor-critic policy-gradient," in *AUAI press for Association for Uncertainty in Artificial Intelligence*, 2018, pp. 208–218. 6, 7, 20, 21, 24, 26, 44, 57, 59, 144, 182, 187

[49] 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. 6

[50] C. Jin, Z. Yang, Z. Wang, and M. I. Jordan, "Provably efficient reinforcement learning with linear function approximation," in *Conference on Learning Theory*, 2020, pp. 2137–2143. 6, 71, 80, 89

[51] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *International Conference on Machine Learning*, 2004, p. 1. 6, 61

[52] S. Arora, S. Du, S. Kakade, Y. Luo, and N. Saunshi, "Provable representation learning for imitation learning via bi-level optimization," in *International Conference on Machine Learning*. PMLR, 2020, pp. 367–376. 6, 61

[53] L. Wang, Q. Cai, Z. Yang, and Z. Wang, "On the global optimality of model-agnostic meta-learning," in *International Conference on Machine Learning*, 2020, pp. 9837–9846. 6

[54] W. Kong, R. Somani, Z. Song, S. Kakade, and S. Oh, "Meta-learning for mixed linear regression," in *International Conference on Machine Learning*. PMLR, 2020, pp. 5394–5404. 6

[55] R. Wang, D. Foster, and S. M. Kakade, "What are the statistical limits of offline {rl} with linear function approximation?" in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=30EvkP2aQLD 6, 50, 54

[56] Y. Duan, Z. Jia, and M. Wang, "Minimax-optimal off-policy evaluation with linear function approximation," in *International Conference on Machine Learning*, 2020, pp. 2701–2709. 6

[57] D. L. Kaufman and A. J. Schaefer, "Robust modified policy iteration," *INFORMS Journal on Computing*, vol. 25, no. 3, pp. 396–410, 2013. 7, 182

[58] C. Tessler, Y. Efroni, and S. Mannor, "Action robust reinforcement learning and applications in continuous control," in *International Conference on Machine Learning*, 2019, pp. 6215–6224. 7, 182

[59] S. H. Lim and A. Autef, "Kernel-based reinforcement learning in robust Markov decision processes," in *International Conference on Machine Learning*, 2019, pp. 3973–3981. 7, 26, 182

[60] D. P. Bertsekas and S. Ioffe, "Temporal Differences-Based Policy Iteration and Applications in Neuro-Dynamic Programming," *Lab. for Info. and Decision Systems Report LIDS-P-2349, MIT, Cambridge, MA*, vol. 14, 1996. 8

[61] A. Nedić and D. P. Bertsekas, "Least squares policy evaluation algorithms with linear function approximation," *Discrete Event Dynamic Systems*, vol. 13, no. 1-2, pp. 79–110, 2003. 8, 12, 17, 128, 130, 131, 132

[62] D. P. Bertsekas, *Dynamic programming and optimal control, Vol - 2*. Athena scientific Belmont, MA, 2012. 8, 10, 11, 12, 20, 126

[63] ——, "Approximate policy iteration: A survey and some new methods," *Journal of Control Theory and Applications*, vol. 9, no. 3, pp. 310–335, 2011. 8, 44, 79, 181

[64] J. N. Tsitsiklis and B. Van Roy, "Analysis of temporal-difference learning with function approximation," in *Advances in Neural Information Processing Systems*, 1997, pp. 1075–1081. 10, 121, 124, 129

[65] D. P. Bertsekas and H. Yu, "Projected equation methods for approximate solution of large linear systems," *Journal of Computational and Applied Mathematics*, vol. 227, no. 1, pp. 27–50, 2009. 12, 16

[66] V. S. Borkar, *Stochastic approximation: a dynamical systems viewpoint*. Springer, 2009, vol. 48. 17

[67] R. Munos, "Error bounds for approximate policy iteration," in *ICML*, vol. 3, 2003, pp. 560–567. 19, 50, 133

[68] R. Munos and C. Szepesvári, "Finite-time bounds for fitted value iteration," *Journal of Machine Learning Research*, vol. 9, no. 27, pp. 815–857, 2008. 19, 44, 181

[69] A. Lazaric, M. Ghavamzadeh, and R. Munos, "Finite-sample analysis of least-squares policy iteration," *Journal of Machine Learning Research*, vol. 13, no. Oct, pp. 3041–3074, 2012. 19, 44, 138, 181

[70] C. D'Eramo, D. Tateo, A. Bonarini, M. Restelli, and J. Peters, "Mushroomrl: Simplifying reinforcement learning research," *arXiv preprint arXiv:2001.01102*, 2020. [Online]. Available: https://github.com/MushroomRL/mushroom-rl 20

[71] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016. 20, 21, 39, 57, 63, 75, 142, 146, 183, 188, 190, 209, 212

[72] S. Mannor, O. Mebel, and H. Xu, "Robust mdps with k-rectangular uncertainty," *Mathematics of Operations Research*, vol. 41, no. 4, pp. 1484–1509, 2016. 24, 44, 62, 81

[73] R. H. Russel and M. Petrik, "Beyond confidence regions: Tight bayesian ambiguity sets for robust mdps," *Advances in Neural Information Processing Systems*, 2019. 24, 44, 62, 81

[74] E. Derman, D. Mankowitz, T. Mann, and S. Mannor, "A bayesian approach to robust reinforcement learning," in *Uncertainty in Artificial Intelligence*, 2020, pp. 648–658. 24, 44

[75] S. P. Singh and R. C. Yee, "An upper bound on the loss from approximate optimal-value functions," *Machine Learning*, vol. 16, no. 3, pp. 227–233, 1994. 25, 152, 181

[76] M. G. Azar, R. Munos, and H. J. Kappen, "Minimax PAC bounds on the sample complexity of reinforcement learning with a generative model," *Mach. Learn.*, vol. 91, no. 3, pp. 325–349, 2013. [Online]. Available: https://doi.org/10.1007/s10994-013-5368-1 25, 31, 32, 181

[77] W. B. Haskell, R. Jain, and D. Kalathil, "Empirical dynamic programming," *Mathematics of Operations Research*, vol. 41, no. 2, pp. 402–429, 2016. 25, 181

[78] A. Sidford, M. Wang, X. Wu, L. F. Yang, and Y. Ye, "Near-optimal time and sample complexities for solving markov decision processes with a generative model," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2018, pp. 5192–5202. 25, 32, 181

[79] A. Agarwal, S. Kakade, and L. F. Yang, "Model-based reinforcement learning with a generative model is minimax optimal," in *Conference on Learning Theory*, 2020, pp. 67–83. 25, 32, 181

[80] D. Kalathil, V. S. Borkar, and R. Jain, "Empirical Q-Value Iteration," *Stochastic Systems*, vol. 11, no. 1, pp. 1–18, 2021. 25, 181

[81] M. Petrik and D. Subramanian, "Raam: The benefits of robustness in approximating aggregated mdps in reinforcement learning." in *NIPS*, 2014, pp. 1979–1987. 26

[82] Z. Zhou, Q. Bai, Z. Zhou, L. Qiu, J. Blanchet, and P. Glynn, "Finite-sample regret bound for distributionally robust offline tabular reinforcement learning," in *International Conference on Artificial Intelligence and Statistics*, 2021, pp. 3331–3339. 27, 33, 44, 159, 182

[83] W. Yang, L. Zhang, and Z. Zhang, "Towards theoretical understandings of robust markov decision processes: Sample complexity and asymptotics," *arXiv preprint arXiv:2105.03863*, 2021. 27, 44, 56, 72, 73, 182

[84] K. Zhou, J. C. Doyle, K. Glover *et al.*, *Robust and optimal control*. Prentice hall New Jersey, 1996, vol. 40. 27, 182

[85] G. E. Dullerud and F. Paganini, *A course in robust control theory: a convex approach*. Springer Science & Business Media, 2013, vol. 36. 27, 182

[86] K. Zhang, B. Hu, and T. Basar, "Policy optimization for H_2 linear control with H_∞ robustness guarantee: Implicit regularization and global convergence," in *Proceedings of the 2nd Annual Conference on Learning for Dynamics and Control, L4DC 2020, Online Event, Berkeley, CA, USA, 11-12 June 2020*, ser. Proceedings of Machine Learning Research, vol. 120, 2020, pp. 179–190. 27, 182

[87] V. S. Borkar, "Q-learning for risk-sensitive control," *Mathematics of operations research*, vol. 27, no. 2, pp. 294–311, 2002. 27, 182

[88] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020. 39, 209

[89] A. Antos, C. Szepesvári, and R. Munos, "Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path," *Machine Learning*, vol. 71, no. 1, pp. 89–129, 2008. 44, 79, 138, 181

[90] S. Lange, T. Gabel, and M. Riedmiller, "Batch reinforcement learning," in *Reinforcement learning*. Springer, 2012, pp. 45–73. 44, 79, 181

[91] J. Chen and N. Jiang, "Information-theoretic considerations in batch reinforcement learning," in *International Conference on Machine Learning*, 2019, pp. 1042–1051. 44, 49, 50, 54, 56, 79, 181

[92] T. Xie and N. Jiang, "Q* approximation schemes for batch reinforcement learning: A theoretical comparison," in *Conference on Uncertainty in Artificial Intelligence*, 2020, pp. 550–559. 44, 79, 181

[93] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *arXiv preprint arXiv:2005.01643*, 2020. 44, 79, 181, 186, 190, 191

[94] T. Xie, C.-A. Cheng, N. Jiang, P. Mineiro, and A. Agarwal, "Bellman-consistent pessimism for offline reinforcement learning," *Advances in neural information processing systems*, vol. 34, 2021. 44, 50, 54, 79, 181

[95] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration," in *International Conference on Machine Learning*, 2019, pp. 2052–2062. 44, 79, 181, 183, 184, 185, 186, 190

[96] A. Kumar, J. Fu, M. Soh, G. Tucker, and S. Levine, "Stabilizing off-policy q-learning via bootstrapping error reduction," in *Advances in Neural Information Processing Systems*, 2019, pp. 11 784–11 794. 44, 79, 181

[97] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1179–1191, 2020. 44, 79, 181

[98] T. Yu, G. Thomas, L. Yu, S. Ermon, J. Zou, S. Levine, C. Finn, and T. Ma, "Mopo: Model-based offline policy optimization," in *Advances in Neural Information Processing Systems*, 2020. 44, 79, 181

[99] S. Zhang and N. Jiang, "Towards hyperparameter-free policy selection for offline reinforcement learning," in *Advances in Neural Information Processing Systems*, 2021, pp. 12 864–12 875. 44, 79, 181

[100] G. J. Gordon, "Stable function approximation in dynamic programming," in *Machine Learning, Proceedings of the Twelfth International Conference on Machine Learning, 1995*, A. Prieditis and S. Russell, Eds., 1995, pp. 261–268. 44, 181

[101] D. Ernst, P. Geurts, and L. Wehenkel, "Tree-based batch mode reinforcement learning," *Journal of Machine Learning Research*, vol. 6, pp. 503–556, 2005. 44, 181

[102] A.-m. Farahmand, C. Szepesvári, and R. Munos, "Error propagation for approximate policy and value iteration," *Advances in Neural Information Processing Systems*, vol. 23, 2010. 44, 71, 181

[103] Y. Liu, A. Swaminathan, A. Agarwal, and E. Brunskill, "Provably good batch off-policy reinforcement learning without great exploration," in *Neural Information Processing Systems*, 2020. 44, 49, 181, 183, 184, 185, 186, 190, 191

[104] Y. Wang and S. Zou, "Online robust reinforcement learning with model uncertainty," *Advances in Neural Information Processing Systems*, vol. 34, pp. 7193–7206, 2021. 44, 62, 82

[105] C. Szepesvári and R. Munos, "Finite time bounds for sampling based fitted value iteration," in *Proceedings of the 22nd international conference on Machine learning*, 2005, pp. 880–887. 49

[106] A. Agarwal, N. Jiang, S. M. Kakade, and W. Sun, "Reinforcement learning: Theory and algorithms," *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep*, 2019. 50, 65, 66, 71, 90, 92, 94, 175, 179, 194, 201

[107] R. T. Rockafellar and R. J.-B. Wets, *Variational analysis*. Springer Science & Business Media, 2009, vol. 317. 53, 168, 169

[108] A. Shapiro, "Distributionally robust stochastic programming," *SIAM Journal on Optimization*, vol. 27, no. 4, pp. 2258–2275, 2017. 53, 169

[109] J. Duchi and H. Namkoong, "Learning models with uniform performance via distributionally robust optimization," *arXiv preprint arXiv:1810.08750*, 2018. 53, 62, 67, 68, 169, 194

[110] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International Conference on Machine Learning*, 2018, pp. 1582–1591. 59, 75, 189, 209

[111] A. Kis, L. Huber, and A. Wilkinson, "Social learning by imitation in a reptile (pogona vitticeps)," *Animal cognition*, vol. 18, 09 2014. 60

[112] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *Advances in Neural Information Processing Systems*, D. Touretzky, Ed., vol. 1. Morgan-Kaufmann, 1988. [Online]. Available: https://proceedings.neurips.cc/paper/1988/file/812b4ba287f5ee0bc9d43bbf5bbe87fb-Paper.pdf 60

[113] M. Bain and C. Sammut, "A framework for behavioural cloning," in *Machine Intelligence 15*, 1995. 60

[114] S. Ross and D. Bagnell, "Efficient reductions for imitation learning," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 661–668. 60, 65, 72, 73

[115] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 627–635. 60, 63, 66

[116] K. Brantley, W. Sun, and M. Henaff, "Disagreement-regularized imitation learning," in *International Conference on Learning Representations*, 2019. 61, 63, 66, 75

[117] N. Rajaraman, L. Yang, J. Jiao, and K. Ramchandran, "Toward the fundamental limits of imitation learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 2914–2924, 2020. 61, 63, 65, 66, 71, 73

[118] J. Ho and S. Ermon, "Generative adversarial imitation learning," *Advances in neural information processing systems*, vol. 29, 2016. 61

[119] M. A. Bashiri, B. Ziebart, and X. Zhang, "Distributionally robust imitation learning," *Advances in neural information processing systems*, vol. 34, pp. 24 404–24 417, 2021. 61

[120] A. Y. Ng and S. J. Russell, "Algorithms for inverse reinforcement learning," in *Proceedings of the Seventeenth International Conference on Machine Learning*, ser. ICML '00. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, p. 663–670. 61

[121] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, ser. AAAI'08. AAAI Press, 2008, p. 1433–1438. 61

[122] C. Finn, S. Levine, and P. Abbeel, "Guided cost learning: Deep inverse optimal control via policy optimization," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML'16. JMLR.org, 2016, p. 49–58. 61

[123] N. Rajaraman, Y. Han, L. F. Yang, K. Ramchandran, and J. Jiao, "Provably breaking the quadratic error compounding barrier in imitation learning, optimally," *arXiv preprint arXiv:2102.12948*, 2021. 61

[124] J. Chang, M. Uehara, D. Sreenivas, R. Kidambi, and W. Sun, "Mitigating covariate shift in imitation learning via offline data with partial coverage," *Advances in Neural Information Processing Systems*, vol. 34, pp. 965–979, 2021. 61, 70, 75

[125] Y. Xu, W. Gao, and D. Hsu, "Receding horizon inverse reinforcement learning," in *Advances in Neural Information Processing Systems*, A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, Eds., 2022. [Online]. Available: https://openreview.net/forum?id=CgkjJaKBvkX 61

[126] N. Corporation, "Closing the sim2real gap with nvidia isaac sim and nvidia isaac replicator," 2021. [Online]. Available: https://developer.nvidia.com/blog/closing-the-sim2real-gap-with-nvidia-isaac-sim-and-nvidia-isaac-replicator/ 62

[127] W. Yang, L. Zhang, and Z. Zhang, "Towards theoretical understandings of robust markov decision processes: Sample complexity and asymptotics," 2021. [Online]. Available: https://arxiv.org/abs/2105.03863 62

[128] K. Panaganti and D. Kalathil, "Sample complexity of robust reinforcement learning with a generative model," in *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, vol. 151. PMLR, 28–30 Mar 2022, pp. 9582–9602. [Online]. Available: https://proceedings.mlr.press/v151/panaganti22a.html 62

[129] A. Shapiro, "Distributionally robust stochastic programming," *SIAM Journal on Optimization*, vol. 27, no. 4, pp. 2258–2275, 2017. 62, 68

[130] R. Gao and A. Kleywegt, "Distributionally robust stochastic optimization with wasserstein distance," *Mathematics of Operations Research*, 2022. 62

[131] D. Bertsimas, V. Gupta, and N. Kallus, "Data-driven robust optimization," *Math. Program.*, vol. 167, no. 2, p. 235–292, feb 2018. [Online]. Available: https://doi.org/10.1007/s10107-017-1125-8 62

[132] H. Namkoong and J. C. Duchi, "Stochastic gradient methods for distributionally robust optimization with f-divergences," *Advances in neural information processing systems*, vol. 29, 2016. 62, 67

[133] J. Blanchet, Y. Kang, and K. Murthy, "Robust wasserstein profile inference and applications to machine learning," *Journal of Applied Probability*, vol. 56, no. 3, p. 830–857, 2019. 62

[134] B. Eysenbach and S. Levine, "Maximum entropy rl (provably) solves some robust rl problems," in *International Conference on Learning Representations*, 2022. 62

[135] S. Meinecke, L. Thurner, and M. Braun, "Review of steady-state electric power distribution system datasets," *Energies*, vol. 13, no. 18, 2020. [Online]. Available: https://www.mdpi.com/1996-1073/13/18/4826 62

[136] M. Travers, "10 best healthcare data sets & examples," 2021. [Online]. Available: https://www.cprime.com/resources/blog/10-best-healthcare-data-sets-examples/ 62

[137] S. Ettinger, S. Cheng, B. Caine, C. Liu, H. Zhao, S. Pradhan, Y. Chai, B. Sapp, C. Qi, Y. Zhou, Z. Yang, A. Chouard, P. Sun, J. Ngiam, V. Vasudevan, A. McCauley, J. Shlens, and D. Anguelov, "Large scale interactive motion forecasting for autonomous driving : The waymo open motion dataset," *arXiv*, 2021. 62

[138] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033. 63, 75

[139] J. Mei, C. Xiao, C. Szepesvari, and D. Schuurmans, "On the global convergence rates of softmax policy gradient methods," in *International Conference on Machine Learning*. PMLR, 2020, pp. 6820–6829. 64, 193

[140] B. O'Donoghue, I. Osband, and C. Ionescu, "Making sense of reinforcement learning and probabilistic inference," *arXiv preprint arXiv:2001.00805*, 2020. 64

[141] R. Chen, I. C. Paschalidis *et al.*, "Distributionally robust learning," *Foundations and Trends® in Optimization*, vol. 4, no. 1-2, pp. 1–243, 2020. 67

[142] C. Dann and E. Brunskill, "Sample complexity of episodic fixed-horizon reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 28, 2015. 71

[143] K. Panaganti and D. Kalathil, "Sample complexity of model-based robust reinforcement learning," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 2240–2245. 71, 82, 84

[144] A. Raffin, "Rl baselines3 zoo," https://github.com/DLR-RM/rl-baselines3-zoo, 2020. 75, 209

[145] S. Cabi, S. G. Colmenarejo, A. Novikov, K. Konyushkova, S. Reed, R. Jeong, K. Zolna, Y. Aytar, D. Budden, M. Vecerik *et al.*, "Scaling data-driven robotics with reward sketching and batch reinforcement learning," *arXiv preprint arXiv:1909.12200*, 2019. 79

[146] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine, "D4rl: Datasets for deep data-driven reinforcement learning," 2020. 79, 186, 190, 191

[147] W. Li, C. Pan, R. Zhang, J. Ren, Y. Ma, J. Fang, F. Yan, Q. Geng, X. Huang, H. Gong *et al.*, "Aads: Augmented autonomous driving simulation using data-driven algorithms," *Science robotics*, vol. 4, no. 28, p. eaaw0863, 2019. 79

[148] A. Amini, I. Gilitschenski, J. Phillips, J. Moseyko, R. Banerjee, S. Karaman, and D. Rus, "Learning robust control policies for end-to-end autonomous driving from data-driven simulation," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1143–1150, 2020. 79

[149] H. V. Dang, H. Tran-Ngoc, T. V. Nguyen, T. Bui-Tien, G. De Roeck, and H. X. Nguyen, "Data-driven structural health monitoring using feature fusion and hybrid deep learning," *IEEE Transactions on Automation Science and Engineering*, vol. 18, no. 4, pp. 2087–2103, 2020. 79

[150] J. Hu, A. Perer, and F. Wang, "Data driven analytics for personalized healthcare," *Healthcare Information Management Systems: Cases, Strategies, and Solutions*, pp. 529–554, 2016. 79

[151] Y. Jin, Z. Yang, and Z. Wang, "Is pessimism provably efficient for offline rl?" in *International Conference on Machine Learning*. PMLR, 2021, pp. 5084–5096. 79, 80, 82, 89

[152] P. Rashidinejad, B. Zhu, C. Ma, J. Jiao, and S. Russell, "Bridging offline reinforcement learning and imitation learning: A tale of pessimism," *Advances in Neural Information Processing Systems*, vol. 34, pp. 11 702–11 716, 2021. 79, 80, 82, 88

[153] G. Li, L. Shi, Y. Chen, Y. Chi, and Y. Wei, "Settling the sample complexity of model-based offline reinforcement learning," *arXiv preprint arXiv:2204.05275*, 2022. 80, 82

[154] G. Li, L. Shi, Y. Chen, Y. Gu, and Y. Chi, "Breaking the sample complexity barrier to regret-optimal model-free reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 17 762–17 776, 2021. 80, 82

[155] P. Rashidinejad, H. Zhu, K. Yang, S. Russell, and J. Jiao, "Optimal conservative offline rl with general function approximation via augmented lagrangian," *arXiv preprint arXiv:2211.00716*, 2022. 80

[156] M. Uehara and W. Sun, "Pessimistic model-based offline reinforcement learning under partial coverage," *arXiv preprint arXiv:2107.06226*, 2023. 80, 81, 82, 84, 90, 92, 95

[157] S. Boucheron, G. Lugosi, and P. Massart, *Concentration Inequalities: A Nonasymptotic Theory of Independence*. OUP Oxford, 2013. [Online]. Available: https://books.google.com/books?id=koNqWRluhP0C 85

[158] X. Ma, Z. Liang, L. Xia, J. Zhang, J. Blanchet, M. Liu, Q. Zhao, and Z. Zhou, "Distributionally robust offline reinforcement learning with linear function approximation," *arXiv preprint arXiv:2209.06620*, 2022. 90

[159] B. K. Sriperumbudur, K. Fukumizu, A. Gretton, B. Schölkopf, and G. R. Lanckriet, "On integral probability metrics,\phi-divergences and binary classification," *arXiv preprint arXiv:0901.2698*, 2009. 91

[160] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3, pp. 229–256, 1992. 98

[161] S. M. Kakade, "A natural policy gradient," in *Advances in neural information processing systems*, vol. 14, 2001, pp. 1531–1538. 98

[162] A. Agarwal, S. M. Kakade, J. D. Lee, and G. Mahajan, "On the theory of policy gradient methods: Optimality, approximation, and distribution shift," *Journal of Machine Learning Research*, vol. 22, no. 98, pp. 1–76, 2021. 98

[163] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., NJ, 2005. 120

[164] R. Vershynin, *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge University press, 2018, vol. 47. 147, 167

[165] A. Maurer and M. Pontil, "Empirical bernstein bounds and sample-variance penalization," in *COLT 2009 - The 22nd Conference on Learning Theory, Montreal, Quebec, Canada, June 18-21, 2009*, 2009. 147

[166] R. van Handel, "Probability in High Dimension," Princeton University NJ, Tech. Rep., 2014. 148

[167] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014. 167

[168] I. Csiszár, "Information-type measures of difference of probability distributions and indirect observation," *studia scientiarum Mathematicarum Hungarica*, vol. 2, pp. 229–318, 1967. 169, 194

[169] A. K. Moses and R. Sundaresan, "Further results on geometric properties of a family of relative entropies," in *2011 IEEE International Symposium on Information Theory Proceedings*, 2011, pp. 1940–1944. 169

[170] L. A. Prashanth and M. Ghavamzadeh, "Variance-constrained actor-critic algorithms for discounted and average reward mdps," *Mach. Learn.*, vol. 105, no. 3, pp. 367–417, 2016. 182

[171] Y. Fei, Z. Yang, Y. Chen, and Z. Wang, "Exponential bellman equation and improved regret bounds for risk-sensitive reinforcement learning," in *Annual Conference on Neural Information Processing Systems 2021*, 2021, pp. 20 436–20 446. 182

[172] Y. Zhang, Z. Yang, and Z. Wang, "Provably efficient actor-critic for risk-sensitive and robust adversarial rl: A linear-quadratic case," in *International Conference on Artificial Intelligence and Statistics*, 2021, pp. 2764–2772. 182

[173] P. Huang, M. Xu, F. Fang, and D. Zhao, "Robust reinforcement learning as a stackelberg game via adaptively-regularized adversarial training," *arXiv preprint arXiv:2202.09514*, 2022. 182

[174] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013. 184

[175] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. 185, 209

[176] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017. 186

[177] A. Raffin, "Rl baselines3 zoo," https://github.com/DLR-RM/rl-baselines3-zoo, 2020. 186

[178] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*, 2018, pp. 1861–1870. 186

[179] Y. Chow, O. Nachum, and M. Ghavamzadeh, "Path consistency learning in tsallis entropy regularized mdps," in *International conference on machine learning*. PMLR, 2018, pp. 979–988. 193

[180] T. M. Cover and J. A. Thomas, "Information theory and the stock market," *Elements of Information Theory. Wiley Inc., New York*, pp. 543–556, 1991. 194, 203

[181] A. L. Gibbs and F. E. Su, "On choosing and bounding probability metrics," *International statistical review*, vol. 70, no. 3, pp. 419–435, 2002. 194

[182] A. Basu, H. Shioya, and C. Park, *Statistical Inference: The Minimum Distance Approach*, ser. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. CRC Press, 2011. [Online]. Available: https://books.google.com/books?id=C-lOIgDp5_0C 194

[183] D. Bertsekas, *Reinforcement learning and optimal control*. Athena Scientific, 2019. 200

[184] D. Kraft, "A software package for sequential quadratic programming," *Forschungsbericht-Deutsche Forschungs- und Versuchsanstalt fur Luft- und Raumfahrt*, 1988. 209

APPENDIX A

APPENDIX FOR CHAPTER 2*

In this appendix, we include complete proofs, all experiments, and all supporting details for the corresponding chapter.

A.1 Proofs of the Results in Section 2.3.1

A.1.1 Proof of Proposition 1

We first restate Proposition 1 formally and then give the proof.

Proposition 9. *(i) For any $V_1, V_2 \in \mathbb{R}^{|\mathcal{S}|}$ and $\lambda \in [0,1)$, $\|T_\pi^{(\lambda)}V_1 - T_\pi^{(\lambda)}V_2\|_\infty \leq \frac{\alpha(1-\lambda)}{(1-\alpha\lambda)}\|V_1 - V_2\|_\infty$. So, $T_\pi^{(\lambda)}$ is a contraction in sup norm for any $\alpha \in (0,1), \lambda \in [0,1)$.*
(ii) The robust value function V_π is the unique fixed point of $T_\pi^{(\lambda)}$, i.e., $T_\pi^{(\lambda)}V_\pi = V_\pi$, for all $\alpha \in (0,1)$ and $\lambda \in [0,1)$.

Proof. From (2.6) we have

$$\begin{aligned}\|T_\pi^{(\lambda)}(V_1) - T_\pi^{(\lambda)}(V_2)\|_\infty &= \|(1-\lambda)\sum_{m=0}^{\infty}\lambda^m(T_\pi^{m+1}(V_1) - T_\pi^{m+1}(V_2))\|_\infty \\ &\leq (1-\lambda)\sum_{m=0}^{\infty}\lambda^m\|T_\pi^{m+1}(V_1) - T_\pi^{m+1}(V_2)\|_\infty \\ &\overset{(a)}{\leq} (1-\lambda)\sum_{m=0}^{\infty}\lambda^m\alpha^{m+1}\|V_1 - V_2\|_\infty = \frac{\alpha(1-\lambda)}{(1-\alpha\lambda)}\|V_1 - V_2\|_\infty\end{aligned}$$

where (a) follows since T_π is a contraction operator with contraction modulus α. This proves (i). Since V_π is the unique fixed point of T_π, (ii) directly follows from (i) and the Banach Fixed Point Theorem [163, Theorem 6.2.3]. □

*Reprinted with permission from Kishan Panaganti, Dileep Kalathil, "Robust Reinforcement Learning using Least Squares Policy Iteration with Provable Performance Guarantees." International Conference on Machine Learning. PMLR, 2021.

A.1.2 Proof of Proposition 2

We first restate Proposition 2 formally and then give the proof.

Proposition 10. *Under Assumption 2, for any* $V_1, V_2 \in \mathbb{R}^{|\mathcal{S}|}$ *and* $\lambda \in [0, 1)$, $\|\Pi T_\pi^{(\lambda)} V_1 - \Pi T_\pi^{(\lambda)} V_2\|_d \leq \frac{\beta(1-\lambda)}{(1-\beta\lambda)} \|V_1 - V_2\|_d$. *So,* $\Pi T_\pi^{(\lambda)}$ *is a contraction mapping in* d*-weighted Euclidean norm for any* $\beta \in (0, 1), \lambda \in [0, 1)$.

Proof. From (2.6) we have

$$\begin{aligned}
\|T_\pi^{(\lambda)}(V_1) - T_\pi^{(\lambda)}(V_2)\|_d &= \|(1-\lambda)\sum_{m=0}^{\infty} \lambda^m (T_\pi^{m+1}(V_1) - T_\pi^{m+1}(V_2))\|_d \\
&\leq (1-\lambda)\sum_{m=0}^{\infty} \lambda^m \|T_\pi^{m+1}(V_1) - T_\pi^{m+1}(V_2)\|_d \\
&\overset{(a)}{\leq} (1-\lambda)\sum_{m=0}^{\infty} \lambda^m \beta^{m+1} \|V_1 - V_2\|_d = \frac{\beta(1-\lambda)}{(1-\beta\lambda)} \|V_1 - V_2\|_d
\end{aligned}$$

where (a) follows since ΠT_π is a contraction in the d-weighted Euclidean norm with contraction modulus β [39, Corollary 4]. From [64], Π is a nonexpansive mapping in the d-weighted Euclidean norm. So, $\Pi T_\pi^{(\lambda)}$ has the stated proptery. □

A.1.3 Derivation of (2.7)

Given w_k, w_{k+1} which satisfies the equation $\Phi w_{k+1} = \Pi T_\pi^{(\lambda)} \Phi w_k$ can be written as the solution of the minimization problem $w_{k+1} = \arg\min_w \|\Phi w - T_\pi^{(\lambda)} \Phi w_k\|_d^2$. Taking gradient w.r.t. w and equating to zero, we get $\Phi^\top D(\Phi w - T_\pi^{(\lambda)} \Phi w_k) = 0$, which implies $w_{k+1} = (\Phi^\top D \Phi)^{-1} \Phi^\top D T_\pi^{(\lambda)} \Phi w_k$. This can be written as

$$\begin{aligned}
w_{k+1} &= (\Phi^\top D \Phi)^{-1} \Phi^\top D T_\pi^{(\lambda)} \Phi w_k = w_k + (\Phi^\top D \Phi)^{-1} (\Phi^\top D T_\pi^{(\lambda)} \Phi w_k - \Phi^\top D \Phi w_k) \\
&= w_k + (\Phi^\top D \Phi)^{-1} \Phi^\top D (T_\pi^{(\lambda)} \Phi w_k - \Phi w_k).
\end{aligned}$$

A.1.4 Derivation of (2.8)

We have

$$TV = r_\pi + \alpha P_\pi^o V + \alpha \sigma_{\mathcal{U}_\pi}(V) \tag{A.1}$$

$$\begin{aligned} T^2V &= r_\pi + \alpha P_\pi^o (TV) + \alpha \sigma_{\mathcal{U}_\pi}(TV) \\ &= (1+\alpha P) r_\pi + (\alpha P_\pi^o)^2 V + \alpha (\alpha P_\pi^o) \sigma_{\mathcal{U}_\pi}(V) + \alpha \sigma_{\mathcal{U}_\pi}(TV). \end{aligned}$$

Suppose we have an induction hypothesis for $T^m V$, i.e.,

$$T^m V = \sum_{k=0}^{m-1} (\alpha P_\pi^o)^k r_\pi + (\alpha P_\pi^o)^m V + \alpha \sum_{k=0}^{m-1} (\alpha P_\pi^o)^k \sigma_{\mathcal{U}_\pi}(T^{(m-1-k)} V).$$

Now, to verify an induction step, observe that,

$$\begin{aligned} T^{m+1}V &= \sum_{k=0}^{m-1} (\alpha P_\pi^o)^k r_\pi + (\alpha P_\pi^o)^m TV + \alpha \sum_{k=0}^{m-1} (\alpha P_\pi^o)^k \sigma_{\mathcal{U}_\pi}(T^{(m-1-k)} TV) \\ &\overset{(a)}{=} \sum_{k=0}^{m-1} (\alpha P_\pi^o)^k r_\pi + (\alpha P_\pi^o)^m (r + \alpha P_\pi^o V + \alpha \sigma_{\mathcal{U}_\pi}(V)) + \alpha \sum_{k=0}^{m-1} (\alpha P_\pi^o)^k \sigma_{\mathcal{U}_\pi}(T^{(m-1-k)} TV) \\ &= \sum_{k=0}^{m} (\alpha P_\pi^o)^k r_\pi + (\alpha P_\pi^o)^{m+1} V + \alpha \sum_{k=0}^{m} (\alpha P_\pi^o)^k \sigma_{\mathcal{U}_\pi}(T^{(m-k)} V) \end{aligned}$$

where (a) follows from (A.1). Now, (2.8) directly follows from (2.6).

A.2 Proofs of the Results in Section 2.3.2

A.2.1 Proof of Proposition 3

Proof. With $\widehat{\mathcal{U}}_\pi = \mathcal{U}_\pi$, we have,

$$\begin{aligned} \widetilde{T}_\pi^{(\lambda)}(V_\pi) &= (1-\lambda) \sum_{m=0}^{\infty} \lambda^m \left[\sum_{t=0}^{m} (\alpha P_\pi^o)^t r_\pi + \alpha \sum_{t=0}^{m} (\alpha P_\pi^o)^t \, \sigma_{\mathcal{U}_\pi}(V_\pi) + (\alpha P_\pi^o)^{m+1} V_\pi \right] \\ &\overset{(a)}{=} (1-\lambda) \sum_{m=0}^{\infty} \lambda^m \left[\sum_{t=0}^{m} (\alpha P_\pi^o)^t (r_\pi + \alpha \sigma_{\mathcal{U}_\pi}(V_\pi)) + (\alpha P_\pi^o)^{m+1} V_\pi \right] \end{aligned}$$

$$= (1-\lambda)\sum_{m=0}^{\infty}\lambda^m\left[\sum_{t=0}^{m}(\alpha P_\pi^o)^t(I-\alpha P_\pi^o)V_\pi + (\alpha P_\pi^o)^{m+1}V_\pi\right]$$
$$\overset{(b)}{=} (1-\lambda)\sum_{m=0}^{\infty}\lambda^m V_\pi = V_\pi.$$

Here, to get (a), we wrote $r_\pi + \alpha\sigma_{\mathcal{U}_\pi}(V_\pi) = (I - \alpha P_\pi^o)V_\pi$ since $T_\pi V_\pi = V_\pi$. (b) is from the telescopic sum of the previous equation. □

A.2.2 Proof of Theorem 1

We have the following lemma which is similar to Lemma 4.2 in [40].

Lemma 11. *For any vector $V \in \mathbb{R}^{|\mathcal{S}|}$ and for all $(s,a) \in \mathcal{S}\times\mathcal{A}$,*

$$|\sigma_{\widehat{\mathcal{U}}_{s,a}}(V) - \sigma_{\mathcal{U}_{s,a}}(V)| \le \rho\|V\|_d.$$

Proof. First note that, for any $x, y \in \mathbb{R}^{|\mathcal{S}|}$ we have

$$x^\top y \;\le\; (x^\top D y)/d_{\min} \;\overset{(a)}{\le}\; (\|x\|_d\|y\|_d)/d_{\min}, \tag{A.2}$$

where (a) follows from Cauchy-Schwarz inequality with respect to $\|\cdot\|_d$ norm.

Consider any $p \in \widehat{\mathcal{U}}_{s,a}$ and $q \in \mathcal{U}_{s,a}\setminus\widehat{\mathcal{U}}_{s,a}$. For any $V \in \mathbb{R}^{|\mathcal{S}|}$, we have

$$p^\top V = q^\top V + (p-q)^\top V \ge \sigma_{\mathcal{U}_{s,a}}(V) + (p-q)^\top V \ge \sigma_{\mathcal{U}_{s,a}}(V) + \min_{x\in\widehat{\mathcal{U}}_{s,a}}\min_{y\in\mathcal{U}_{s,a}\setminus\widehat{\mathcal{U}}_{s,a}}(x-y)^\top V$$
$$= \sigma_{\mathcal{U}_{s,a}}(V) - \max_{x\in\widehat{\mathcal{U}}_{s,a}}\max_{y\in\mathcal{U}_{s,a}\setminus\widehat{\mathcal{U}}_{s,a}}(y-x)^\top V \overset{(b)}{\ge} \sigma_{\mathcal{U}_{s,a}}(V) - \max_{x\in\widehat{\mathcal{U}}_{s,a}}\max_{y\in\mathcal{U}_{s,a}\setminus\widehat{\mathcal{U}}_{s,a}}(\|y-x\|_d\|V\|_d)/d_{\min}$$
$$\ge \sigma_{\mathcal{U}_{s,a}}(V) - \rho_{s,a}\|V\|_d,$$

where (b) follows from (A.2). By taking infimum on both sides with respect to $p \in \widehat{\mathcal{U}}_{s,a}$, we get,

$$\sigma_{\widehat{\mathcal{U}}_{s,a}}(V) \ge \sigma_{\mathcal{U}_{s,a}}(V) - \rho_{s,a}\|V\|_d.$$

We can also get $\sigma_{\mathcal{U}_{s,a}}(V) \geq \sigma_{\widehat{\mathcal{U}}_{s,a}}(V) - \rho_{s,a}\|V\|_d$ by similar arguments. Combining, we get,

$$|\sigma_{\widehat{\mathcal{U}}_{s,a}}(V) - \sigma_{\mathcal{U}_{s,a}}(V)| \leq \rho_{s,a}\|V\|_d.$$

Since $\rho = \max_{s,a} \rho_{s,a}$, we get the desired result. □

We will use the following result which follows directly from [64, Lemma 1].

Lemma 12. *Under Assumption 2, for any V, we have $\|P^o_{\pi_e} V\|_d \leq \|V\|_d$.*

Remark 12. *The inequality in the Assumption 2 can be written as, $\alpha P^o_{s,\pi(s)}(s') + \alpha U_{s,\pi(s)}(s') \leq \beta P^o_{s,\pi_e(s)}(s')$. From this, we can conclude that $\alpha U_{s,\pi(s)}(s') \leq \beta P^o_{s,\pi_e(s)}(s')$.*

Next, we show the following.

Lemma 13. *For any $V_1, V_2 \in \mathbb{R}^{|\mathcal{S}|}$,*

$$\|\sigma_{\widehat{\mathcal{U}}_\pi}(V_1) - \sigma_{\widehat{\mathcal{U}}_\pi}(V_2)\|_d \leq \left(\frac{\beta}{\alpha} + \rho\right)\|V_1 - V_2\|_d.$$

Proof. For any $s \in \mathcal{S}$ we have

$$\sigma_{\widehat{\mathcal{U}}_{s,\pi(s)}}(V_2) - \sigma_{\widehat{\mathcal{U}}_{s,\pi(s)}}(V_1) = \inf_{q \in \widehat{\mathcal{U}}_{s,\pi(s)}} q^\top V_2 - \inf_{\tilde{q} \in \widehat{\mathcal{U}}_{s,\pi(s)}} \tilde{q}^\top V_1 = \inf_{q \in \widehat{\mathcal{U}}_{s,\pi(s)}} \sup_{\tilde{q} \in \widehat{\mathcal{U}}_{s,\pi(s)}} q^\top V_2 - \tilde{q}^\top V_1$$

$$\geq \inf_{q \in \widehat{\mathcal{U}}_{s,\pi(s)}} q^\top (V_2 - V_1) = \sigma_{\widehat{\mathcal{U}}_{s,\pi(s)}}(V_2 - V_1) \overset{(a)}{\geq} \sigma_{\mathcal{U}_{s,\pi(s)}}(V_2 - V_1) - \rho\|V_1 - V_2\|_d, \tag{A.3}$$

where (a) follows from Lemma 11. By definition, for any arbitrary $\varepsilon > 0$, there exists a $U_{s,\pi(s)} \in \mathcal{U}_{s,\pi(s)}$ such that

$$U^\top_{s,\pi(s)}(V_2 - V_1) - \varepsilon \leq \sigma_{\mathcal{U}_{s,\pi(s)}}(V_2 - V_1). \tag{A.4}$$

Using (A.4) and (A.3),

$$\alpha(\sigma_{\widehat{\mathcal{U}}_{s,\pi(s)}}(V_1) - \sigma_{\widehat{\mathcal{U}}_{s,\pi(s)}}(V_2)) \leq \alpha U^\top_{s,\pi(s)}(V_1 - V_2) + \alpha\varepsilon + \rho\alpha\|V_1 - V_2\|_d$$

$$\leq \alpha U^{\top}_{s,\pi(s)}|(V_2 - V_1)| + \alpha\varepsilon + \rho\alpha\|V_1 - V_2\|_d$$
$$\overset{(b)}{\leq} \beta(P^o_{s,\pi_e(s)})^{\top}|(V_1 - V_2)| + \alpha\varepsilon + \rho\alpha\|V_1 - V_2\|_d \tag{A.5}$$

where (b) follows from Remark 12. Since ε is arbitrary, we get,

$$\alpha(\sigma_{\widehat{\mathcal{U}}_{s,\pi(s)}}(V_1) - \sigma_{\widehat{\mathcal{U}}_{s,\pi(s)}}(V_2)) \leq \beta(P^o_{s,\pi_e(s)})^{\top}|(V_1 - V_2)| + \rho\alpha\|V_1 - V_2\|_d.$$

By exchanging the roles of V_1 and V_2, we get $\alpha(\sigma_{\widehat{\mathcal{U}}_{s,\pi(s)}}(V_2) - \sigma_{\widehat{\mathcal{U}}_{s,\pi(s)}}(V_1)) \leq \beta(P^o_{s,\pi_e(s)})^{\top}|(V_1 - V_2)| + \rho\alpha\|V_1 - V_2\|_d$. Combining these and writing compactly in vector form, we get

$$\alpha|\sigma_{\widehat{\mathcal{U}}_\pi}(V_1) - \sigma_{\widehat{\mathcal{U}}_\pi}(V_2)| \leq \beta P^o_{\pi_e}|V_1 - V_2| + \rho\alpha\|V_1 - V_2\|_d\mathbf{1},$$

where $\mathbf{1} = (1, 1, \ldots, 1)^{\top}$, an $|\mathcal{S}|$-dimensional unit vector. Since $\alpha|\sigma_{\widehat{\mathcal{U}}_\pi}(V_1) - \sigma_{\widehat{\mathcal{U}}_\pi}(V_2)| \geq 0$, by the property of the norm, we get

$$\begin{aligned}\alpha\|\sigma_{\widehat{\mathcal{U}}_\pi}(V_1) - \sigma_{\widehat{\mathcal{U}}_\pi}(V_2)\|_d &\leq \|\beta P^o_{\pi_e}|V_1 - V_2| + \rho\alpha\|V_1 - V_2\|_d\mathbf{1}\|_d \\ &\overset{(c)}{\leq} \beta\|P^o_{\pi_e}|V_1 - V_2|\|_d + \rho\alpha\|V_1 - V_2\|_d\|\mathbf{1}\|_d \\ &\overset{(d)}{\leq} \beta\|V_1 - V_2\|_d + \rho\alpha\|V_1 - V_2\|_d\end{aligned}$$

where (c) follows from triangle inequality and (d) from Lemma 12 and from the fact that and $\|\mathbf{1}\|_d = 1$. Dividing by α and rearranging, we get the desired result. □

Proof of Theorem 1. We first observe

$$\|\alpha P^o_\pi|V|\|_d \overset{(a)}{\leq} \|\beta P^o_{\pi_e}|V|\|_d \overset{(b)}{\leq} \beta\|V\|_d, \tag{A.6}$$

where (a) follows from the Assumption 2 and (b) from the Lemma 12.

Notice that for any finite t we have,

$$\|(\alpha P_\pi^o)^t|V|\|_d = \|\alpha P_\pi^o(\alpha P_\pi^o)^{t-1}|V|\|_d \overset{(c)}{\leq} \beta\|(\alpha P_\pi^o)^{t-1}|V|\|_d \tag{A.7}$$

where (c) follows from (A.6). Using this repeatedly, we get,

$$\|(\alpha P_\pi^o)^t|V|\|_d \leq \beta^t\|V\|_d. \tag{A.8}$$

Now,

$$\begin{aligned}
&\|\widetilde{T}_\pi^{(\lambda)}(V_1) - \widetilde{T}_\pi^{(\lambda)}(V_2)\|_d \\
&= \|(1-\lambda)\sum_{m=0}^{\infty}\lambda^m[\alpha\sum_{t=0}^{m}(\alpha P_\pi^o)^t\,(\sigma_{\widehat{\mathcal{U}_\pi}}(V_1) - \sigma_{\widehat{\mathcal{U}_\pi}}(V_2)) + (\alpha P_\pi^o)^{m+1}(V_1 - V_2)]\|_d \\
&\leq (1-\lambda)\sum_{m=0}^{\infty}\lambda^m[\alpha\sum_{t=0}^{m}\|(\alpha P_\pi^o)^t|\sigma_{\widehat{\mathcal{U}_\pi}}(V_1) - \sigma_{\widehat{\mathcal{U}_\pi}}(V_2)|\|_d + \|(\alpha P_\pi^o)^{m+1}|V_1 - V_2|\|_d] \\
&\overset{(d)}{\leq} (1-\lambda)\sum_{m=0}^{\infty}\lambda^m[\alpha\sum_{t=0}^{m}(\beta)^t\,\|\sigma_{\widehat{\mathcal{U}_\pi}}(V_1) - \sigma_{\widehat{\mathcal{U}_\pi}}(V_2)\|_d + \beta^{m+1}\|V_1 - V_2\|_d] \\
&\overset{(e)}{\leq} (1-\lambda)\sum_{m=0}^{\infty}\lambda^m[(\beta+\rho\alpha)\sum_{t=0}^{m}(\beta)^t\,\|(V_1 - V_2)\|_d + \beta^{m+1}\|(V_1 - V_2)\|_d] \qquad \text{(A.9)} \\
&= \left[\frac{(\beta+\rho\alpha)}{(1-\beta)}\left(1 - \frac{\beta(1-\lambda)}{(1-\beta\lambda)}\right) + \frac{\beta(1-\lambda)}{(1-\beta\lambda)}\right]\|(V_1 - V_2)\|_d \\
&= \frac{\beta(2-\lambda)+\rho\alpha}{(1-\beta\lambda)}\|(V_1 - V_2)\|_d, \qquad \text{(A.10)}
\end{aligned}$$

where (d) follows from (A.8) and (e) follows from Lemma 13.

From [62], Π is a non-expansive operator in $\|\cdot\|_d$. Thus,

$$\|\Pi\widetilde{T}_\pi^{(\lambda)}V_1 - \Pi\widetilde{T}_\pi^{(\lambda)}V_2\|_d \leq \|\widetilde{T}_\pi^{(\lambda)}(V_1) - \widetilde{T}_\pi^{(\lambda)}(V_2)\|_d \leq c(\alpha,\beta,\rho,\lambda)\,\|V_1 - V_2\|_d. \tag{A.11}$$

where $c(\alpha,\beta,\rho,\lambda) = (\beta(2-\lambda)+\rho\alpha)/(1-\beta\lambda)$. This concludes the proof of getting (2.10).

For proving (2.11), first denote the operator $\widetilde{T}_{\pi}^{(\lambda)}$ as $\bar{T}_{\pi}^{(\lambda)}$ when $\widehat{\mathcal{U}}_{\pi} = \mathcal{U}_{\pi}$. Now observe that

$$\begin{aligned}
\|\bar{T}_{\pi}^{(\lambda)}(V) - \widetilde{T}_{\pi}^{(\lambda)}(V)\|_d &= \|(1-\lambda)\sum_{m=0}^{\infty}\lambda^m[\alpha\sum_{t=0}^{m}(\alpha P_{\pi}^{o})^t\,(\sigma_{\mathcal{U}_{\pi}}(V) - \sigma_{\widehat{\mathcal{U}}_{\pi}}(V))]\|_d \\
&\leq (1-\lambda)\sum_{m=0}^{\infty}\lambda^m[\alpha\sum_{t=0}^{m}\|(\alpha P_{\pi}^{o})^t\,|(\sigma_{\mathcal{U}_{\pi}}(V) - \sigma_{\widehat{\mathcal{U}}_{\pi}}(V))|\|_d] \\
&\overset{(f)}{\leq} (1-\lambda)\sum_{m=0}^{\infty}\lambda^m[\alpha\sum_{t=0}^{m}\beta^t\|\sigma_{\mathcal{U}_{\pi}}(V) - \sigma_{\widehat{\mathcal{U}}_{\pi}}(V)\|_d] \\
&\overset{(g)}{\leq} (1-\lambda)\sum_{m=0}^{\infty}\lambda^m[\frac{(1-\beta^{m+1})}{1-\beta}\,\alpha\rho\|V\|_d] = \frac{\alpha\rho\|V\|_d}{1-\beta\lambda} \qquad \text{(A.12)}
\end{aligned}$$

where (f) follows (A.8) and (g) from Lemma 11.

Now,

$$\begin{aligned}
\|V_{\pi} - \Phi w_{\pi}\|_d &\leq \|V_{\pi} - \Pi V_{\pi}\|_d + \|\Pi V_{\pi} - \Phi w_{\pi}\|_d \\
&\overset{(h)}{=} \|V_{\pi} - \Pi V_{\pi}\|_d + \|\Pi\bar{T}_{\pi}^{(\lambda)}V_{\pi} - \Pi\widetilde{T}_{\pi}^{(\lambda)}\Phi w_{\pi}\|_d \\
&\overset{(i)}{\leq} \|V_{\pi} - \Pi V_{\pi}\|_d + \|\Pi\bar{T}_{\pi}^{(\lambda)}V_{\pi} - \Pi\widetilde{T}_{\pi}^{(\lambda)}V_{\pi}\|_d + \|\Pi\widetilde{T}_{\pi}^{(\lambda)}V_{\pi} - \Pi\widetilde{T}_{\pi}^{(\lambda)}\Phi w_{\pi}\|_d \\
&\overset{(j)}{\leq} \|V_{\pi} - \Pi V_{\pi}\|_d + \frac{\beta\rho\|V_{\pi}\|_d}{1-\beta\lambda} + c(\alpha,\beta,\rho,\lambda)\|V_{\pi} - \Phi w_{\pi}\|_d
\end{aligned}$$

We get (h) because $\bar{T}_{\pi}^{(\lambda)}V_{\pi} = V_{\pi}$ from Proposition 3 and $\Pi\widetilde{T}_{\pi}^{(\lambda)}\Phi w_{\pi} = \Phi w_{\pi}$ by the premise of the proposition, (i) by triangle inequality, (j) from (A.12) and (2.10). Rearranging, we get,

$$\|V_{\pi} - \Phi w_{\pi}\|_d \leq \frac{1}{1 - c(\alpha,\beta,\rho,\lambda)}\left(\|V_{\pi} - \Pi V_{\pi}\|_d + \frac{\beta\rho\|V_{\pi}\|_d}{1-\beta\lambda}\right). \qquad \text{(A.13)}$$

This completes the proof of Theorem 1. □

A.2.3 Derivation of (2.13)

For any bounded mapping W, observe that

$$\sum_{t=0}^{\infty}(\alpha\lambda P_{\pi}^{o})^t W = (1-\lambda)\sum_{m=0}^{\infty}\lambda^m\sum_{t=0}^{m}(\alpha P_{\pi}^{o})^t W,$$

yielded by exchanging summations. Using this observation with equations (2.9) and (2.12) we get (2.13).

A.2.4 Proof of Theorem 2

To prove this theorem, we will use the following result from [61]. Let $\|\cdot\|$ denote the standard Euclidean norm.

Proposition 11. *[61, Proposition 4.1] Consider a sequence $\{x_t\}$ generated by the update equation*

$$x_{t+1} = x_t + \gamma_t(h_t(x_t) + e_t),$$

where $h_t : \mathbb{R}^n \to \mathbb{R}^n$, γ_t is a positive deterministic stepsize, and e_t is a random noise vector. Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continously differentiable function. Assume the following:
(i) Function f is positive, i.e., $f(x) \geq 0$. Also, f has Lipschitz continuous gradient, i.e., there exists some scalar $L > 0$ such that

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|, \ \ \forall\, x, y \in \mathbb{R}^n. \tag{A.14}$$

(ii) Let $\mathcal{F}_t = \{x_0, x_1, \ldots, x_t\}$. There exists positive scalars c_1, c_2, and c_3 such that

$$(\nabla f(x_t))^\top \mathbb{E}[h_t(x_t)|\mathcal{F}_t] \leq -c_1\|\nabla f(x_t)\|^2, \ \ \forall\, t, \tag{A.15}$$

$$\|\mathbb{E}[e_t|\mathcal{F}_t]\| \leq c_2\varepsilon_t(1 + \|\nabla f(x_t)\|), \ \ \forall\, t, \tag{A.16}$$

$$\mathbb{E}[\|h_t(x_t) + e_t\|^2|\mathcal{F}_t] \leq c_3(1 + \|\nabla f(x_t)\|^2), \ \ \forall\, t, \tag{A.17}$$

where ε_t is a positive deterministic scalar.

(ii) The deterministic sequences $\{\gamma_t\}$ and $\{\varepsilon_t\}$ satsify

$$\sum_{t=0}^{\infty} \gamma_t = \infty, \ \sum_{t=0}^{\infty} \gamma_t^2 < \infty, \ \sum_{t=0}^{\infty} \gamma_t\varepsilon_t^2 < \infty, \ \lim_{t\to\infty} \varepsilon_t = 0. \tag{A.18}$$

Then, with probability 1:

(i) The sequence $\{f(x_t)\}$ converges.

(ii) The sequence $\{\nabla f(x_t)\}$ converges to zero.

(iii) Every limit point of $\{x_t\}$ is a stationary point of f.

Using the above result, we now prove Theorem 2.

Proof of Theorem 2. We rewrite (2.18) as $w_{t+1} = w_t + \gamma_t(h_t(w_t) + e_t)$, where,

$$h_t(w_t) = B^{-1}(Aw_t + b + C(w_t)),$$
$$e_t = B_t^{-1}(A_t w_t + b_t + C_t(w_t)) - B^{-1}(Aw_t + b + C(w_t)).$$

Define

$$f(w) = \frac{1}{2}(w - w_\pi)^\top \Phi^\top D\Phi(w - w_\pi).$$

We now verify the conditions given in Proposition 11.

Verifying (A.14)*:* By definition $f(w) \geq 0$. Also, $\nabla f(w) = \Phi^\top D\Phi(w - w_\pi)$. Hence,

$$\|\nabla f(w) - \nabla f(w')\| = \|\Phi^\top D\Phi(w - w')\| \leq \|\Phi^\top D\Phi\|_{\text{op}}\|w - w'\|,$$

where $\|\cdot\|_{\text{op}}$ is the operator norm corresponding to the Euclidean space. Set $L = \|\Phi^\top D\Phi\|_{\text{op}}$. From (ii) in Assumption 2 and Φ being full rank, we know that $\Phi^\top D\Phi$ is a positive definite matrix and hence L is a positive scalar. This verifies (A.14).

Verifying (A.15)*:* It is straightforward to verify that $Aw_t + b + C(w_t) = \Phi^\top D(\widetilde{T}_\pi^{(\lambda)}\Phi w_t - \Phi w_t)$ by comparing (2.12) - (2.16). Then,

$$(\nabla f(w_t))^\top \mathbb{E}[h_t(w_t)|\mathcal{F}_t] = (w_t - w_\pi)^\top (Aw_t + b + C(w_t))$$
$$= (w_t - w_\pi)^\top \Phi^\top D(\widetilde{T}_\pi^{(\lambda)}\Phi w_t - \Phi w_t) < 0,$$

where the last inequality is by using Lemma 9 from [64] with the fact that, from Theorem 1, $\Pi\widetilde{T}_\pi^{(\lambda)}$ is a contraction. Since $(\nabla f(w_t))^\top \mathbb{E}[h_t|\mathcal{F}_t]$ is strictly negative, we can find a positive scalar c_2 that

satisfies (A.15).

Verifying (A.16)*:* We can write $e_t = \Delta_t^1 w_t + \Delta_t^2 + \Delta_t^3(w_t)$, where,

$$\Delta_t^1 = B_t^{-1}A_t - B^{-1}A,\ \Delta_t^2 = B_t^{-1}b_t - B^{-1}b,\ \Delta_t^3(w_t) = B_t^{-1}C_t(w_t) - B^{-1}C(w_t).$$

From Proposition 2.1 of [61], we have

$$\|\mathbb{E}[\Delta_t^1]\| \leq \frac{\bar{c}_1}{\sqrt{t+1}},\ \ \|\mathbb{E}[\Delta_t^2]\| \leq \frac{\bar{c}_2}{\sqrt{t+1}}\ \ \forall\, t. \tag{A.19}$$

So, we will now bound $\mathbb{E}[\Delta_t^3(w_t)|\mathcal{F}_t]$.

$$\begin{aligned}
&\|\mathbb{E}[\Delta_t^3(w_t)|\mathcal{F}_t]\| = \|\mathbb{E}[B_t^{-1}C_t(w_t) - B^{-1}C(w_t)|\mathcal{F}_t]\| \\
&= \|\mathbb{E}[B_t^{-1}C_t(w_t) - B^{-1}C_t(w_t) + B^{-1}C_t(w_t) - B^{-1}C(w_t)|\mathcal{F}_t]\| \\
&\leq \mathbb{E}[\|(B_t^{-1} - B^{-1})\|\ \|C_t(w_t)\| \mid \mathcal{F}_t] + \|B^{-1}\|\ \mathbb{E}[\|C_t(w_t) - C(w_t)\| \mid \mathcal{F}_t].
\end{aligned} \tag{A.20}$$

First consider $\|C_t(w_t)\|$. We can bound

$$\|z_\tau\| \leq \sum_{m=0}^{\tau}(\alpha\lambda)^{\tau-m}\|\varphi(s_m)\| \overset{(a)}{\leq} \bar{c}_{31} \tag{A.21}$$

$$\begin{aligned}
|\sigma_{\widehat{\mathcal{U}}_{s_\tau,\pi(s_\tau)}}(\Phi w_t)| &= |\inf_{u\in\widehat{\mathcal{U}}_{s_\tau,\pi(s_\tau)}} u^\top(\Phi w_t)| \leq \sup_{u\in\widehat{\mathcal{U}}_{s_\tau,\pi(s_\tau)}} |u^\top(\Phi w_t)| \\
&\leq \sup_{u\in\widehat{\mathcal{U}}_{s_\tau,\pi(s_\tau)}} \|u\|\ \|\Phi w_t\| \overset{(b)}{\leq} \bar{c}_{32}\|w_t\|
\end{aligned} \tag{A.22}$$

where $\bar{c}_{31}$ and $\bar{c}_{32}$ are finite positive scalars. For (a), we used the fact that $\sup_s \|\varphi(s)\| < \infty$. For (b), we used the fact that $\sup_{u\in\widehat{\mathcal{U}}_{s_\tau,\pi(s_\tau)}} \|u\| < \infty$ since $\widehat{\mathcal{U}}_{s,\pi(s)}$ is a finite set for any $s \in \mathcal{S}$ and $\|\Phi w_t\| \leq c'\|w_t\|$ for some finite positive scalar c'. Using (A.21) and (A.22),

$$\|C_t(w_t)\| \leq \frac{\alpha}{(t+1)}\sum_{\tau=0}^{t}\|z_\tau\sigma_{\widehat{\mathcal{U}}_{s_\tau,\pi(s_\tau)}}(\Phi w_t)\| \leq \frac{\alpha}{(t+1)}\sum_{\tau=0}^{t}\|z_\tau\|\ \ |\sigma_{\widehat{\mathcal{U}}_{s_\tau,\pi(s_\tau)}}(\Phi w_t)| \leq \bar{c}_{33}\|w_t\|. \tag{A.23}$$

From Lemma 4.3 of [61], we have $\mathbb{E}[\|(B_t^{-1} - B^{-1})\|] \leq \bar{c}_{34}/\sqrt{(t+1)}$. Using this and (A.23), we get,

$$\mathbb{E}[\|(B_t^{-1} - B^{-1})\| \, \|C_t(w_t)\|] \leq \frac{\bar{c}_{35}}{\sqrt{t+1}} \, \|w_t\|. \tag{A.24}$$

For bounding $\mathbb{E}[\|C_t(w_t) - C(w_t)\|]$ in (A.20), we define V_t and n_t as,

$$V_t = \frac{\alpha}{(t+1)} \sum_{m=0}^{t} \varphi(s_m) \sum_{\tau=t+1}^{\infty} \left[(\alpha\lambda P_\pi^o)^{\tau-m} \sigma_{\widehat{\mathcal{U}}_\pi}(\Phi w_t)\right](s_m), \;\; n_t(s) = \sum_{\tau=0}^{t} \mathbb{I}\{s_\tau = s\}.$$

Note that $n_t(s)$ is the number of visits to state s until time t. Then,

$$\begin{aligned}
\mathbb{E}[C_t(w_t)|\mathcal{F}_t] &= \mathbb{E}\left[\frac{\alpha}{(t+1)} \sum_{\tau=0}^{t} \sum_{m=0}^{\tau} (\alpha\lambda)^{\tau-m} \varphi(s_m) \sigma_{\widehat{\mathcal{U}}_{s_\tau,\pi(s_\tau)}}(\Phi w_t) | \mathcal{F}_t\right] \\
&= \mathbb{E}\left[\frac{\alpha}{(t+1)} \sum_{\tau=0}^{t} \sum_{m=0}^{\tau} (\alpha\lambda)^{\tau-m} \varphi(s_m) \mathbb{E}\left[\sigma_{\widehat{\mathcal{U}}_{s_\tau,\pi(s_\tau)}}(\Phi w_t) | \mathcal{F}_m\right] | \mathcal{F}_t\right] \\
&\overset{(a)}{=} \mathbb{E}\left[\frac{\alpha}{(t+1)} \sum_{\tau=0}^{t} \sum_{m=0}^{\tau} (\alpha\lambda)^{\tau-m} \varphi(s_m) ((P_\pi^o)^{\tau-m} \sigma_{\widehat{\mathcal{U}}_\pi}(\Phi w_t))(s_m) | \mathcal{F}_t\right] \\
&\overset{(b)}{=} \mathbb{E}\left[\frac{\alpha}{(t+1)} \sum_{m=0}^{t} \varphi(s_m) \sum_{\tau=m}^{t} ((\alpha\lambda P_\pi^o)^{\tau-m} \sigma_{\widehat{\mathcal{U}}_\pi}(\Phi w_t))(s_m) | \mathcal{F}_t\right] \\
&= \mathbb{E}\left[\frac{\alpha}{(t+1)} \sum_{m=0}^{t} \varphi(s_m) \sum_{\tau=m}^{\infty} ((\alpha\lambda P_\pi^o)^{\tau-m} \sigma_{\widehat{\mathcal{U}}_\pi}(\Phi w_t))(s_m) | \mathcal{F}_t\right] - \mathbb{E}[V_t|\mathcal{F}_t] \\
&\overset{(c)}{=} \mathbb{E}\left[\frac{\alpha}{(t+1)} \sum_{s\in\mathcal{S}} n_t(s) \varphi(s) \sum_{i=0}^{\infty} ((\alpha\lambda P_\pi^o)^{i} \sigma_{\widehat{\mathcal{U}}_\pi}(\Phi w_t))(s) | \mathcal{F}_t\right] - \mathbb{E}[V_t|\mathcal{F}_t],
\end{aligned} \tag{A.25}$$

where (a) follows since the transition probability matrix governing along policy π is P_π^o, (b) by exchanging the order of summation, and (c) by using the definition of $n_t(s)$. Note that,

$$\begin{aligned}
\mathbb{E}[C(w_t)|\mathcal{F}_t] &= \mathbb{E}[\alpha\Phi^\top D \sum_{i=0}^{\infty} (\alpha\lambda P_\pi^o)^i \sigma_{\widehat{\mathcal{U}}_\pi}(\Phi w_t) | \mathcal{F}_t] \\
&= \mathbb{E}[\alpha \sum_{s\in\mathcal{S}} d_s \varphi(s) \sum_{i=0}^{\infty} ((\alpha\lambda P_\pi^o)^i \sigma_{\widehat{\mathcal{U}}_\pi}(\Phi w_t))(s) | \mathcal{F}_t].
\end{aligned} \tag{A.26}$$

So, using (A.26) and (A.25),

$$\begin{aligned}
&\|\mathbb{E}\left[(C_t(w_t) - C(w_t))\,|\mathcal{F}_t\right]\| \\
&= \left\| \alpha \sum_{s\in\mathcal{S}} \left(\frac{\mathbb{E}[n_t(s)]}{t+1} - d_s \right) \varphi(s) \sum_{i=0}^{\infty} ((\alpha\lambda P_\pi^o)^i \sigma_{\widehat{\mathcal{U}}_\pi}(\Phi w_t))(s) - \mathbb{E}[V_t|\mathcal{F}_t] \right\| \\
&\overset{(d)}{\leq} \bar{c}_{36} \sum_{s\in\mathcal{S}} \left| \left(\frac{\mathbb{E}[n_t(s)]}{t+1} - d_s \right) \right| \|w_t\| + \mathbb{E}[\|V_t\|\,|\mathcal{F}_t] \overset{(e)}{\leq} \frac{\bar{c}_{37}}{t+1}\|w_t\| + \frac{\bar{c}_{38}}{t+1}\|w_t\|
\end{aligned} \tag{A.27}$$

For getting (d), note that $\|\varphi(s)\sum_{i=0}^{\infty}((\alpha\lambda P_\pi^o)^i \sigma_{\widehat{\mathcal{U}}_\pi}(\Phi w_t))(s)\| \leq c'\|w_t\|$ for some positive constant c', using (A.22) and the fact that the summation is bounded due to the discounted factor $(\alpha\lambda)$. For getting (e), we use the result from Lemma 4.2 [61] that $\left| \frac{\mathbb{E}[n_t(s)]}{t+1} - d_s \right| \leq \frac{c'}{t+1}$ for some positive number c'. Also, it is straightforward to show that $\mathbb{E}[\|V_t\|\,|\mathcal{F}_t] \leq c' \frac{\|w_t\|}{t+1}$ for some positive number c' using (A.22) and the fact that the summation is bounded due to the discounted factor $(\alpha\lambda)$.

Using (A.24) and (A.27) in (A.20), we get

$$\|\mathbb{E}[\Delta_t^3(w_t)]\| \leq \frac{\bar{c}_3}{\sqrt{t+1}}\|w_t\|. \tag{A.28}$$

Notice that

$$\begin{aligned}
\|w_t\| \leq \|w_t - w_\pi\| + \|w_\pi\| &\leq \|(\Phi^\top D\Phi)^{-1}\|\,\|(\Phi^\top D\Phi)(w_t - w_\pi)\| + \|w_\pi\| \\
&\leq \bar{c}_{39}(1 + \|\nabla f(w_t)\|).
\end{aligned} \tag{A.29}$$

Now,

$$\begin{aligned}
\|\mathbb{E}[e_t|\mathcal{F}_t]\| &= \|\mathbb{E}[\Delta_t^1]\|\,\|w_t\| + \|\mathbb{E}[\Delta_t^2]\| + \|\mathbb{E}[\Delta_t^3(w_t)|\mathcal{F}_t]\| \\
&\overset{(f)}{\leq} \frac{\bar{c}_1}{\sqrt{t+1}}\|w_t\| + \frac{\bar{c}_2}{\sqrt{t+1}} + \frac{\bar{c}_3}{\sqrt{t+1}}\|w_t\| \overset{(g)}{\leq} \frac{c_2}{\sqrt{t+1}}(1 + \|\nabla f(w_t)\|)
\end{aligned} \tag{A.30}$$

where (f) is by using (A.19) and (A.28) and (g) is by using (A.29). This completes the verification of the condition (A.16).

Verifying (A.17)*:* From definition, we have $h_t(w_t) + e_t = B_t^{-1}(A_t w_t + b_t + C_t(w_t))$, for all t. Using similar steps as before, it is straightforward to show that $\|B_t\| \leq c_{41}, |A_t| \leq c_{42}, |b_t| \leq c_{43}$ for some positive scalars c_{41}, c_{42}, c_{43}. From (A.23) we have $\|c_t(w_t)\| \leq \bar{c}_{33}\|w_t\|$. Combining these, we get

$$\|h_t(w_t) + e_t\| \leq c_{44}(1 + \|w_t\|). \tag{A.31}$$

Now, from (A.29) and (A.14), $\|h_t(w_t) + e_t\|^2 \leq c_{45}(1 + \|\nabla f(w_t)\|^2)$ for all t. So, finally we have,

$$\mathbb{E}[\|h_t(w_t) + e_t\|^2 | \mathcal{F}_t] = \mathbb{E}[\|h_t(w_t) + e_t\|^2 | w_t] \leq c_3(1 + \|\nabla f(w_t)\|^2) \ \ \forall\, t,$$

showing that (A.17) is satisfied.

Verifying (A.18)*:* From (A.30), $\varepsilon_t = 1/\sqrt{(t+1)}$. So, this condition is satisfied. ‘ So, all the assumption of Proposition 11 are satisfied. Hence the result of that Proposition is true. In particular, $\nabla f(w_t) = \Phi^\top D \Phi(w_t - w_\pi)$ converges to 0. Since $\Phi^\top D \Phi$ is positive definite, this implies that $w_t \to w_\pi$. □

A.3 Proof of the Results in Section 2.4

Let $\bar{V}_k = \Phi w_{\pi_k}$ and V^* be the optimal robust value function. Define

$$e_k = V_{\pi_k} - \bar{V}_k, \ \ l_k = V^* - V_{\pi_k}, \ \ g_k = V_{\pi_{k+1}} - V_{\pi_k}. \tag{A.32}$$

Interpretations of these expressions: Since the robust value function in the k^{th} iteration $\bar{V}_k$ is used as a surrogate for the robust value function V_{π_k}, e_k quantifies the *approximation error*. g_k signifies the *gain* of value functions between iterations k and $k+1$. Finally, l_k encapsulates the *loss* in the value function because of using policy π_k instead of the optimal policy.

Let $|x|$ denote element-wise absolute values of vector $x \in \mathbb{R}^{|\mathcal{S}|}$. We first prove the following result. This parallels to the result for the nonrobust setting in [67].

Lemma 14. *We have*

$$|l_{k+1}| \leq c(\alpha,\beta,0,\lambda)\bar{H}_*(|l_k|+|e_k|) + c(\alpha,\beta,0,\lambda)\bar{H}_k(|g_k|+|e_k|) \;\; \textit{and} \tag{A.33}$$

$$|g_k| \leq c(\alpha,\beta,0,\lambda)(I - c(\alpha,\beta,0,\lambda)\bar{H}_{k+1})^{-1}(\bar{H}_{k+1}+\bar{H}_k)|e_k| \tag{A.34}$$

where the stochastic matrices $\bar{H}_*$, $\bar{H}_k$, and $\bar{H}_{k+1}$ are defined in (A.37)-(A.38).

Proof. As before, denote the operator $\widetilde{T}_\pi^{(\lambda)}$ as $\bar{T}_\pi^{(\lambda)}$ when $\widehat{\mathcal{U}}_\pi = \mathcal{U}_\pi$. Now, similar to (A.9), for any policy π and V_1, V_2, we get that

$$\begin{aligned}
&|\bar{T}_\pi^{(\lambda)}(V_1) - \bar{T}_\pi^{(\lambda)}(V_2)| \\
&= |(1-\lambda)\sum_{m=0}^{\infty}\lambda^m[\alpha\sum_{t=0}^{m}(\alpha P_\pi^o)^t\,(\sigma_{\mathcal{U}_\pi}(V_1) - \sigma_{\mathcal{U}_\pi}(V_2)) + (\alpha P_\pi^o)^{m+1}(V_1 - V_2)]| \\
&\leq (1-\lambda)\sum_{m=0}^{\infty}\lambda^m[\alpha\sum_{t=0}^{m}\,|(\alpha P_\pi^o)^t|\sigma_{\mathcal{U}_\pi}(V_1) - \sigma_{\mathcal{U}_\pi}(V_2)|\,| + |(\alpha P_\pi^o)^{m+1}|V_1 - V_2||] \\
&\overset{(a)}{\leq} (1-\lambda)\sum_{m=0}^{\infty}\lambda^m[\alpha\sum_{t=0}^{m}(\beta P_{\pi_e}^o)^t|\sigma_{\mathcal{U}_\pi}(V_1) - \sigma_{\mathcal{U}_\pi}(V_2)| + |(\beta P_{\pi_e}^o)^{m+1}|V_1 - V_2||] \\
&\overset{(b)}{\leq} (1-\lambda)\sum_{m=0}^{\infty}\lambda^m[\beta\sum_{t=0}^{m}(\beta P_{\pi_e}^o)^t|V_1 - V_2| + |(\beta P_{\pi_e}^o)^{m+1}|V_1 - V_2||]
\end{aligned} \tag{A.35}$$

where (a) follows from Assumption 2 and π_e (dependent on π) being the exploration policy. (b) follows from (A.5) in Lemma 13.

Recall that the optimal robust value function V^* and the optimal robust policy π^* satisfy the equation $\bar{T}_{\pi^*}^{(\lambda)}V^* = V^*$. Using this,

$$\begin{aligned}
l_{k+1} &= V^* - V_{\pi_{k+1}} = \bar{T}_{\pi^*}^{(\lambda)}V^* - \bar{T}_{\pi_{k+1}}^{(\lambda)}V_{\pi_{k+1}} \\
&= (\bar{T}_{\pi^*}^{(\lambda)}V^* - \bar{T}_{\pi^*}^{(\lambda)}V_{\pi_k}) + (\bar{T}_{\pi^*}^{(\lambda)}V_{\pi_k} - \bar{T}_{\pi^*}^{(\lambda)}\bar{V}_k) + (\bar{T}_{\pi^*}^{(\lambda)}\bar{V}_k - \bar{T}_{\pi_{k+1}}^{(\lambda)}\bar{V}_k) \\
&\qquad + (\bar{T}_{\pi_{k+1}}^{(\lambda)}\bar{V}_k - \bar{T}_{\pi_{k+1}}^{(\lambda)}V_{\pi_k}) + (\bar{T}_{\pi_{k+1}}^{(\lambda)}V_{\pi_k} - \bar{T}_{\pi_{k+1}}^{(\lambda)}V_{\pi_{k+1}}) \\
&\overset{(c)}{\leq} (\bar{T}_{\pi^*}^{(\lambda)}V^* - \bar{T}_{\pi^*}^{(\lambda)}V_{\pi_k}) + (\bar{T}_{\pi^*}^{(\lambda)}V_{\pi_k} - \bar{T}_{\pi^*}^{(\lambda)}\bar{V}_k) \\
&\qquad + (\bar{T}_{\pi_{k+1}}^{(\lambda)}\bar{V}_k - \bar{T}_{\pi_{k+1}}^{(\lambda)}V_{\pi_k}) + (\bar{T}_{\pi_{k+1}}^{(\lambda)}V_{\pi_k} - \bar{T}_{\pi_{k+1}}^{(\lambda)}V_{\pi_{k+1}})
\end{aligned}$$

$$
\begin{aligned}
&\overset{(d)}{\leq} (1-\lambda)\sum_{m=0}^{\infty}\lambda^m[\beta\sum_{t=0}^{m}(\beta P^o_{\pi^*})^t(|l_k|+|e_k|)+(\beta P^o_{\pi^*})^{m+1}(|l_k|+|e_k|)] \\
&\qquad + (1-\lambda)\sum_{m=0}^{\infty}\lambda^m[\beta\sum_{t=0}^{m}(\beta P^o_{\pi_{k+1}})^t(|g_k|+|e_k|)+(\beta P^o_{\pi_{k+1}})^{m+1}(|g_k|+|e_k|)] \\
&\overset{(e)}{=} c(\alpha,\beta,0,\lambda)\bar{H}_*(|l_k|+|e_k|)+c(\alpha,\beta,0,\lambda)\bar{H}_{k+1}(|g_k|+|e_k|) \qquad \text{(A.36)}
\end{aligned}
$$

Here (c) follows because π_{k+1} is the greedy policy w.r.t. $\bar{V}_k$ and hence $\bar{T}^{(\lambda)}_{\pi^*}\bar{V}_k - \bar{T}^{(\lambda)}_{\pi_{k+1}}\bar{V}_k \leq 0$. (d) follows from (A.35) noting (i) in Assumption 3. Finally, (e) follows by taking

$$
\bar{H}_* = \frac{1}{c(\alpha,\beta,0,\lambda)}(1-\lambda)\sum_{m=0}^{\infty}\lambda^m[\beta\sum_{t=0}^{m}(\beta P^o_{\pi^*})^t+(\beta P^o_{\pi^*})^{m+1}], \text{ and} \qquad \text{(A.37)}
$$

$$
\bar{H}_j = \frac{1}{c(\alpha,\beta,0,\lambda)}(1-\lambda)\sum_{m=0}^{\infty}\lambda^m[\beta\sum_{t=0}^{m}(\beta P^o_{\pi_j})^t+(\beta P^o_{\pi_j})^{m+1}], \text{ for } j\geq 1. \qquad \text{(A.38)}
$$

Note that, matrices $\bar{H}_*$ and $\bar{H}_j$ are stochastic matrices. This follows easily by verifying $\bar{H}_*\mathbf{1}=\mathbf{1}$, $\bar{H}_j\mathbf{1}=\mathbf{1}$ using algebra analysis as in (A.10).

The same argument can be repeated to get

$$
-l_{k+1}\leq c(\alpha,\beta,0,\lambda)\bar{H}_*(|l_k|+|e_k|)+c(\alpha,\beta,0,\lambda)\bar{H}_{k+1}(|g_k|+|e_k|).
$$

Combining, we get

$$
|l_{k+1}|\leq c(\alpha,\beta,0,\lambda)\bar{H}_*(|l_k|+|e_k|)+c(\alpha,\beta,0,\lambda)\bar{H}_{k+1}(|g_k|+|e_k|).
$$

Now,

$$
\begin{aligned}
g_k &= \bar{T}^{(\lambda)}_{\pi_{k+1}}V_{\pi_{k+1}} - \bar{T}^{(\lambda)}_{\pi_k}V_{\pi_k} \\
&= \bar{T}^{(\lambda)}_{\pi_{k+1}}V_{\pi_{k+1}} - \bar{T}^{(\lambda)}_{\pi_{k+1}}V_{\pi_k} + \bar{T}^{(\lambda)}_{\pi_{k+1}}V_{\pi_k} - \bar{T}^{(\lambda)}_{\pi_{k+1}}\bar{V}_k + (\bar{T}^{(\lambda)}_{\pi_{k+1}}\bar{V}_k - \bar{T}^{(\lambda)}_{\pi_k}\bar{V}_k) + \bar{T}^{(\lambda)}_{\pi_k}\bar{V}_k - \bar{T}^{(\lambda)}_{\pi_k}V_{\pi_k} \\
&\geq \bar{T}^{(\lambda)}_{\pi_{k+1}}V_{\pi_{k+1}} - \bar{T}^{(\lambda)}_{\pi_{k+1}}V_{\pi_k} + \bar{T}^{(\lambda)}_{\pi_{k+1}}V_{\pi_k} - \bar{T}^{(\lambda)}_{\pi_{k+1}}\bar{V}_k + \bar{T}^{(\lambda)}_{\pi_k}\bar{V}_k - \bar{T}^{(\lambda)}_{\pi_k}V_{\pi_k}
\end{aligned}
$$

where the last inequality follows because π_{k+1} is the greedy policy w.r.t. $\bar{V}_k$ and hence $(\bar{T}^{(\lambda)}_{\pi_{k+1}}\bar{V}_k - \bar{T}^{(\lambda)}_{\pi_k}\bar{V}_k) \geq 0$. From (A.35), noting (i) in Assumption 3, we have

$$\begin{aligned}
-g_k \leq (1-\lambda)\sum_{m=0}^{\infty}\lambda^m[\beta\sum_{t=0}^{m}(\beta P^o_{\pi_{k+1}})^t(|g_k|+|e_k|) + (\beta P^o_{\pi_{k+1}})^{m+1}(|g_k|+|e_k|)] \\
+ (1-\lambda)\sum_{m=0}^{\infty}\lambda^m[\beta\sum_{t=0}^{m}(\beta P^o_{\pi_k})^t(|e_k|) + (\beta P^o_{\pi_k})^{m+1}(|e_k|)] \\
\leq c(\alpha,\beta,0,\lambda)\bar{H}_{k+1}(|g_k|+|e_k|) + c(\alpha,\beta,0,\lambda)\bar{H}_k(|e_k|).
\end{aligned}$$

Repeating the above argument for $-g_k = \bar{T}^{(\lambda)}_{\pi_k}V_{\pi_k} - \bar{T}^{(\lambda)}_{\pi_{k+1}}V_{\pi_{k+1}}$, we get

$$|g_k| \leq c(\alpha,\beta,0,\lambda)\bar{H}_{k+1}(|g_k|+|e_k|) + c(\alpha,\beta,0,\lambda)\bar{H}_k(|e_k|).$$

So, $|g_k| \leq c(\alpha,\beta,0,\lambda)(I - c(\alpha,\beta,0,\lambda)\bar{H}_{k+1})^{-1}(\bar{H}_{k+1}+\bar{H}_k)|e_k|$. Thus, proving the lemma. □

Proof of Theorem 3. From the Lemma 14, taking $\limsup$ on both sides of (A.33) we have

$$\begin{aligned}
\limsup_{k\to\infty}|l_k| &\leq \limsup_{k\to\infty}\ c(\alpha,\beta,0,\lambda)(I - c(\alpha,\beta,0,\lambda)\bar{H}_*)^{-1}(\bar{H}_*+\bar{H}_k)|e_k| \\
&\quad + c(\alpha,\beta,0,\lambda)(I - c(\alpha,\beta,0,\lambda)\bar{H}_*)^{-1}\bar{H}_k|g_k| \\
&\overset{(a)}{\leq} \limsup_{k\to\infty}\ c(\alpha,\beta,0,\lambda)(I - c(\alpha,\beta,0,\lambda)\bar{H}_*)^{-1}(\bar{H}_*+\bar{H}_k)|e_k| \\
&\quad + c^2(\alpha,\beta,0,\lambda)(I - c(\alpha,\beta,0,\lambda)\bar{H}_*)^{-1}\bar{H}_k(I - c(\alpha,\beta,0,\lambda)\bar{H}_{k+1})^{-1}(\bar{H}_{k+1}+\bar{H}_k)|e_k| \\
&\overset{(b)}{=} \frac{2c(\alpha,\beta,0,\lambda)}{(1-c(\alpha,\beta,0,\lambda))^2}\limsup_{k\to\infty}\ H_k|e_k| \qquad \text{(A.39)}
\end{aligned}$$

where (a) follows from (A.34) in Lemma 14. (b) follows by taking

$$\begin{aligned}
H_k = (1-c(\alpha,\beta,0,\lambda))^2(I - c(\alpha,\beta,0,\lambda)\bar{H}_*)^{-1}\Bigg(\frac{(\bar{H}_*+\bar{H}_k)}{2} \\
+ c(\alpha,\beta,0,\lambda)\bar{H}_k(I - c(\alpha,\beta,0,\lambda)\bar{H}_{k+1})^{-1}\frac{(\bar{H}_{k+1}+\bar{H}_k)}{2}\Bigg). \qquad \text{(A.40)}
\end{aligned}$$

Notice that H_k is a stochastic matrix. To see this, we know that, if $P_i,\ i = \{1, 2, 3, 4\}$ are stochastic matrices and $c < 1$, then $P_1P_2P_3P_4$, $(P_1 + P_2)/2$, and $(1 - c)(I - cP_1)^{-1}$ are valid stochastic matrices as well. Now, it is easy to verify that $H_k\mathbf{1} = \mathbf{1}$. Then, $\mu_k = \mu H_k$ is a valid probability distribution.

Let x^2 denote element-wise squares of vector $x \in \mathbb{R}^{|\mathcal{S}|}$ and also let $\|x\|_\mu^2 = \int_{s\in\mathcal{S}} |x(s)|^2 d\mu(s) = \mu|x|^2$. Now, from (A.39) we have

$$\begin{aligned}
\limsup_{k\to\infty} \|l_k\|_\mu^2 = \limsup_{k\to\infty} \mu|l_k|^2 &\le \frac{4c^2(\alpha,\beta,0,\lambda)}{(1-c(\alpha,\beta,0,\lambda))^4} \limsup_{k\to\infty}\ \mu[H_k|e_k|]^2 \\
&\overset{(c)}{\le} \frac{4c^2(\alpha,\beta,0,\lambda)}{(1-c(\alpha,\beta,0,\lambda))^4} \limsup_{k\to\infty}\ \mu H_k|e_k|^2 \\
&= \frac{4c^2(\alpha,\beta,0,\lambda)}{(1-c(\alpha,\beta,0,\lambda))^4} \limsup_{k\to\infty}\ \mu_k|e_k|^2 = \frac{4c^2(\alpha,\beta,0,\lambda)}{(1-c(\alpha,\beta,0,\lambda))^4} \limsup_{k\to\infty}\ \|e_k\|_{\mu_k}^2 \\
&\overset{(d)}{\le} \frac{4C_1C_2c^2(\alpha,\beta,0,\lambda)}{(1-c(\alpha,\beta,0,\lambda))^4} \limsup_{k\to\infty}\ \|e_k\|_{d\pi_k}^2.
\end{aligned}$$

Here (c) follows from Jensen's inequality. To see this, let

$$H_k = \begin{pmatrix} -q_1- \\ -q_2- \\ \vdots \\ -q_{|\mathcal{S}|}- \end{pmatrix} \text{ where } q_i \in \mathbb{R}^{|\mathcal{S}|},\ \text{for all } i,\ \text{are probability vectors.}$$

For any $x \in \mathbb{R}^{|\mathcal{S}|}$, let $x(j)$ denote the j^{th} coordinate value in x. Now, for each $i \in \{1, 2, \ldots, |\mathcal{S}|\}$, define $|\mathcal{S}|$-discrete valued random variable X_i such that it takes value $|e_k|(j)$ with probability $q_i(j)$ for all $j \in \{1, 2, \ldots, |\mathcal{S}|\}$. Thus, from Jensen's inequality, we have

$$[H_k|e_k|]^2 = ((\mathbb{E}[X_1])^2, (\mathbb{E}[X_2])^2, \cdots, (\mathbb{E}[X_\mathcal{S}])^2)^\top \le (\mathbb{E}[X_1^2], \mathbb{E}[X_2^2], \cdots, \mathbb{E}[X_\mathcal{S}^2])^\top = H_k|e_k|^2. \tag{A.41}$$

(d) follows by noting that for any $x \in \mathbb{R}^{|\mathcal{S}|}$, from (iv) in Assumption 3, we have $\|x\|_{\mu_k}^2 \le C_1\|x\|_{\bar{\mu}}^2 \le C_1C_2\|x\|_{d\pi_k}^2$ for all k. Thus proving (2.26) of this theorem.

Now, since $\rho = 0$, (2.27) of this theorem directly follows from (iii) in Assumption 3 and (2.11) in Theorem 1. This completes the proof of this theorem. □

Now we make an alternative assumption to Assumption 3.(iv) to get a guarantee for any K^{th} iteration of the RLSPI algorithm.

Assumption 10. *For an arbitrary sequence of stationary policies $\{\pi_i\}_{i\geq 1}$, let some probability distributions μ and $\bar{\mu}$ satisfy*

$$C_{\mu,\bar{\mu}} = (1 - c(\alpha, \beta, 0, \lambda)) \sum_{m\geq 1} (c(\alpha, \beta, 0, \lambda))^m c_{\mu,\bar{\mu}}(m) < \infty, \tag{A.42}$$

where for any given $m \geq 1$ the coefficients are defined as

$$c_{\mu,\bar{\mu}}(m) = \sup_{\pi_1,\ldots,\pi_m} \left\| \frac{\mathrm{d}(\mu \bar{H}_1 \bar{H}_2 \ldots \bar{H}_m)}{\mathrm{d}\bar{\mu}} \right\|_\infty, \tag{A.43}$$

and $\bar{H}_k$ is a stochastic matrix that depends on $P^o_{\pi_k}$ for any $1 \leq k \leq m$. Also assume that $d_{\pi_k} \geq \bar{\mu}/C$ for all k.

We note that $c_{\mu,\bar{\mu}}(m)$ can potentially diverge to ∞, but $C_{\mu,\bar{\mu}}$ is finite as long as $(c(\alpha, \beta, 0, \lambda))^m$ converges to 0 at a faster rate. Assumption 10 being similar as in the non-robust setting, we refer the reader to [89, 69] for its detailed interpretation.

Here is a result that provides a guarantee for the performance of the policy learned in K^{th} iteration of the RLSPI algorithm.

Theorem 18. *Let Assumption 2, Assumption 3.(i)-(iii), and Assumption 10 hold. Let the range of reward function r be $(0, R_{\max}]$. Let $\{\pi_k\}_{k=1}^K$ be the sequence of policies generated by the RLSPI algorithm for some $K \geq 1$. 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 $c(\alpha,\beta,0,\lambda)<1$*,*

$$\|V^*-V_{\pi_K}\|_\mu \leq \frac{2\sqrt{2}(c(\alpha,\beta,0,\lambda))^{(K+1)/2}}{(1-c(\alpha,\beta,0,\lambda))^{3/2}}\,R_{\max}+\frac{2\sqrt{2}c(\alpha,\beta,0,\lambda)\sqrt{CC_{\mu,\bar{\mu}}}}{(1-c(\alpha,\beta,0,\lambda))^2}\max_{0\leq k<K}\|V_{\pi_k}-\bar{V}_k\|_{d_{\pi_k}}. \tag{A.44}$$

Moreover, from Theorem 1 and Assumption 3.(iii)*, as* $K\to\infty$ *we have*

$$\|V^*-V_{\pi_K}\|_\mu \leq \frac{2\sqrt{2}c(\alpha,\beta,0,\lambda)\sqrt{CC_{\mu,\bar{\mu}}}}{(1-c(\alpha,\beta,0,\lambda))^3}\,\delta. \tag{A.45}$$

Proof of Theorem 18. Let $\zeta=c(\alpha,\beta,0,\lambda)$. From the Lemma 14, using (A.34) in (A.33) we have

$$\begin{aligned}|l_{k+1}| &\leq \zeta\bar{H}_*|l_k|+\zeta(\bar{H}_*+\bar{H}_k)|e_k|+\zeta\bar{H}_k|g_k|\\ &\leq \zeta\bar{H}_*|l_k|+\zeta(\bar{H}_*+\bar{H}_k)|e_k|+\zeta^2\bar{H}_k(I-\zeta\bar{H}_{k+1})^{-1}(\bar{H}_{k+1}+\bar{H}_k)|e_k|\\ &\overset{(a)}{=} \zeta\bar{H}_*|l_k|+\frac{2\zeta}{1-\zeta}H_k|e_k|,\end{aligned} \tag{A.46}$$

where (a) follows since

$$H_k=(1-\zeta)\left(\frac{(\bar{H}_*+\bar{H}_k)}{2}+\zeta\bar{H}_k(I-\zeta\bar{H}_{k+1})^{-1}\frac{(\bar{H}_{k+1}+\bar{H}_k)}{2}\right). \tag{A.47}$$

Taking $(K-1)$-recursions of (A.46) we get

$$|l_K|\leq(\zeta\bar{H}_*)^K|l_0|+\frac{2\zeta}{1-\zeta}\sum_{k=0}^{K-1}(\zeta\bar{H}_*)^{K-k-1}H_k|e_k|. \tag{A.48}$$

Note that since $\alpha\leq\beta$, we have that $\alpha\leq\zeta$. Now, since the rewards are in $(0,R_{\max}]$, we also have that $|l_0|=|V^*-V_{\pi_0}|\leq\frac{2R_{\max}}{(1-\alpha)}\mathbb{1}\leq\frac{2R_{\max}}{(1-\zeta)}\mathbb{1}$. Thus, bounding (A.48) further we get

$$|l_K|\leq\frac{2\zeta_K\zeta}{(1-\zeta)^2}\left[\alpha_K G_K R_{\max}\mathbb{1}+\sum_{k=0}^{K-1}\alpha_k G_k|e_k|\right], \tag{A.49}$$

where $\zeta_K = \zeta^{K-1}(2-\zeta-\zeta^K) \leq 2$, the positive coefficients αs are

$$\alpha_k = \frac{\zeta^{K-k-1}(1-\zeta)}{\zeta_K}, \text{ for } 0 \leq k \leq K-1, \text{ and } \alpha_K = \frac{\zeta^{K-1}(1-\zeta)}{\zeta_K}, \tag{A.50}$$

and the operators Gs are

$$G_k = (\bar{H}_*)^{K-k-1} H_k, \text{ for } 0 \leq k \leq K-1, \text{ and } G_K = (\bar{H}_*)^K.$$

Note that $\sum_{k=0}^{K} \alpha_k = 1$ and G_k for $0 \leq k \leq K$ are stochastic matrices.

Now, from (A.49) we have

$$\begin{aligned}
\|l_K\|_\mu^2 &\overset{(b)}{\leq} \frac{4\zeta_K^2\zeta^2}{(1-\zeta)^4}\,\mu\left[\alpha_K(G_K R_{\max}\mathbb{1})^2 + \sum_{k=0}^{K-1}\alpha_k(G_k|e_k|)^2\right] \\
&\overset{(c)}{\leq} \frac{4\zeta_K^2\zeta^2}{(1-\zeta)^4}\,\mu\left[\alpha_K G_K R_{\max}^2\mathbb{1} + \sum_{k=0}^{K-1}\alpha_k G_k|e_k|^2\right] \\
&= \frac{4\zeta_K^2\zeta^2}{(1-\zeta)^4}\,\left[\alpha_K R_{\max}^2 + \mu\sum_{k=0}^{K-1}\alpha_k G_k|e_k|^2\right] \\
&\overset{(d)}{\leq} \frac{4\zeta_K^2\zeta^2}{(1-\zeta)^4}\,\left[\alpha_K R_{\max}^2 + (1-\zeta)\sum_{k=0}^{K-1}\alpha_k\sum_{m\geq 0}\zeta^m c_{\mu,\bar{\mu}}(m+K-k)\|e_k\|_{\bar{\mu}}^2\right] \\
&\overset{(e)}{\leq} \frac{4\zeta_K^2\zeta^2}{(1-\zeta)^4}\,\left[\alpha_K R_{\max}^2 + (1-\zeta)\|e\|_{\bar{\mu}}^2\sum_{k=0}^{K-1}\alpha_k\sum_{m\geq 0}\zeta^m c_{\mu,\bar{\mu}}(m+K-k)\right] \\
&\overset{(f)}{=} \frac{4\zeta_K\zeta^2}{(1-\zeta)^4}\,\left[\zeta^{K-1}(1-\zeta)R_{\max}^2 + (1-\zeta)\|e\|_{\bar{\mu}}^2\sum_{k=0}^{K-1}\zeta^{K-k-1}(1-\zeta)\sum_{m\geq 0}\zeta^m c_{\mu,\bar{\mu}}(m+K-k)\right] \\
&\overset{(g)}{\leq} \frac{8\zeta^2}{(1-\zeta)^4}\,\left[\zeta^{K-1}(1-\zeta)R_{\max}^2 + \|e\|_{\bar{\mu}}^2 C_{\mu,\bar{\mu}}\right] \\
&= \frac{8\zeta^{K+1}}{(1-\zeta)^3}\,R_{\max}^2 + \frac{8\zeta^2}{(1-\zeta)^4}\|e\|_{\bar{\mu}}^2 C_{\mu,\bar{\mu}} \overset{(h)}{\leq} \frac{8\zeta^{K+1}}{(1-\zeta)^3}\,R_{\max}^2 + \frac{8\zeta^2 C C_{\mu,\bar{\mu}}}{(1-\zeta)^4}\|e\|_{d_{\pi_k}}^2,
\end{aligned}$$

where (b) follows from Jensen's inequality, i.e., $f(\sum_{k=0}^{K}\alpha_k x_k) \leq \sum_{k=0}^{K}\alpha_k f(x_k)$ for any convex function f. Also, (c) follows from Jensen's inequality, similar to (A.41), with stochastic matrix G_k. (d) follows from the definition of the coefficients $c_{\mu,\bar{\mu}}(m)$ (A.43), (e) follows by taking e

such that $\|e\|_{\bar{\mu}}^2 = \max_{0\leq k\leq K-1} \|e_k\|_{\bar{\mu}}^2$, (f) follows from (A.50), (g) follows since $\zeta_K \leq 2$ and the definition of $C_{\mu,\bar{\mu}}$ (A.42), and (h) follows by noting that for any $x \in \mathbb{R}^{|\mathcal{S}|}$, from Assumption 10, we have $\|x\|_{\bar{\mu}}^2 \leq C\|x\|_{d_{\pi_k}}^2$ for all k. Using the fact that $a^2 + b^2 \leq (a+b)^2$ for $a \geq 0, b \geq 0$ completes the proof of (A.44) in this theorem.

Now, since $\rho = 0$, (A.45) of this theorem directly follows from (iii) in Assumption 3 and (2.11) in Theorem 1. This completes the proof of this theorem. □

A.4 Experiments

In all the experiments reported, we use a spherical uncertainty set $\{x : \|x\|_2 \leq r\}$ where r is the radius parameter. For such a set, we can compute a closed form solution for $\sigma_{\widehat{\mathcal{U}}_{s,\pi(s)}}(\Phi w)$ as $\sqrt{rw^\top \Phi^\top \Phi w}$ [40]. Note that, we can precompute $\Phi^\top \Phi$ once and reuse it in every iteration of the RLSPI Algorithm, thus saving the computational overhead.

Chain MDP: We first consider a tabular MDP problem represented in the Figure A.1 for verifying the convergence of RLSPI algorithm. This MDP consists of 10 states depicted by circles here. We have two actions, that is, move left or right. The actions fail to remain in a given direction with probability 0.1, depicted by the red arrows. Thus, with probability 0.9, actions succeed to be in a given direction, depicted by the blue (action left being unchanged) and green (action right being unchanged) arrows. Finally, visiting states of yellow color, that is 0 and 9, are rewarded 1, and visiting other states are rewarded 0.

[42] observes that learning algorithms often attain sub-optimal policies under such MDPs due to the randomization of actions (as depicted by red arrows in the Figure A.1). It is also straightforward that the optimal policy of this MDP is moving left for states 0 through 4 and moving right for states 5 through 9. We train RLSPI algorithm on this MDP with $\alpha = 0.9, \lambda = 0$. We use the space spanned by polynomials, degree up to 2, as the feature space and set $\delta = 0.1$ (error of weights as mentioned in Step 8 of the RLSPI algorithm). We select r as 0.01 times the constant $\|\Phi^\top \Phi\|_F^{-1}$ where $\|\cdot\|_F$ is the Frobenius norm.

Figure A.2 shows how the Q-value functions in RLSPI algorithm training evolve as the iteration progress. From this, we note that RLSPI algorithm is able to find the optimal policy with relatively

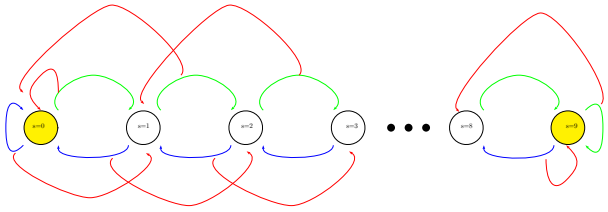

Figure A.1

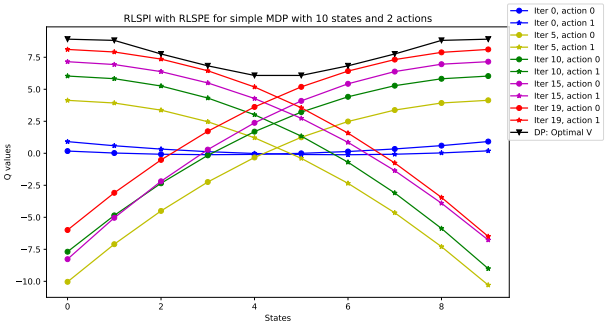

Figure A.2

less number of iterations. From this figure, we also note that the Q-value functions corresponding to the optimal policy in RLSPI algorithm converges to the optimal robust value function.

Examples from OpenAI Gym [71]: We now provide more details for the OpenAI Gym experiments demonstrated in Section 2.5. We use the radial basis functions (RBFs) for the purpose of feature spaces in our experiments. The general expression for RBFs is $\psi(x) = \exp(-\|x-\mu\|^2/\sigma)$ where the RBF parameters μ and σ are chosen before running the experiment. Here x is a concatenation of states and actions when both $\mathcal{S}, \mathcal{A}$ are continuous spaces. In this case, the feature map is simply defined as $\varphi(s,a) = \psi((s,a))$ where (s,a) represents the concatenation operation. While working with experiments whose action space is discrete, we naturally choose $\varphi(s,a)$ to be the vector $(\mathbb{1}(a = 1)\psi(s), \ldots, \mathbb{1}(a = |\mathcal{A}|)\psi(s))^\top$ where $\mathbb{1}(E)$ is the indicator function which produces value 1 if the event E is true, and 0 otherwise. After a few trials, we observed that using few (typically 3-4) uniformly spaced RBFs in each dimension of x, here x is as described before, with approximate overlap percentage of 2.5 works suitably

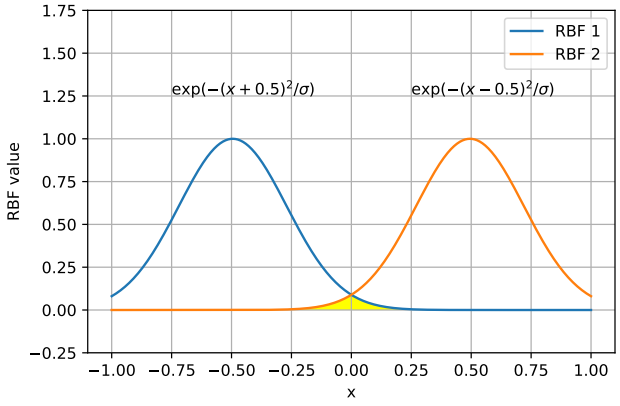

Figure A.3: We have discretized x with 10^4 points here. The overlap percentage (ratio of the area in the yellow region to the area of either RBF 1 or RBF 2) is 2.88.

for getting the desired results shown here and in Section 2.5. Figure A.3 illustrates this for the case of using two($= n$) uniformly spaced RBFs with one dimensional x variable. For this illustration, we have the low(l) and high(h) values of x to be -0.5 and 0.5 respectively. Thus, the centers (i.e., parameter μ) of the two uniformly spaced RBFs in Figure A.3 are -0.5 and 0.5. We select the parameter σ as $(h-l)^2/n^3 = 0.125$. We execute this idea on the OpenAI Gym environments. We also experiment on FrozenLake8x8 OpenAI Gym environment, for which we use the tabular feature space since the state space is discrete.

A short description of the OpenAI Gym tasks **CartPole, MountainCar, Acrobot, and FrozenLake8x8** we used are as follows.

CartPole: By a hinge, a pole is attached to a cart, which moves along a one-dimensional path. The motion of the cart is controllable, which is either to move it left or right. The pole starts upright, and the goal is to prevent it from falling over. A reward of $+1$ is provided for every time-step that the pole remains upright. CartPole consists of a 4-dimension continuous state space with 2 discrete actions.

MountainCar: A car is placed in the valley and there exists a flag on top of the hill. The goal is to reach the flag. The control signal is the acceleration and deceleration in continuous domain. A reward of 0 is provided if the car reaches goal, otherwise it is provided -1. MountainCar consists of a 3-dimension $\mathcal{S} \times \mathcal{A}$ continuous space.

Acrobot: Two poles attached to each other by a free moving joint and one of the poles is attached to a hinge on a wall. Initially, the poles are hanging downwards. An action, positive and negative torques can be applied to the movable joint. A reward of -1 is provided every time-step until the end of the lower pole reaches a given height, at termination 0 reward is provided. The goal is to maximize the reward gathered. Acrobot consists of a 6-dimension continuous state space with 3 discrete actions.

FrozenLake8x8: A grid of size 8×8 consists of some tiles which lead to the agent falling into the

water. The agent is rewarded 1 after reaching a goal tile without falling and rewarded 0 in every other timestep.

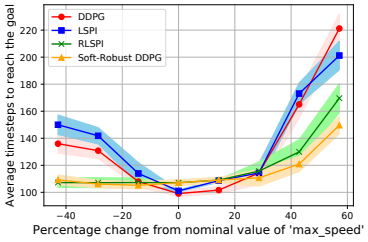

Figure A.4

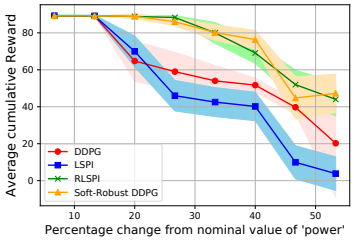

Figure A.5

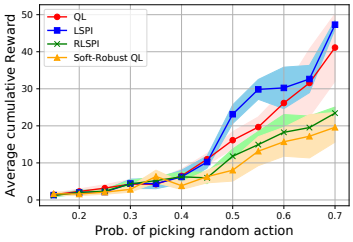

Figure A.6

In Section 2.5, we provided performance evaluation curves in Figures 2.1-2.6. Here, we provide more results.

Figures A.4 shows the average time steps to reach the goal in MountainCar environment as we change the parameter *max_speed*. The nominal value of this parameter is 0.07. As the parameter deviates from the nominal value, the performance of the policy obtained by the LSPI algorithm degrades quickly whereas the performance of the policy obtained by the RLSPI algorithm is fairly robust. Figure A.5 shows the average cumulative reward on the MountainCar environment as we change the parameter *power*. The nominal value of this parameter is 15×10^{-4}. We again note that the RLSPI algorithm showcases robust performance. Figure A.6 shows the ratio of average time to reach the goal and the number of trajectories which actually reach the goal on the FrozenLake8x8 environment against probability of picking a random action. Note that for large values of this probability all algorithms take more time to reach the goal or often fall into the water. Here again, RLSPI shows robust performance. Intuitively, perturbation in the parameters (like the action space, CartPole's *force_mag, gravity, length*, MountainCar's *max_speed, power*) of the environment is captured by the uncertainty set in the RMDP framework. Thus we see good performances of the robust algorithms like our RLSPI algorithm, Soft-Robust algorithms [48], and Robust Q-learning algorithm [40] compared to the non-robust algorithms.

In each policy iteration loop, in both LSPI and RLSPI algorithms, we generate t trajectories of horizon length h using the last updated policy (the initial policy π_0 is random.) We generally stop the simulation after 10-20 policy iteration loops. The details of the hyper-parameters are shown in Table A.1 in addition to λ being set to zero.

OpenAI Gym Environment	Discount α	Weights error ε_0	t	h
CartPole	0.95	0.01	150	200
MountainCar	0.95	0.05	1000	20
Acrobot	0.98	0.1	100	200
FrozenLake8x8	0.99	0.01	3000	200

Table A.1: Details of hyper-parameters in LSPI and RLSPI algorithms experiments.

Here are the details on the Q-learning based algorithms. The Q-learning algorithm with linear function approximation on CartPole uses 64 parameterized (centers and variances) RBFs chosen by the Adam optimizer. Both Q-learning and Soft-Robust Q-learning algorithms for FrozenLake8x8 uses the tabular method, instead of deep neural or linear function architectures. We use the usual decaying-epsilon-greedy for the exploration policies, such that it exponentially decays to 0.01 at half-way through the total number of training episodes (Epochs in Table A.2) starting from 0.99. We provide the hyper-parameters in Table A.2 like the discount factor, size of hidden layers h starting from the first hidden layer in the given array, and size of the batch of tuples (state, action, next state, reward) chosen uniformly from the experience buffer of size 5000 to update the neural network (Batch in Table A.2). For all neural networks, we used the *relu* activation functions. Note that the DDPG algorithm uses two same sized neural networks for actor and critic.

For completeness, we also point out some weaknesses of the experiments we have done. Firstly, we are not optimizing over the parameter r which is the radius of the spherical set associated with the uncertainty. We believe that performing a hyper-parameter search for the best r will make the policy obtained by the RLSPI further robust. Secondly, since we are focusing on the

OpenAI Gym Environment	Discount α	Hidden layers h	Batch	Epochs
CartPole	0.9	$[200, 100]$	200	3000
MountainCar	0.99	$[300, 200]$	40	1000
Acrobot	0.995	$[128, 64, 64]$	500	10000
FrozenLake8x8	0.95	-	-	80000

Table A.2: Details of hyper-parameters in Q-learning based algorithms experiments.

linear approximation architecture for developing the theoretical understanding of model-free robust RL, the experiments may not be immediately scalable to very high dimensional OpenAI Gym environments which typically require nonlinear approximation architecture.

To end this section, we mention the software configurations used to generate these results: *Python3.7 with OpenAI Gym [71] and few basic libraries (non-exhaustive) like numpy, scipy, matplotlib*. Also, the hardware configurations used was *macOS High Sierra Version 10.13.6, 16 GB LPDDR3, Intel Core i7.*

APPENDIX B

APPENDIX FOR CHAPTER 3*

In this appendix, we include complete proofs, all experiments, and all supporting details for the corresponding chapter.

B.1 Useful Technical Results

In this section we state some existing results that are useful in our analysis.

Lemma 15 (Hoeffding's inequality [164, Theorem 2.2.6]). *Let $X_1, \cdots, X_T$ be independent random variables. Assume that $X_t \in [m_t, M_t]$ for every t with $M_t > m_t$. Then, for any $\varepsilon > 0$, we have*

$$\mathbb{P}\left(\sum_{t=1}^{T}(X_t - \mathbb{E}[X_t]) \geq \varepsilon\right) \leq \exp\left(-\frac{2\varepsilon^2}{\sum_{t=1}^{T}(M_t - m_t)^2}\right).$$

Lemma 16 (Self-bounding variance inequality [165, Theorem 10]). *Let $X_1, \cdots, X_T$ be independent and identically distributed random variables with finite variance, that is, $\mathrm{Var}(X_1) < \infty$. Assume that $X_t \in [0, M]$ for every t with $M > 0$, and let $S_T^2 = \frac{1}{T}\sum_{t=1}^{T} X_t^2 - (\frac{1}{T}\sum_{t=1}^{T} X_t)^2$. Then, for any $\varepsilon > 0$, we have*

$$\mathbb{P}\left(|S_T - \sqrt{\mathrm{Var}(X_1)}| \geq \varepsilon\right) \leq 2\exp\left(-\frac{T\varepsilon^2}{2M^2}\right).$$

Proof. The proof of this lemma directly follows from [165, Theorem 10] by noting that we can rewrite S_T^2 as follows

$$\frac{T}{T-1}S_T^2 = \frac{1}{T(T-1)}\sum_{i,j=1}^{T}(X_i - X_j)^2.$$

Also, note that we apply [165, Theorem 10] for the scaled random variables $X_t/M \in [0, 1]$. □

We now provide a covering number result that is useful to get high probability concentration

*Reprinted with permission from Kishan Panaganti, Dileep Kalathil, "Sample Complexity of Robust Reinforcement Learning with a Generative Model." AISTATS 2022.

bounds for value function classes. We first define minimal η-cover of a set.

Definition 2 (Minimal η-cover; [166, Definition 5.5])**.** *A set $\mathcal{N}_{\mathcal{V}}(\eta)$ is called an η-cover for a metric space $(\mathcal{V}, d)$ if for every $V \in \mathcal{V}$, there exists a $V' \in \mathcal{N}$ such that $d(V, V') \leq \eta$. Furthermore, $\mathcal{N}_{\mathcal{V}}(\eta)$ with the minimal cardinality ($|\mathcal{N}_{\mathcal{V}}(\eta)|$) is called a minimal η-cover.*

From [166, Exercise 5.5 and Lemma 5.13] we have the following result.

Lemma 17 (Covering Number)**.** *Let $\mathcal{V} = \{V \in \mathbb{R}^{|\mathcal{S}|} \ : \ \|V\| \leq V_{\max}\}$. Let $\mathcal{N}_{\mathcal{V}}(\eta)$ be a minimal η-cover of $\mathcal{V}$ with respect to the distance metric $d(V, V') = \|V - V'\|$ for some fixed $\eta \in (0, 1)$. Then we have*

$$\log |\mathcal{N}_{\mathcal{V}}(\eta)| \leq |\mathcal{S}| \cdot \log \left(\frac{3\, V_{\max}}{\eta} \right).$$

Proof. We will consider the normalized metric space $(\mathcal{V}_n, d_n)$, where

$$\mathcal{V}_n := \mathcal{V}/V_{\max} = \{V \in \mathbb{R}^{|\mathcal{S}|} \ : \ \|V\| \leq 1\}$$

and $d_n := d/V_{\max}$ to make use of the fact that the covering number is invariant to the scaling of a metric space. Let $\eta_n := \eta/V_{\max}$. Then, it follows from [166, Exercise 5.5 and Lemma 5.13] that

$$\log |\mathcal{N}_{\mathcal{V}}(\eta)| = \log |\mathcal{N}_{\mathcal{V}_n}(\eta_n)| \leq |\mathcal{S}| \cdot \log \left(\frac{3}{\eta_n} \right) = |\mathcal{S}| \cdot \log \left(\frac{3\, V_{\max}}{\eta} \right).$$

This completes the proof. □

Here we present another covering number result, with a similar proof as Lemma 17, that is useful to get our upperbound for the KL uncertainty set.

Lemma 18 (Covering Number of a bounded real line)**.** *Let $\Theta \subset \mathbb{R}$ with $\Theta = [l, u]$ for some real numbers $u > l$. Let $\mathcal{N}_{\Theta}(\eta)$ be a minimal η-cover of Θ with respect to the distance metric $d(\theta, \theta') = |\theta - \theta'|$ for some fixed $\eta \in (0, 1)$. Then we have $|\mathcal{N}_{\Theta}(\eta)| \leq 3(u - l)/\eta$.*

B.2 Proof of the Theorems

B.2.1 Concentration Results

Here, we prove Lemma 3. We state the following result first.

Lemma 19. *For any $V \in \mathbb{R}^{|\mathcal{S}|}$ with $\|V\| \leq V_{\max}$, with probability at least $1-\delta$,*

$$\max_{(s,a)} |P^o_{s,a}V - \widehat{P}_{s,a}V| \leq V_{\max}\sqrt{\frac{\log(2|\mathcal{S}||\mathcal{A}|/\delta)}{2N}}$$

Proof. Fix any (s,a) pair. Consider a discrete random variable X taking value $V(j)$ with probability $P^o_{s,a}(j)$ for all $j \in \{1,2,\cdots,|\mathcal{S}|\}$. From Hoeffding's inequality (Lemma 15), we have

$$\mathbb{P}(P^o_{s,a}V - \widehat{P}_{s,a}V \geq \varepsilon) \leq \exp(-2N\varepsilon^2/V^2_{\max}), \quad \mathbb{P}(\widehat{P}_{s,a}V - P^o_{s,a}V \geq \varepsilon) \leq \exp(-2N\varepsilon^2/V^2_{\max}).$$

Choosing $\varepsilon = V_{\max}\sqrt{\frac{\log(2|\mathcal{S}||\mathcal{A}|/\delta)}{2N}}$, we get $\mathbb{P}(|P^o_{s,a}V - \widehat{P}_{s,a}V| \geq V_{\max}\sqrt{\frac{\log(2|\mathcal{S}||\mathcal{A}|/\delta)}{2N}}) \leq \frac{\delta}{|\mathcal{S}||\mathcal{A}|}$. Now, using union bound, we get

$$\mathbb{P}(\max_{(s,a)} |P^o_{s,a}V - \widehat{P}_{s,a}V| \geq V_{\max}\sqrt{\frac{\log(2|\mathcal{S}||\mathcal{A}|/\delta)}{2N}}) \leq \sum_{s,a} \mathbb{P}(|P^o_{s,a}V - \widehat{P}_{s,a}V| \geq V_{\max}\sqrt{\frac{\log(2|\mathcal{S}||\mathcal{A}|/\delta)}{2N}}) \leq \delta.$$

This completes the proof. □

Proof of Lemma 3: Let $\mathcal{V} = \{V \in \mathbb{R}^{|\mathcal{S}|} : \|V\|_\infty \leq 1/(1-\gamma)\}$. Let $\mathcal{N}_{\mathcal{V}}(\eta)$ be a minimal η-cover of $\mathcal{V}$. Fix a $\mu \leq V$. By the definition of $\mathcal{N}_{\mathcal{V}}(\eta)$, there exists a $\mu' \in \mathcal{N}_{\mathcal{V}}(\eta)$ such that $\|\mu - \mu'\| \leq \eta$. Now, for these particular μ and μ', we get

$$\begin{aligned}
|\widehat{P}_{s,a}\mu - P^o_{s,a}\mu| &\leq |\widehat{P}_{s,a}\mu - \widehat{P}_{s,a}\mu'| + |\widehat{P}_{s,a}\mu' - P^o_{s,a}\mu'| + |P^o_{s,a}\mu' - P^o_{s,a}\mu| \\
&\overset{(a)}{\leq} \|\widehat{P}_{s,a}\|_1\|\mu - \mu'\|_\infty + |\widehat{P}_{s,a}\mu' - P^o_{s,a}\mu'| + \|P^o_{s,a}\|_1\|\mu' - \mu\|_\infty \\
&\leq \sup_{\mu' \in \mathcal{N}_{\mathcal{V}}(\eta)} \max_{s,a} |\widehat{P}_{s,a}\mu' - P^o_{s,a}\mu'| + 2\eta
\end{aligned}$$

where (a) follows from Hölder's inequality. Now, taking max on both sides with respect to μ and

(s,a) we get

$$\begin{aligned}\sup_{\mu\in\mathcal{V}}\max_{s,a}|\widehat{P}_{s,a}\mu - P^o_{s,a}\mu| &\le \sup_{\mu'\in\mathcal{N}_{\mathcal{V}}(\eta)}\max_{s,a}|\widehat{P}_{s,a}\mu' - P^o_{s,a}\mu'| + 2\eta \\ &\overset{(b)}{\le} \frac{1}{1-\gamma}\sqrt{\frac{\log(4|\mathcal{S}||\mathcal{A}||\mathcal{N}_{\mathcal{V}}(\eta)|/\delta)}{2N}} + 2\eta \\ &\overset{(c)}{\le} \frac{1}{1-\gamma}\sqrt{\frac{|\mathcal{S}|\log(12|\mathcal{S}||\mathcal{A}|/(\delta\eta(1-\gamma)))}{2N}} + 2\eta,\end{aligned}$$

with probability at least $1-\delta/2$. Here, (b) follows from Lemma 19 and the union bound and (c) from Lemma 17. □

B.2.2 Proof of Theorem 4

Proof of Lemma 1. We only prove the first inequality since the proof is analogous for the other inequality. For any $(s,a)\in\mathcal{S}\times\mathcal{A}$ we have

$$\begin{aligned}\sigma_{\mathcal{P}_{s,a}}(V_2) - \sigma_{\mathcal{P}_{s,a}}(V_1) &= \inf_{q\in\mathcal{P}_{s,a}} q^\top V_2 - \inf_{\tilde{q}\in\mathcal{P}_{s,a}} \tilde{q}^\top V_1 = \inf_{q\in\mathcal{P}_{s,a}}\sup_{\tilde{q}\in\mathcal{P}_{s,a}} q^\top V_2 - \tilde{q}^\top V_1 \\ &\ge \inf_{q\in\mathcal{P}_{s,a}} q^\top(V_2 - V_1) = \sigma_{\mathcal{P}_{s,a}}(V_2 - V_1). \qquad \text{(B.1)}\end{aligned}$$

By definition, for any arbitrary $\varepsilon > 0$, there exists a $P_{s,a}\in\mathcal{P}_{s,a}$ such that

$$P^\top_{s,a}(V_2 - V_1) - \varepsilon \le \sigma_{\mathcal{P}_{s,a}}(V_2 - V_1). \qquad \text{(B.2)}$$

Using (B.2) and (B.1),

$$\sigma_{\mathcal{P}_{s,a}}(V_1) - \sigma_{\mathcal{P}_{s,a}}(V_2) \le P^\top_{s,a}(V_1 - V_2) + \varepsilon \overset{(a)}{\le} \|P_{s,a}\|_1\|V_1 - V_2\| + \varepsilon = \|V_1 - V_2\| + \varepsilon$$

where (a) follows from Holder's inequality. Since ε is arbitrary, we get, $\sigma_{\mathcal{P}_{s,a}}(V_1) - \sigma_{\mathcal{P}_{s,a}}(V_2) \le \|V_1 - V_2\|$. Exchanging the roles of V_1 and V_2 completes the proof. □

Proof of Lemma 2. Fix any (s,a) pair. From [33, Lemma 4.3] we have that

$$\sigma_{\mathcal{P}^{\text{tv}}_{s,a}}(V) = P^o_{s,a}V + \max_{\mu:0\le\mu\le V}\Big(- P^o_{s,a}\mu - c_r \max_s(V_\mu(s)) + c_r \min_s(V_\mu(s))\Big) \tag{B.3}$$

$$\sigma_{\widehat{\mathcal{P}}^{\text{tv}}_{s,a}}(V) = \widehat{P}_{s,a}V + \max_{\mu:0\le\mu\le V}\Big(- \widehat{P}_{s,a}\mu - c_r \max_s(V_\mu(s)) + c_r \min_s(V_\mu(s))\Big), \tag{B.4}$$

where $0 \le \mu \le V$ is an elementwise inequality and $V_\mu(s) = V(s) - \mu(s)$ for all $s \in \mathcal{S}$.

Using the fact that $|\max_x f(x) - \max_x g(x)| \le \max_x |f(x) - g(x)|$, it directly follows that

$$|\sigma_{\widehat{\mathcal{P}}^{\text{tv}}_{s,a}}(V) - \sigma_{\mathcal{P}^{\text{tv}}_{s,a}}(V)| \le |\widehat{P}_{s,a}V - P^o_{s,a}V| + \max_{\mu:0\le\mu\le V} |\widehat{P}_{s,a}\mu - P^o_{s,a}\mu|.$$

Further simplifying we get

$$\begin{aligned}|\sigma_{\widehat{\mathcal{P}}^{\text{tv}}_{s,a}}(V) - \sigma_{\mathcal{P}^{\text{tv}}_{s,a}}(V)| &\le |\widehat{P}_{s,a}V - P^o_{s,a}V| + \max_{\mu:0\le\mu\le V} |\widehat{P}_{s,a}\mu - P^o_{s,a}\mu| \\ &\le \max_{\mu\in\mathcal{V}} |\widehat{P}_{s,a}\mu - P^o_{s,a}\mu| + \max_{\mu:0\le\mu\le V} |\widehat{P}_{s,a}\mu - P^o_{s,a}\mu| \le 2\max_{\mu\in\mathcal{V}} |\widehat{P}_{s,a}\mu - P^o_{s,a}\mu|.\end{aligned}$$

This completes the proof. □

We are now ready to prove Proposition 4.

Proof of Proposition 4. For any given $V \in \mathcal{V}$ and (s,a), from Lemma 2, we have

$$|\sigma_{\widehat{\mathcal{P}}^{\text{tv}}_{s,a}}(V) - \sigma_{\mathcal{P}^{\text{tv}}_{s,a}}(V)| \le 2\max_{\mu\in\mathcal{V}} |\widehat{P}_{s,a}\mu - P^o_{s,a}\mu| \le 2\max_{\mu\in\mathcal{V}}\max_{s,a} |\widehat{P}_{s,a}\mu - P^o_{s,a}\mu|.$$

Taking the maximum over V and (s,a) on both sides, we get

$$\max_{V\in\mathcal{V}}\max_{s,a} |\sigma_{\widehat{\mathcal{P}}^{\text{tv}}_{s,a}}(V) - \sigma_{\mathcal{P}^{\text{tv}}_{s,a}}(V)| \le 2\max_{\mu\in\mathcal{V}}\max_{s,a} |\widehat{P}_{s,a}\mu - P^o_{s,a}\mu|. \tag{B.5}$$

Now, from the proof of Lemma 3, for any $\eta, \delta \in (0,1)$, we get

$$\max_{\mu \in \mathcal{V}} \max_{s,a} |\widehat{P}_{s,a}\mu - P^o_{s,a}\mu| \leq \frac{1}{1-\gamma}\sqrt{\frac{|\mathcal{S}|\log(6|\mathcal{S}||\mathcal{A}|/(\delta\eta(1-\gamma)))}{2N}} + 2\eta, \tag{B.6}$$

with probability greater than $1-\delta$. Using (B.6) in (B.5), we get the desired result. □

We also need the following result that specifies the amplification when replacing the algorithm iterate value function with the value function of the policy towards approximating the optimal value.

Lemma 20. *Let V_k and Q_k be as given in the REVI algorithm for $k \geq 1$. Also, let $\pi_k = \operatorname{argmax}_a Q_k(s,a)$. Then,*

$$\|\widehat{V}^* - \widehat{V}^{\pi_k}\| \leq \frac{2\gamma}{1-\gamma}\|V_k - \widehat{V}^*\|.$$

Furthermore,

$$\|V^* - V^{\pi_k}\| \leq \frac{2}{1-\gamma}\|Q_k - Q^*\|.$$

Proof. The proof is similar to the proof in [75, Main Theorem, Corollary 2]. A straight forward modification to this proof, using the fact that $\sigma_{\widehat{\mathcal{P}}_{s,a}}$ and $\sigma_{\mathcal{P}_{s,a}}$ are 1-Lipschitz functions as shown in Lemma 1, will give the desired result. □

Proof of Theorem 4. Recall the empirical RMDP $\widehat{M} = (\mathcal{S}, \mathcal{A}, r, \widehat{\mathcal{P}}^{\text{tv}}, \gamma)$. For any policy π, let $\widehat{V}^\pi$ be robust value function of policy π with respect to the RMDP $\widehat{M}$. The optimal robust policy, value function, and state-action value function of $\widehat{M}$ are denoted as $\widehat{\pi}^\star, \widehat{V}^\star$ and $\widehat{Q}^\star$, respectively. Also, for any policy π, we have $\widehat{Q}^\pi(s,a) = r(s,a) + \gamma\sigma_{\widehat{\mathcal{P}}^{\text{tv}}_{s,a}}(\widehat{V}^\pi)$ and $Q^\pi(s,a) = r(s,a) + \gamma\sigma_{\mathcal{P}^{\text{tv}}_{s,a}}(V^\pi)$.

Let V_k and Q_k be as given in the REVI algorithm for $k \geq 1$. Also, let $\pi_k(s) = \operatorname{argmax}_a Q_k(s,a)$. Now,

$$\|V^* - V^{\pi_k}\| \leq \|V^* - \widehat{V}^*\| + \|\widehat{V}^* - \widehat{V}^{\pi_k}\| + \|\widehat{V}^{\pi_k} - V^{\pi_k}\|. \tag{B.7}$$

1) Bounding the first term in (B.7)*:* Let $\mathcal{V} = \{V \in \mathbb{R}^{|\mathcal{S}|} : \|V\| \leq 1/(1-\gamma)\}$. For any $s \in \mathcal{S}$,

$$\begin{aligned}
V^*(s) - \widehat{V}^*(s) &= Q^*(s, \pi^*(s)) - \widehat{Q}^*(s, \hat{\pi}^*(s)) \overset{(a)}{\leq} Q^*(s, \pi^*(s)) - \widehat{Q}^*(s, \pi^*(s)) \\
&\overset{(b)}{=} \gamma \sigma_{\mathcal{P}^{\mathrm{tv}}_{s,\pi^*(s)}}(V^*) - \gamma \sigma_{\widehat{\mathcal{P}}^{\mathrm{tv}}_{s,\pi^*(s)}}(\widehat{V}^*) \\
&= \gamma(\sigma_{\mathcal{P}^{\mathrm{tv}}_{s,\pi^*(s)}}(V^*) - \sigma_{\widehat{\mathcal{P}}^{\mathrm{tv}}_{s,\pi^*(s)}}(V^*)) + \gamma(\sigma_{\widehat{\mathcal{P}}^{\mathrm{tv}}_{s,\pi^*(s)}}(V^*) - \sigma_{\widehat{\mathcal{P}}^{\mathrm{tv}}_{s,\pi^*(s)}}(\widehat{V}^*)) \\
&\overset{(c)}{\leq} \gamma(\sigma_{\mathcal{P}^{\mathrm{tv}}_{s,\pi^*(s)}}(V^*) - \sigma_{\widehat{\mathcal{P}}^{\mathrm{tv}}_{s,\pi^*(s)}}(V^*)) + \gamma\|V^* - \widehat{V}^*\| \\
&\leq \gamma \max_{V \in \mathcal{V}} \max_{s,a} |\sigma_{\widehat{\mathcal{P}}^{\mathrm{tv}}_{s,a}}(V) - \sigma_{\mathcal{P}^{\mathrm{tv}}_{s,a}}(V)| + \gamma\|V^* - \widehat{V}^*\|
\end{aligned}$$

where (a) follows since $\hat{\pi}^*$ is the robust optimal policy for $\widehat{M}$, (b) follows from the definitions of Q^* and $\widehat{Q}^*$, (c) follows from Lemma 1. Similarly analyzing for $\widehat{V}^*(s) - V^*(s)$, we get

$$\|V^* - \widehat{V}^*\| \leq \frac{\gamma}{(1-\gamma)} \max_{V \in \mathcal{V}} \max_{s,a} |\sigma_{\widehat{\mathcal{P}}^{\mathrm{tv}}_{s,a}}(V) - \sigma_{\mathcal{P}^{\mathrm{tv}}_{s,a}}(V)|. \tag{B.8}$$

Now, using Proposition 4, with probability greater than $1-\delta$, we get

$$\|V^* - \widehat{V}^*\| \leq \frac{\gamma}{(1-\gamma)} C_u^{\mathrm{tv}}(N, \eta, \delta), \tag{B.9}$$

where $C_u^{\mathrm{tv}}(N, \eta, \delta)$ is given in equation (3.15) in the statement of Proposition 4.

2) Bounding the second term in (B.7)*:* Let $\widehat{T}$ be the robust Bellman operator corresponding to $\widehat{M}$. So, $\widehat{T}$ is a γ-contraction mapping and $\widehat{V}^*$ is its unique fixed point [33]. The REVI iterates $V_k, k \geq 0$, with $V_0 = 0$, can now be expressed as $V_{k+1} = \widehat{T}V_k$. Using the properties of $\widehat{T}$, we get

$$\|V_k - \widehat{V}^*\| = \|\widehat{T}V_{k-1} - \widehat{T}\widehat{V}^*\| \leq \gamma\|V_{k-1} - \widehat{V}^*\| \leq \cdots \leq \gamma^k\|V_0 - \widehat{V}^*\| \leq \gamma^k/(1-\gamma). \tag{B.10}$$

Now, using Lemma 20, we get

$$\|\widehat{V}^{\pi_k} - \widehat{V}^*\| \leq \frac{2\gamma^{k+1}}{(1-\gamma)^2}. \tag{B.11}$$

3) Bounding the third term in (B.7)*:* This is similar to bounding the first term. For any $s \in \mathcal{S}$,

$$\begin{aligned} V^{\pi_k}(s) - \widehat{V}^{\pi_k}(s) &= Q^{\pi_k}(s, \pi_k(s)) - \widehat{Q}^{\pi_k}(s, \pi_k(s)) = \gamma \sigma_{\mathcal{P}_{s,\pi_k(s)}}(V^{\pi_k}) - \gamma \sigma_{\widehat{\mathcal{P}}_{s,\pi_k(s)}}(\widehat{V}^{\pi_k}) \\ &= \gamma(\sigma_{\mathcal{P}_{s,\pi_k(s)}}(V^{\pi_k}) - \sigma_{\mathcal{P}_{s,\pi_k(s)}}(\widehat{V}^{\pi_k})) + \gamma(\sigma_{\mathcal{P}_{s,\pi_k(s)}}(\widehat{V}^{\pi_k}) - \sigma_{\widehat{\mathcal{P}}_{s,\pi_k(s)}}(\widehat{V}^{\pi_k})) \\ &\overset{(d)}{\leq} \gamma \|V^{\pi_k} - \widehat{V}^{\pi_k}\| + \gamma(\sigma_{\mathcal{P}_{s,\pi_k(s)}}(\widehat{V}^{\pi_k}) - \sigma_{\widehat{\mathcal{P}}_{s,\pi_k(s)}}(\widehat{V}^{\pi_k})) \\ &\leq \gamma \|V^{\pi_k} - \widehat{V}^{\pi_k}\| + \gamma \max_{V \in \mathcal{V}} \max_{s,a} |\sigma_{\widehat{\mathcal{P}}^{\text{tv}}_{s,a}}(V) - \sigma_{\mathcal{P}^{\text{tv}}_{s,a}}(V)| \end{aligned}$$

where (d) follows from Lemma 1. Similarly analyzing for $\widehat{V}^{\pi_k}(s) - V^{\pi_k}(s)$, we get,

$$\|V^{\pi_k} - \widehat{V}^{\pi_k}\| \leq \frac{\gamma}{(1-\gamma)} \max_{V \in \mathcal{V}} \max_{s,a} |\sigma_{\widehat{\mathcal{P}}^{\text{tv}}_{s,a}}(V) - \sigma_{\mathcal{P}^{\text{tv}}_{s,a}}(V)|. \tag{B.12}$$

Now, using Proposition 4, with probability greater than $1 - \delta$, we get

$$\|V^{\pi_k} - \widehat{V}^{\pi_k}\| \leq \frac{\gamma}{(1-\gamma)} C_u^{\text{tv}}(N, \eta, \delta). \tag{B.13}$$

Using (B.9) - (B.13) in (B.7), we get, with probability at least $1 - 2\delta$,

$$\|V^* - V^{\pi_k}\| \leq \frac{2\gamma^{k+1}}{(1-\gamma)^2} + \frac{2\gamma}{(1-\gamma)} C_u^{\text{tv}}(N, \eta, \delta). \tag{B.14}$$

Using the value of $C_u^{\text{tv}}(N, \eta, \delta)$ as given in Proposition 4, we get

$$\|V^* - V^{\pi_k}\| \leq \frac{2\gamma^{k+1}}{(1-\gamma)^2} + \frac{4\gamma}{(1-\gamma)^2} \sqrt{\frac{|\mathcal{S}| \log(6|\mathcal{S}||\mathcal{A}|/(\delta\eta(1-\gamma)))}{2N}} + \frac{8\gamma\eta}{(1-\gamma)} \tag{B.15}$$

with probability at least $1 - 2\delta$.

Now, choose $\eta = \varepsilon(1-\gamma)/(24\gamma)$. Since $\varepsilon \in (0, 24\gamma/(1-\gamma))$, this particular η is in $(0, 1)$. Now, choosing

$$k \geq K_0 = \frac{1}{\log(1/\gamma)} \log(\frac{6\gamma}{\varepsilon(1-\gamma)^2}), \tag{B.16}$$

$$N \geq N^{\mathrm{tv}} = \frac{72\gamma^2}{(1-\gamma)^4} \frac{|\mathcal{S}| \log(144\gamma|\mathcal{S}||\mathcal{A}|/(\delta\varepsilon(1-\gamma)^2))}{\varepsilon^2}, \tag{B.17}$$

we get $\|V^* - V^{\pi_k}\| \leq \varepsilon$ with probability at least $1 - 2\delta$. □

B.2.3 Proof of Theorem 5

Proof of Lemma 4. Fix an (s, a) pair. From [33, Lemma 4.2], we have

$$\sigma_{\mathcal{P}^{\mathrm{c}}_{s,a}}(V) = \max_{\mu: 0 \leq \mu \leq V} \left(P^o_{s,a}(V - \mu) - \sqrt{c_r \mathrm{Var}_{P^o_{s,a}}(V - \mu)} \right), \tag{B.18}$$

where $\mathrm{Var}_{P^o_{s,a}}(V - \mu) = P^o_{s,a}(V - \mu)^2 - (P^o_{s,a}(V - \mu))^2$. We get a similar expression for $\sigma_{\widehat{\mathcal{P}}^{\mathrm{c}}_{s,a}}(V)$. Using these expressions, with the additional facts that $|\max_x f(x) - \max_x g(x)| \leq \max_x |f(x) - g(x)|$ and $\max_x(f(x) + g(x)) \leq \max_x f(x) + \max_x g(x)$, we get the desired result. □

We state the following concentration result that is useful for the proof of Proposition 5.

Lemma 21. *For any $V \in \mathbb{R}_+^{|\mathcal{S}|}$ with $\|V\| \leq V_{\max}$, with probability at least $1 - \delta$,*

$$\max_{(s,a)} |\sqrt{\mathit{Var}_{P^o_{s,a}} V} - \sqrt{\mathit{Var}_{\widehat{P}_{s,a}} V}| \leq V_{\max} \sqrt{\frac{2 \log(2|\mathcal{S}||\mathcal{A}|/\delta)}{N}}$$

Proof. Fix any (s, a) pair. Consider a discrete random variable X taking value $V(j)$ with probability $P^o_{s,a}(j)$ for all $j \in \{1, 2, \cdots, |\mathcal{S}|\}$. From the Self-bounding variance inequality (Lemma 16), we have

$$\mathbb{P}(|\sqrt{\mathrm{Var}_{P^o_{s,a}} V} - \sqrt{\mathrm{Var}_{\widehat{P}_{s,a}} V}| \geq \varepsilon) \leq 2 \exp(-N\varepsilon^2/(2V^2_{\max})).$$

Choosing $\varepsilon = V_{\max} \sqrt{\frac{2 \log(2|\mathcal{S}||\mathcal{A}|/\delta)}{N}}$, we get $\mathbb{P}(|P^o_{s,a} V - \widehat{P}_{s,a} V| \geq V_{\max} \sqrt{\frac{2 \log(2|\mathcal{S}||\mathcal{A}|/\delta)}{N}}) \leq \frac{\delta}{|\mathcal{S}||\mathcal{A}|}$. Now, using union bound, we get

$$\mathbb{P}(\max_{(s,a)} |\sqrt{\mathrm{Var}_{P^o_{s,a}} V} - \sqrt{\mathrm{Var}_{\widehat{P}_{s,a}} V}| \geq V_{\max} \sqrt{\frac{2 \log(2|\mathcal{S}||\mathcal{A}|/\delta)}{N}})$$

$$\leq \sum_{s,a} \mathbb{P}(|\sqrt{\text{Var}_{P^o_{s,a}} V} - \sqrt{\text{Var}_{\widehat{P}_{s,a}} V}| \geq V_{\max} \sqrt{\frac{2\log(2|\mathcal{S}||\mathcal{A}|/\delta)}{N}}) \leq \delta.$$

This completes the proof. □

We are now ready to prove Proposition 5.

Proof of Proposition 5. Fix an (s, a) pair. From Lemma 4, for any given $V \in \mathcal{V}$, we have

$$|\sigma_{\widehat{\mathcal{P}}^{\text{c}}_{s,a}}(V) - \sigma_{\mathcal{P}^{\text{c}}_{s,a}}(V)| \leq \max_{\mu:0\leq\mu\leq V} |\sqrt{c_r \text{Var}_{\widehat{P}_{s,a}}(V-\mu)} - \sqrt{c_r \text{Var}_{P^o_{s,a}}(V-\mu)}| + \max_{\mu:0\leq\mu\leq V} |\widehat{P}_{s,a}(V-\mu) - P^o_{s,a}(V-\mu)|.$$

By a simple variable substitution, we get

$$|\sigma_{\widehat{\mathcal{P}}^{\text{c}}_{s,a}}(V) - \sigma_{\mathcal{P}^{\text{c}}_{s,a}}(V)| \leq \max_{\mu\in\mathcal{V}_+} \max_{s,a} |\sqrt{c_r \text{Var}_{\widehat{P}_{s,a}}\mu} - \sqrt{c_r \text{Var}_{P^o_{s,a}}\mu}| + \max_{\mu\in\mathcal{V}} \max_{s,a} |\widehat{P}_{s,a}\mu - P^o_{s,a}\mu|,$$

which will give

$$\max_{V\in\mathcal{V}} \max_{s,a} |\sigma_{\widehat{\mathcal{P}}^{\text{c}}_{s,a}}(V) - \sigma_{\mathcal{P}^{\text{c}}_{s,a}}(V)| \leq \max_{\mu\in\mathcal{V}_+} \max_{s,a} |\sqrt{c_r \text{Var}_{\widehat{P}_{s,a}}\mu} - \sqrt{c_r \text{Var}_{P^o_{s,a}}\mu}| + \max_{\mu\in\mathcal{V}} \max_{s,a} |\widehat{P}_{s,a}\mu - P^o_{s,a}\mu|, \tag{B.19}$$

where $\mathcal{V}_+ = \{V \in \mathbb{R}^{|\mathcal{S}|}_+ : \|V\| \leq 1/(1-\gamma)\}$.

We will first bound the second term on the RHS of (B.19). From the proof of Lemma 3, for any $\eta, \delta \in (0, 1)$, we get

$$\max_{\mu\in\mathcal{V}} \max_{s,a} |\widehat{P}_{s,a}\mu - P^o_{s,a}\mu| \leq \frac{1}{1-\gamma} \sqrt{\frac{|\mathcal{S}| \log(12|\mathcal{S}||\mathcal{A}|/(\delta\eta(1-\gamma)))}{2N}} + 2\eta, \tag{B.20}$$

with probability greater than $1 - \delta/2$.

Now, we will focus on the first term on the RHS of (B.19). Fix a $\mu \in \mathcal{V}_+$. Consider a minimal η-cover $\mathcal{N}_{\mathcal{V}_+}(\eta)$ of the set $\mathcal{V}_+$. By definition, there exists $\mu' \in \mathcal{N}_{\mathcal{V}_+}(\eta)$ such that $\|\mu - \mu'\| \leq \eta$.

Now, following the same step as in the proof of Lemma 3, we get

$$
\begin{aligned}
|\sqrt{\text{Var}_{\widehat{P}_{s,a}}\mu} - \sqrt{\text{Var}_{P^o_{s,a}}\mu}| &\leq |\sqrt{\text{Var}_{\widehat{P}_{s,a}}\mu} - \sqrt{\text{Var}_{\widehat{P}_{s,a}}\mu'}| + |\sqrt{\text{Var}_{\widehat{P}_{s,a}}\mu'} - \sqrt{\text{Var}_{P^o_{s,a}}\mu'}| + |\sqrt{\text{Var}_{P^o_{s,a}}\mu} - \sqrt{\text{Var}_{P^o_{s,a}}} \\
&\overset{(a)}{\leq} |\sqrt{\text{Var}_{\widehat{P}_{s,a}}\mu'} - \sqrt{\text{Var}_{P^o_{s,a}}\mu'}| + \sqrt{|\text{Var}_{\widehat{P}_{s,a}}\mu - \text{Var}_{\widehat{P}_{s,a}}\mu'|} + \sqrt{|\text{Var}_{P^o_{s,a}}\mu - \text{Var}_{P^o_{s,a}}\mu'|} \\
&\overset{(b)}{\leq} |\sqrt{\text{Var}_{\widehat{P}_{s,a}}\mu'} - \sqrt{\text{Var}_{P^o_{s,a}}\mu'}| + \sqrt{|\widehat{P}_{s,a}(\mu^2 - \mu'^2)|} + \sqrt{|(\widehat{P}_{s,a}\mu)^2 - (\widehat{P}_{s,a}\mu')^2)|} + \\
&\qquad\qquad \sqrt{|P^o_{s,a}(\mu^2 - \mu'^2)|} + \sqrt{|(P^o_{s,a}\mu)^2 - (P^o_{s,a}\mu')^2)|} \\
&\overset{(c)}{\leq} |\sqrt{\text{Var}_{\widehat{P}_{s,a}}\mu'} - \sqrt{\text{Var}_{P^o_{s,a}}\mu'}| + \sqrt{\frac{32\eta}{1-\gamma}} \\
&\leq \sup_{\mu' \in \mathcal{N}_{\mathcal{V}_+}(\eta)} \max_{s,a} |\sqrt{\text{Var}_{\widehat{P}_{s,a}}\mu'} - \sqrt{\text{Var}_{P^o_{s,a}}\mu'}| + \sqrt{\frac{32\eta}{1-\gamma}}
\end{aligned}
$$

where (a) follows from the fact $|\sqrt{x} - \sqrt{y}| \leq \sqrt{|x-y|}$ for all $x, y \in \mathbb{R}_+$, (b) follows from the fact $|\sqrt{x+y}| \leq \sqrt{x} + \sqrt{y}$ for all $x, y \in \mathbb{R}_+$, and (c) follows by using the fact $x^2 - y^2 = (x+y)(x-y)$, $\|\mu\| \leq 1/(1-\gamma)$, and $\|\mu'\| \leq 1/(1-\gamma)$ with Hölder's inequality. Now, taking max on both sides with respect to μ and (s,a) we get

$$
\begin{aligned}
\sup_{\mu \in \mathcal{V}_+} \max_{s,a} |\sqrt{\text{Var}_{\widehat{P}_{s,a}}\mu} - \sqrt{\text{Var}_{P^o_{s,a}}\mu}| &\leq \sup_{\mu' \in \mathcal{N}_{\mathcal{V}_+}(\eta)} \max_{s,a} |\sqrt{\text{Var}_{\widehat{P}_{s,a}}\mu'} - \sqrt{\text{Var}_{P^o_{s,a}}\mu'}| + \sqrt{\frac{32\eta}{1-\gamma}} \\
&\overset{(d)}{\leq} \frac{1}{1-\gamma}\sqrt{\frac{2\log(4|\mathcal{S}||\mathcal{A}||\mathcal{N}_{\mathcal{V}_+}(\eta)|/\delta)}{N}} + \sqrt{\frac{32\eta}{1-\gamma}} \\
&\overset{(e)}{\leq} \frac{1}{1-\gamma}\sqrt{\frac{2|\mathcal{S}|\log(12|\mathcal{S}||\mathcal{A}|/(\delta\eta(1-\gamma)))}{N}} + \sqrt{\frac{32\eta}{1-\gamma}},
\end{aligned}
\tag{B.21}
$$

with probability at least $1 - \delta/2$. Here, (d) follows from Lemma 21 and the union bound and (e) from Lemma 17.

Applying (B.20) and (B.21) in (B.19), we get

$$
\begin{aligned}
\max_{V \in \mathcal{V}} \max_{s,a} |\sigma_{\widehat{\mathcal{P}}^c_{s,a}}(V) - \sigma_{\mathcal{P}^c_{s,a}}(V)| &\leq \frac{1}{1-\gamma}\sqrt{\frac{2c_r|\mathcal{S}|\log(12|\mathcal{S}||\mathcal{A}|/(\delta\eta(1-\gamma)))}{N}} + \sqrt{\frac{32\eta c_r}{1-\gamma}} \\
&\quad + \frac{1}{1-\gamma}\sqrt{\frac{|\mathcal{S}|\log(12|\mathcal{S}||\mathcal{A}|/(\delta\eta(1-\gamma)))}{2N}} + 2\eta,
\end{aligned}
$$

with probability greater than $1-\delta$. This completes the proof. $\square$

Proof of Theorem 5. The basic steps of the proof is similar to that of Theorem 4. So, we present only the important steps.

Following the same steps as given before (B.9) and using Proposition 5, we get, with probability greater than $1-\delta$,

$$\|V^* - \widehat{V}^*\| \leq \frac{\gamma}{(1-\gamma)} C_u^{\mathrm{c}}(N, \eta, \delta) \tag{B.22}$$

Similarly, following the steps as given before (B.11), we get

$$\|\widehat{V}^{\pi_k} - \widehat{V}^*\| \leq \frac{2\gamma^{k+1}}{(1-\gamma)^2}. \tag{B.23}$$

In the same vein, following the steps as given before (B.13) and using Proposition 5, we get, with probability greater than $1-\delta$,

$$\|V^{\pi_k} - \widehat{V}^{\pi_k}\| \leq \frac{\gamma}{(1-\gamma)} C_u^{\mathrm{c}}(N, \eta, \delta). \tag{B.24}$$

Using (B.22) - (B.24), similar to (B.14), we get, with probability greater than $1-2\delta$,

$$\|V^* - V^{\pi_k}\| \leq \frac{2\gamma^{k+1}}{(1-\gamma)^2} + \frac{2\gamma}{(1-\gamma)} C_u^{\mathrm{c}}(N, \eta, \delta). \tag{B.25}$$

Using the value of $C_u^{\mathrm{c}}(N, \eta, \delta)$ as given in Proposition 5, we get, with probability greater than $1-2\delta$,

$$\|V^* - V^{\pi_k}\| \leq \frac{2\gamma^{k+1}}{(1-\gamma)^2} + \frac{8\gamma\sqrt{2\eta c_r}}{(1-\gamma)^{3/2}} + \frac{4\gamma\eta}{1-\gamma} + \frac{2\gamma}{(1-\gamma)^2}\sqrt{\frac{(2c_r+1)|\mathcal{S}|\log(12|\mathcal{S}||\mathcal{A}|/(\delta\eta(1-\gamma)))}{N}}.$$

We can now choose k, ε, η to make each of the term on the RHS of the above inequality small. In particular, we select $\varepsilon \in (0, \min\{16\gamma/(1-\gamma), 32\gamma\sqrt{2c_r}/(1-\gamma)^{3/2}\})$ and $\eta = \min\{\varepsilon(1-$

$\gamma)/(16\gamma), \varepsilon^2(1-\gamma)^3/(2048c_r\gamma^2)\}$. Note that this choice also ensure $\eta \in (0,1)$. Now, by choosing

$$k \geq K_0 = \frac{1}{\log(1/\gamma)} \cdot \log(\frac{8\gamma}{\varepsilon(1-\gamma)^2}), \tag{B.26}$$

$$N \geq N^{\mathrm{c}} = \frac{64\gamma^2}{(1-\gamma)^4} \cdot \frac{(2c_r+1)|\mathcal{S}|\log(12|\mathcal{S}||\mathcal{A}|/(\delta\eta(1-\gamma)))}{\varepsilon^2}, \tag{B.27}$$

we will get $\|V^* - V^{\pi_k}\| \leq \varepsilon$ with probability at least $1-2\delta$. □

B.2.4 Proof of Theorem 6

We state a result from [82] that will be useful in the proof of Theorem 6.

Lemma 22 ([82, Lemma 4])**.** *Fix any $\delta \in (0,1)$. Let $X \sim P$ be a bounded random variable with $X \in [0, M]$ and let P_N denote the empirical distribution of P with N samples. For $t > 0$, for any*

$$\lambda^* \in \operatorname*{argmax}_{\lambda \geq 0} \left\{ -\lambda \log(\mathbb{E}_P[\exp(-X/\lambda)]) - \lambda t \right\},$$

(1) $\lambda^* = 0$. Furthermore, let the support of X be finite. Then there exists a problem dependent constant

$$N'(\delta, t, P) := \max\{\log(2/\delta)/\log(1/(1-\min_{x \in \mathrm{supp}(X)} P(X=x))),\ 2M^2\log(4/\delta)/(P(X=\operatorname{ess\,inf} X)-\exp(-t))^2\}$$

such that for $N \geq N'(\delta, t, P)$ we have, with probability at least $1-\delta$,

$$0 \in \operatorname*{argmax}_{\lambda \geq 0} \left\{ -\lambda \log(\mathbb{E}_{P_N}[\exp(-X/\lambda)]) - \lambda t \right\}.$$

(2) $\lambda^* > 0$. Then there exists a problem dependent constant

$$N''(\delta, t, P) := \max_{\lambda \in \{\underline{\lambda}, \lambda^*, M/t\}} \frac{8M^2\exp(2M/\lambda)}{\tau^2}\log(6/\delta),$$

where $\underline{\lambda} = \lambda^*/2 > 0$ *(independent of* N*) and*

$$\tau = \min\{\underline{\lambda}\log(\mathbb{E}_P[\exp(-X/\underline{\lambda})]) + \underline{\lambda}t, (M/t)\log(\mathbb{E}_P[\exp(-tX/M)]) + M\} \\ - (\lambda^*\log(\mathbb{E}_P[\exp(-X/\lambda^*)]) + \lambda^* t) > 0,$$

such that for $N \geq N''(\delta, t, P)$*, with probability at least* $1-\delta$*, there exists a*

$$\widehat{\lambda}^* \in \operatorname*{argmax}_{\lambda \geq 0} \left\{-\lambda\log(\mathbb{E}_{P_N}[\exp(-X/\lambda)]) - \lambda t\right\},$$

such that $\lambda^*, \widehat{\lambda}^* \in [\underline{\lambda}, M/t]$.

We now prove the following result.

Lemma 23. *For any* $(s,a) \in \mathcal{S} \times \mathcal{A}$ *and for any* $V \in \mathbb{R}^{|\mathcal{S}|}$ *with* $\|V\| \leq 1/(1-\gamma)$*,*

$$|\sigma_{\widehat{\mathcal{P}}^{\text{kl}}_{s,a}}(V) - \sigma_{\mathcal{P}^{\text{kl}}_{s,a}}(V)| \leq \frac{\exp(1/\lambda_{\text{kl}}(1-\gamma))}{c_r(1-\gamma)} \max_{\lambda \in [\lambda_{\text{kl}}, \frac{1}{c_r(1-\gamma)}]} |(P^o_{s,a} - \widehat{P}_{s,a})\exp(-V/\lambda)| \quad \text{(B.28)}$$

holds with probability at least $1-\delta/(2|\mathcal{S}||\mathcal{A}|)$ *for* $N \geq \max\{N'(\delta/(4|\mathcal{S}||\mathcal{A}|), c_r, P^o_{s,a}), N''(\delta/(4|\mathcal{S}||\mathcal{A}|), c_r, P^o_{s,a})\}$ *where both* N', N'' *are defined as in Lemma 22.*

Proof. Fix any (s,a) pair. From [33, Lemma 4.1], we have

$$\sigma_{\mathcal{P}^{\text{kl}}_{s,a}}(V) = \max_{\lambda \geq 0}(-c_r\lambda - \lambda\log(P^o_{s,a}\exp(-V/\lambda))), \quad \sigma_{\widehat{\mathcal{P}}^{\text{kl}}_{s,a}}(V) = \max_{\lambda \geq 0}(-c_r\lambda - \lambda\log(\widehat{P}_{s,a}\exp(-V/\lambda))), \quad \text{(B.29)}$$

where $\exp(-V/\lambda)$ is an element-wise exponential function. It is straight forward to show that $(-c_r\lambda - \lambda\log(P^o_{s,a}\exp(-V/\lambda)))$ is a concave function in λ. So, there exists an optimal solution λ^*. Similarly, let $\widehat{\lambda}^*$ be the optimal solution of the second problem above.

We can now give an upperbound for $\lambda^*, \widehat{\lambda}^*$ as follows: Since $\sigma_{\mathcal{P}^{\text{kl}}_{s,a}}(V) \geq 0$, we have

$$0 \leq -c_r\lambda^* - \lambda^*\log(P^o_{s,a}\exp(-V/\lambda^*)) \overset{(a)}{\leq} -c_r\lambda^* - \lambda^*\log(\exp(-1/(\lambda^*(1-\gamma)))) \leq -c_r\lambda^* + 1/(1-\gamma),$$

from which we can conclude that $\lambda^* \leq 1/(c_r(1-\gamma))$. Same argument applies for the case of $\widehat{\lambda}^*$.

From [34, Appendix C] it follows that whenever the maximizer λ^* is 0 ($\widehat{\lambda}^*$ is 0), we have $\sigma_{\mathcal{P}^{\text{kl}}_{s,a}}(V) = V_{\min}$ ($\sigma_{\widehat{\mathcal{P}}^{\text{kl}}_{s,a}}(V) = V_{\min}$) where $V_{\min} = \min_{j \in \mathcal{S}} V(j)$. We include this part in detail for completeness.

$$
\begin{aligned}
\lim_{\lambda \downarrow 0} -c_r\lambda - \lambda \log(P^o_{s,a} \exp(-V/\lambda)) &= \lim_{\lambda \downarrow 0} -c_r\lambda - \lambda \log(\exp(-V_{\min}/\lambda) \sum_{s'} P^o_{s,a}(s') \exp((V_{\min} - V(s'))/\lambda)) \\
&= \lim_{\lambda \downarrow 0} V_{\min} - c_r\lambda - \lambda \log(\sum_{s'} P^o_{s,a}(s') \exp((V_{\min} - V(s'))/\lambda)) \\
&= \lim_{\lambda \downarrow 0} V_{\min} - c_r\lambda - \lambda \log(\sum_{s':V(s')=V_{\min}} P^o_{s,a}(s') + \sum_{s':V(s')>V_{\min}} P^o_{s,a}(s') \exp((V_{\min} - V(s'))/\lambda)) \\
&\overset{(a)}{=} \lim_{\lambda \downarrow 0} V_{\min} - c_r\lambda - \lambda \log(\sum_{s':V(s')=V_{\min}} P^o_{s,a}(s') + \mathcal{O}(\exp(-t/\lambda))) \\
&\overset{(b)}{=} \lim_{\lambda \downarrow 0} V_{\min} - c_r\lambda - \lambda \log(\sum_{s':V(s')=V_{\min}} P^o_{s,a}(s')) - \lambda \log(1 + \mathcal{O}(\exp(-t/\lambda))) \\
&\overset{(c)}{=} \lim_{\lambda \downarrow 0} V_{\min} - \lambda(c_r + \log(\sum_{s':V(s')=V_{\min}} P^o_{s,a}(s'))) - \mathcal{O}(\lambda \exp(-t/\lambda)) = V_{\min},
\end{aligned}
$$

where (a) follows by taking $t = \min_{s':V(s')>V_{\min}} V(s') - V_{\min} > 0$, and (b) and (c) follows from the Taylor series expansion. Thus when λ^* is 0, we have $\sigma_{\mathcal{P}^{\text{kl}}_{s,a}}(V) = V_{\min}$. A similar argument applies for $\sigma_{\widehat{\mathcal{P}}^{\text{kl}}_{s,a}}(V)$.

Now consider the case when $\lambda^* = 0$. From Lemma 22, it follows that, with probability at least $1 - \delta/(4|\mathcal{S}||\mathcal{A}|)$, $\widehat{\lambda}^* = 0$ for $N \geq N'(\delta/(4|\mathcal{S}||\mathcal{A}|), c_r, P^o_{s,a})$, where N' is defined in Lemma 22. Thus, whenever $\lambda^* = 0$, we have $|\sigma_{\widehat{\mathcal{P}}_{s,a}}(V) - \sigma_{\mathcal{P}_{s,a}}(V)| = |V_{\min} - V_{\min}| = 0$, with probability at least $1 - \delta/(4|\mathcal{S}||\mathcal{A}|)$. Thus having resolving this trivial case, we now focus on the case when $\lambda^* > 0$.

Consider the case when $\lambda^* > 0$. Let $\lambda_{\text{kl}} := \lambda^*/2 > 0$ (dependent on $P^o_{s,a}, V$, and c_r but independent of N). Again from Lemma 22, if $\lambda^* \in [\lambda_{\text{kl}}, 1/(c_r(1-\gamma))]$, then with probability at least $1 - \delta/(4|\mathcal{S}||\mathcal{A}|)$ we have $\widehat{\lambda}^* \in [\lambda_{\text{kl}}, 1/(c_r(1-\gamma))]$ for $N \geq N''(\delta/(4|\mathcal{S}||\mathcal{A}|), c_r, P^o_{s,a})$, where N'' is defined in Lemma 22.

From these arguments, it is clear that we can restrict the optimization problem (B.29) to the set $\lambda \in [\lambda_{\text{kl}}, 1/(c_r(1-\gamma))$. Using this, with the additional fact that $|\max_x f(x) - \max_x g(x)| \leq \max_x |f(x) - g(x)|$, we get

$$|\sigma_{\widehat{\mathcal{P}}^{\text{kl}}_{s,a}}(V) - \sigma_{\mathcal{P}^{\text{kl}}_{s,a}}(V)| \leq \max_{\lambda \in [\lambda_{\text{kl}}, \frac{1}{c_r(1-\gamma)}]} |\lambda \log(\frac{\widehat{P}_{s,a} \exp(-V/\lambda)}{P^o_{s,a} \exp(-V/\lambda)})|. \tag{B.30}$$

Now,

$$\left|\log(\frac{\widehat{P}_{s,a} \exp(-V/\lambda)}{P^o_{s,a} \exp(-V/\lambda)})\right| = \left|\log(1 + \frac{(\widehat{P}_{s,a} - P^o_{s,a}) \exp(-V/\lambda)}{P^o_{s,a} \exp(-V/\lambda)})\right| \leq \frac{|(P^o_{s,a} - \widehat{P}_{s,a}) \exp(-V/\lambda)|}{|P^o_{s,a} \exp(-V/\lambda)|}$$
$$\overset{(d)}{\leq} \frac{|(P^o_{s,a} - \widehat{P}_{s,a}) \exp(-V/\lambda)|}{\exp(\frac{-1}{\lambda_{\text{kl}}(1-\gamma)})}, \tag{B.31}$$

where (d) follows since $\lambda \geq \lambda_{\text{kl}}$ and $\|V\| \leq 1/(1-\gamma)$. Using (B.31) in (B.30) along with the fact that $\lambda \leq 1/(c_r(1-\gamma))$, we get the desired result. □

Proof of Theorem 6. The basic steps of the proof is similar to that of Theorem 4. So, we present only the important steps.

Following the same steps as given before (B.8) and (B.12) , we get

$$\|V^* - \widehat{V}^*\| + \|V^{\pi_k} - \widehat{V}^{\pi_k}\| \leq \frac{2\gamma}{(1-\gamma)} \max_{V \in \mathcal{V}} \max_{s,a} |\sigma_{\widehat{\mathcal{P}}^{\text{kl}}_{s,a}}(V) - \sigma_{\mathcal{P}^{\text{kl}}_{s,a}}(V)|. \tag{B.32}$$

Similarly, following the steps as given before (B.11), we get

$$\|\widehat{V}^{\pi_k} - \widehat{V}^*\| \leq \frac{2\gamma^{k+1}}{(1-\gamma)^2}. \tag{B.33}$$

Using Lemma 23 in (B.32), we get

$$\|V^* - \widehat{V}^*\| + \|V^{\pi_k} - \widehat{V}^{\pi_k}\| \leq \frac{2\gamma}{(1-\gamma)} \frac{\exp(1/(\lambda_{\text{kl}}(1-\gamma)))}{c_r(1-\gamma)} \max_{s,a} \max_{V \in \mathcal{V}} \max_{\lambda \in [\lambda_{\text{kl}}, \frac{1}{c_r(1-\gamma)}]} |(P^o_{s,a} - \widehat{P}_{s,a}) \exp(-V/\lambda)| \tag{B.34}$$

We now bound the max term in (B.34). We reparameterize $1/\lambda$ as θ and consider the set $\Theta = [c_r(1-\gamma), \frac{1}{\lambda_{\text{kl}}}]$. Also, consider the minimal η-cover $\mathcal{N}_\Theta(\eta)$ of Θ and fix a $V \in \mathcal{V}$. Then, for any given $\theta \in \Theta$, there exits a $\theta' \in \mathcal{N}_\Theta(\eta)$ such that $|\theta - \theta'| \leq \eta$. Now, for this particular θ, θ',

$$\begin{aligned}
&|(P^o_{s,a} - \widehat{P}_{s,a}) \exp(-V\theta)| = |(\widehat{P}_{s,a} - P^o_{s,a})(\exp(-V\theta') \circ \exp(-V(\theta - \theta'))| \\
&\overset{(c)}{\leq} |(\widehat{P}_{s,a} - P^o_{s,a}) \exp(-V\theta')| \exp(\eta/(1-\gamma)) \leq \max_{s,a} \max_{\theta' \in \mathcal{N}_\Theta(\eta)} |(\widehat{P}_{s,a} - P^o_{s,a}) \exp(-V\theta')| \exp(\eta/(1-\gamma)),
\end{aligned}$$

where (c) follows because V is non-negative and $\|V\| \leq 1/(1-\gamma)$. Now consider a minimal η-cover $\mathcal{N}_\mathcal{V}(\eta)$ of the set $\mathcal{V}$. By definition, there exists $V' \in \mathcal{N}_\mathcal{V}(\eta)$ such that $\|V - V'\| \leq \eta$. So, we get

$$\begin{aligned}
|(P^o_{s,a} - \widehat{P}_{s,a}) \exp(-V\theta)| &\leq |(\widehat{P}_{s,a} - P^o_{s,a}) \exp(-V\theta')| \exp(\eta/(1-\gamma)) \\
&= |(\widehat{P}_{s,a} - P^o_{s,a})(\exp(-V'\theta') \circ \exp(\theta'(V' - V)))| \exp(\eta/(1-\gamma)) \\
&\overset{(d)}{\leq} |(\widehat{P}_{s,a} - P^o_{s,a})(\exp(-V'\theta'))| \exp(\eta/(1-\gamma)) \exp(\eta/\lambda_{\text{kl}}) \\
&\leq \max_{s,a} \max_{V' \in \mathcal{V}} \max_{\theta' \in \mathcal{N}_\Theta(\eta)} |(\widehat{P}_{s,a} - P^o_{s,a})(\exp(-V'\theta'))| \exp(\eta/(1-\gamma)) \exp(\eta/\lambda_{\text{kl}})
\end{aligned}$$

where (d) follows because $\theta' \in \mathcal{N}_\Theta(\eta) \subseteq \Theta$. Now, taking maximum on both sides with respect to (s,a), θ, and V, we get

$$\begin{aligned}
\max_{s,a} \max_{\theta \in \Theta} \max_{V \in \mathcal{V}} |(\widehat{P}_{s,a} - P^o_{s,a}) \exp(-V\theta)| &\leq \exp(\eta/(1-\gamma)) \exp(\eta/\lambda_{\text{kl}}) \max_{s,a} \max_{V' \in \mathcal{V}} \max_{\theta' \in \mathcal{N}_\Theta(\eta)} |(\widehat{P}_{s,a} - P^o_{s,a}) \exp(-V'\theta')| \\
&\overset{(e)}{\leq} \exp(\eta/(1-\gamma)) \exp(\eta/\lambda_{\text{kl}}) \sqrt{\frac{\log(2|\mathcal{S}||\mathcal{A}||\mathcal{N}_\Theta(\eta)||\mathcal{N}_\mathcal{V}(\eta)|/\delta)}{2N}}
\end{aligned}$$

$$\overset{(f)}{\leq} \exp(\eta/(1-\gamma))\exp(\eta/\lambda_{\text{kl}})\sqrt{\frac{|\mathcal{S}|\log(18|\mathcal{S}||\mathcal{A}|/(\delta\eta^2(1-\gamma)\lambda_{\text{kl}}))}{2N}} \tag{B.35}$$

with probability greater than $1-\delta$. Here, (e) follows from Lemma 19 with a union bound accounting for $|\mathcal{N}_\Theta(\eta)|$, $|\mathcal{N}_\mathcal{V}(\eta)|$ and the fact that $\|\exp(-V'\theta')\| \leq 1$, and (f) follows from Lemmas 17 and 18.

Using (B.32) - (B.35), we get, with probability greater than $1-\delta$,

$$\begin{aligned}\|V^* - V^{\pi_k}\| \leq & \frac{2\gamma^{k+1}}{(1-\gamma)^2} + \\ & \frac{2\gamma}{(1-\gamma)}\frac{\exp(1/(\lambda_{\text{kl}}(1-\gamma)))}{c_r(1-\gamma)}\exp(\eta/(1-\gamma))\exp(\eta/\lambda_{\text{kl}})\sqrt{\frac{|\mathcal{S}|\log(18|\mathcal{S}||\mathcal{A}|/(\delta\eta^2(1-\gamma)\lambda_{\text{kl}}))}{2N}}.\end{aligned}$$

We can now choose k, ε, η to make each of the term on the RHS of the above inequality small. In particular, choosing $\eta = 1$, $\varepsilon \in (0, 1/(1-\gamma))$, and k, N satisfying the conditions

$$\begin{aligned} & k \geq K_0 = \frac{1}{\log(1/\gamma)} \cdot \log(\frac{4}{\varepsilon(1-\gamma)^2}) \qquad \text{and} \\ & N \geq N^{\text{kl}} = \max\Bigg\{ \max_{s,a} N'(\delta/(4|\mathcal{S}||\mathcal{A}|), c_r, P^o_{s,a}),\ \max_{s,a} N''(\delta/(4|\mathcal{S}||\mathcal{A}|), c_r, P^o_{s,a}), \\ & \qquad \frac{8\gamma^2|\mathcal{S}|}{c_r^2(1-\gamma)^4\varepsilon^2}\exp(\frac{4+2\lambda_{\text{kl}}}{\lambda_{\text{kl}}(1-\gamma)})\log(\frac{18|\mathcal{S}||\mathcal{A}|}{\delta\lambda_{\text{kl}}(1-\gamma)})\Bigg\}, \end{aligned}$$

we get $\|V^* - V^{\pi_k}\| \leq \varepsilon$ with probability greater than $1-\delta$. □

B.2.5 Proof of Theorem 7

Proof. We consider the deterministic MDP $(\mathcal{S}, \mathcal{A}, r, P^o, \gamma)$ shown in Fig.B.1 to be the nominal model. We fix $\gamma \in (0.01, 1]$ and $s_1 = 0$. The state space is $\mathcal{S} = \{0, 1\}$ and action space is $\mathcal{A} = \{a_l, a_r\}$, where a_l denotes 'move left' and a_r denotes 'move right' action. Reward for state 1 and action a_r pair is $r(1, a_r) = 1$, for state 0 and action a_r pair is $r(0, a_r) = -100\gamma/99$, and the reward is 0 for all other (s, a). Transition function P^o is deterministic, as indicated by the arrows.

Similarly, we consider another deterministic model P', as shown in Fig.B.2. We consider the

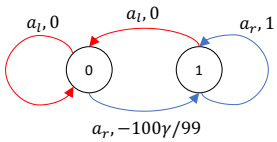

Figure B.1: Transitions and rewards corresponding to the nominal model P^o. The states $\{0, 1\}$ are given inside the circles, and the actions $\{a_l, a_r\}$ and associated rewards are given on the corresponding transitions.

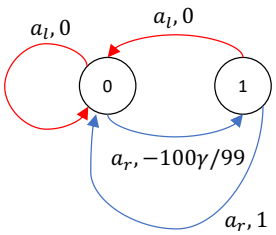

Figure B.2: Transitions and rewards corresponding to the model P'.

set $\mathcal{P} = \{P^o, P'\}$.

It is straight forward to show that taking action a_r in any state is the optimal non-robust policy π^o corresponding to the nominal model P^o. This is obvious if for state $s = 1$. For $s = 0$, notice that taking action a_l will give a value zero and taking action a_r will give a value $\frac{\gamma}{1-\gamma} - \frac{100\gamma}{99}$. Since $\gamma > 0.01$, taking action a_r will give a positive value and hence is optimal. So, we get

$$V_{\pi^o, P^o}(0) = \frac{\gamma}{1-\gamma} - \frac{100\gamma}{99}.$$

We can now compute $V_{\pi^o, P'}(0)$ using the recursive equation

$$V_{\pi^o, P'}(0) = -\frac{100\gamma}{99} + \gamma + \gamma^2 V_{\pi^o, P'}(0).$$

Solving this, we get $V_{\pi^o, P'}(0) = -\gamma/(99(1-\gamma^2))$.

Now the robust value of π^o is given by

$$V^{\pi^o}(0) = \min\{V_{\pi^o, P^o}(0), V_{\pi^o, P'}(0)\} = -\gamma/(99(1-\gamma^2)).$$

We will now compute the optimal non-robust value from state 0 of model P'.

$$\max_{\pi} V_{\pi, P'}(0) = \max\{V_{(\pi(0)=a_r, \pi(1)=a_r), P'}(0),\ V_{(\pi(0)=a_l, \pi(1)=a_l), P'}(0),$$
$$V_{(\pi(0)=a_r, \pi(1)=a_l), P'}(0),\ V_{(\pi(0)=a_l, \pi(1)=a_r), P'}(0)\}$$

$$= \max\{ -\frac{\gamma}{99(1-\gamma^2)},\ 0,\ -\frac{100\gamma}{99(1+\gamma^2)},\ 0\} = 0.$$

Now, we find the optimal robust value $V^*(0)$. From the perfect duality result of robust MDP [34, Theorem 1], we have

$$V^*(0) = \min\{\max_{\pi} V_{\pi,P^o}(0), \max_{\pi} V_{\pi,P'}(0)\} = \min\{V_{\pi^o,P^o}(0), \max_{\pi} V_{\pi,P'}(0)\} = 0.$$

We finally have

$$V^*(0) - V^{\pi^o}(0) = \frac{\gamma}{99(1-\gamma^2)} \geq \frac{\gamma}{198(1-\gamma)},$$

where the inequality follows since $1+\gamma \leq 2$. Thus, setting $c = \gamma/198$ and $\gamma_o = 0.01$, completes the proof of this theorem. □

APPENDIX C

APPENDIX FOR CHAPTER 4*

In this appendix, we include complete proofs, all experiments, and all supporting details for the corresponding chapter.

C.1 Useful Technical Results

In this section, we state some existing results from concentration inequalities, generalization bounds, and optimization theory that we will use later in our analysis. We first state the Berstein's inequality that utilizes second-moment to get a tighter concentration inequality.

Lemma 24 (Bernstein's inequality [164, Theorem 2.8.4]). *Let $X_1, \cdots, X_T$ be independent random variables. Assume that $|X_t - \mathbb{E}[X_t]| \leq M$, for all t. Then, for any $\varepsilon > 0$, we have*

$$\mathbb{P}\left(\left|\frac{1}{T}\sum_{t=1}^{T}(X_t - \mathbb{E}[X_t])\right| \geq \varepsilon\right) \leq 2\exp\left(-\frac{T^2\varepsilon^2}{2\sigma^2 + \frac{2MT\varepsilon}{3}}\right),$$

where $\sigma^2 = \sum_{t=1}^{T}\mathbb{E}[X_t^2]$. Furthermore, if $X_1, \cdots, X_T$ are independent and identically distributed random variables, then for any $\delta \in (0,1)$, we have

$$\left|\mathbb{E}[X_1] - \frac{1}{T}\sum_{t=1}^{T}X_t\right| \leq \sqrt{\frac{2\mathbb{E}[X_1^2]\log(2/\delta)}{T}} + \frac{M\log(2/\delta)}{3T},$$

with probability at least $1-\delta$.

We now state a result for the generalization bounds on empirical risk minimization (ERM) problems. This result is adapted from [167, Theorem 26.5, Lemma 26.8, Lemma 26.9].

Lemma 25 (ERM generalization bound). *Let P be the data generating distribution on the space $\mathcal{X}$ and let $\mathcal{H}$ be a given hypothesis class of functions. Assume that for all $x \in \mathcal{X}$ and $h \in \mathcal{H}$ we have*

*Reprinted with permission from Kishan Panaganti, Zaiyan Xu, Dileep Kalathil, Mohammad Ghavamzadeh, "Robust Reinforcement Learning using Offline Data." Neural Information Processing Systems. PMLR, 2022.

that $|l(h,x)| \leq c_1$ *for some positive constant* $c_1 > 0$*. Given a dataset* $\mathcal{D} = \{X_i\}_{i=1}^N$*, generated independently from* P*, denote* $\hat{h}$ *as the ERM solution, i.e.* $\hat{h} = \operatorname{argmin}_{h\in\mathcal{H}}(1/N)\sum_{i=1}^N l(h,X_i)$*. For any fixed* $\delta \in (0,1)$ *and* $h^* \in \operatorname{argmin}_{h\in\mathcal{H}} \mathbb{E}_{X\sim P}[l(h,X)]$*, we have*

$$\mathbb{E}_{X\sim P}[l(\hat{h},X)] - \mathbb{E}_{X\sim P}[l(h^*,X)] \leq 2R(l\circ\mathcal{H}\circ\mathcal{D}) + 5c_1\sqrt{\frac{2\log(8/\delta)}{N}}, \tag{C.1}$$

with probability at least $1-\delta$*, where* $R(\cdot)$ *is the Rademacher complexity of* $l\circ\mathcal{H}$ *given by*

$$R(l\circ\mathcal{H}\circ\mathcal{D}) = \frac{1}{N}\mathbb{E}_{\{\sigma_i\}_{i=1}^N}\left(\sup_{g\in l\circ\mathcal{H}}\sum_{i=1}^N \sigma_i g(X_i)\right),$$

in which σ_i*'s are independent from* X_i*'s and are independently and identically distributed according to the Rademacher random variable* σ*, i.e.* $\mathbb{P}(\sigma=1) = 0.5 = \mathbb{P}(\sigma=-1)$*.*

Furthermore, if $\mathcal{H}$ *is a finite hypothesis class, i.e.* $|\mathcal{H}| < \infty$*, with* $|h\circ x| \leq c_2$ *for all* $h\in\mathcal{H}$ *and* $x\in\mathcal{X}$*, and* $l(h,x)$ *is* c_3*-Lipschitz in* h*, then we have*

$$\mathbb{E}_{X\sim P}[l(\hat{h},X)] - \mathbb{E}_{X\sim P}[l(h^*,X)] \leq 2c_2c_3\sqrt{\frac{2\log(|\mathcal{H}|)}{N}} + 5c_1\sqrt{\frac{2\log(8/\delta)}{N}}, \tag{C.2}$$

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

We now mention two important concepts from variational analysis [107] literature that is useful to relate minimization of integrals and the integrals of pointwise minimization under special class of functions.

Definition 3 ([107, Definition 14.59, Example 14.29] Decomposable spaces and Normal integrands)**.** *A space* $\mathcal{X}$ *of measurable functions is a decomposable space relative to an underlying measure space* $(\Omega,\mathcal{A},\mu)$*, if for every function* $x_0\in\mathcal{X}$*, every set* $A\in\mathcal{A}$ *with* $\mu(A)<\infty$*, and any bounded measurable function* $x_1 : A\to\mathbb{R}$*, the function* $x(\omega) = x_0(\omega)\mathbb{1}(\omega\notin A) + x_1(\omega)\mathbb{1}(\omega\in A)$ *belongs to* $\mathcal{X}$*. A function* $f : \Omega\times\mathbb{R}\to\mathbb{R}$ *(finite-valued) is a normal integrand, if and only if* $f(\omega,x)$ *is* $\mathcal{A}$*-measurable in* ω *for each* x *and is continuous in* x *for each* ω*.*

Remark 13. *A few examples of decomposable spaces are $L^p(\mathcal{S}\times\mathcal{A},\Sigma(\mathcal{S}\times\mathcal{A}),\mu)$ for any $p\geq 1$ and $\mathcal{M}(\mathcal{S}\times\mathcal{A},\Sigma(\mathcal{S}\times\mathcal{A}))$, the space of all $\Sigma(\mathcal{S}\times\mathcal{A})$-measurable functions.*

Lemma 26 ([107, Theorem 14.60])**.** *Let $\mathcal{X}$ be a space of measurable functions from Ω to $\mathbb{R}$ that is decomposable relative to a σ-finite measure μ on the σ-algebra $\mathcal{A}$. Let $f:\Omega\times\mathbb{R}\to\mathbb{R}$ (finite-valued) be a normal integrand. Then, we have*

$$\inf_{x\in\mathcal{X}}\int_{\omega\in\Omega} f(\omega,x(\omega))\mu(\mathrm{d}\omega)=\int_{\omega\in\Omega}\left(\inf_{x\in\mathbb{R}} f(\omega,x)\right)\mu(\mathrm{d}\omega).$$

Moreover, as long as the above infimum is not $-\infty$, we have that

$$x'\in\operatorname*{argmin}_{x\in\mathcal{X}}\int_{\omega\in\Omega} f(\omega,x(\omega))\mu(\mathrm{d}\omega),$$

if and only if $x'(\omega)\in\operatorname{argmin}_{x\in\mathbb{R}} f(\omega,x)\cdot\mu$ almost surely.

We now give one result from distributionally robust optimization. The f-divergence between the distributions P and P^o is defined as

$$D_f(P\|P^o)=\int f(\frac{\mathrm{d}P}{\mathrm{d}P^o})\mathrm{d}P^o, \tag{C.3}$$

where f is a convex function [168, 169]. We obtain different divergences for different forms of the function f, including some well-known divergences. For example, $f(t)=|t-1|/2$ gives Total Variation (TV), $f(t)=t\log t$ gives Kullback-Liebler (KL), $f(t)=(t-1)^2$ gives Chi-square, and $f(t)=(\sqrt{t}-1)^2$ gives squared Hellinger divergences.

Let P^o be a distribution on the space $\mathcal{X}$ and let $l:\mathcal{X}\to\mathbb{R}$ be a loss function. We have the following result from the *distributionally robust optimization* literature, see e.g., [109, Proposition 1] and [108, Section 3.2].

Proposition 12. *Let D_f be the f-divergence as defined in* (C.3)*. Then,*

$$\sup_{D_f(P\|P^o)\leq\rho} \mathbb{E}_P\left[l(X)\right] = \inf_{\lambda>0,\eta\in\mathbb{R}} \ \mathbb{E}_{P^o}\left[\lambda f^*\left(\frac{l(X)-\eta}{\lambda}\right)\right] + \lambda\rho + \eta, \tag{C.4}$$

where $f^*(s) = \sup_{t\geq 0}\{st - f(t)\}$ is the Fenchel conjugate.

Note that on the right hand side of (C.4), the expectation is taken only with respect to P^o. We will use the above result to derive the dual reformulation of the robust Bellman operator.

C.2 Proof of the Proposition 6

As the first step, we adapt the result given in Proposition 12 in two ways: (i) Since Proposition 6 considers the TV uncertainty set, we will derive the specific form of this result for the TV uncertainty set, (ii) Since Proposition 6 considers the minimization problem instead of the maximization problem, unlike in Proposition 12, we will derive the specific form of this result for minimization.

Lemma 27. *Let D_f be as defined in* (C.3) *with $f(t) = |t-1|/2$ corresponding to the TV uncertainty set. Then,*

$$\inf_{D_f(P\|P^o)\leq\rho} \mathbb{E}_P[l(X)] = -\inf_{\eta\in\mathbb{R}} \ \mathbb{E}_{P^o}[(\eta - l(X))_+] + (\eta - \inf_{x\in\mathcal{X}} l(x))_+ \times \rho - \eta,$$

Proof. First, we will compute the Fenchel conjugate of $f(t) = |t-1|/2$. We have

$$f^*(s) = \sup_{t\geq 0}\left\{st - \frac{1}{2}|t-1|\right\} = \max\left\{\sup_{t\in[0,1]}\{(s+\frac{1}{2})t - \frac{1}{2}\}\ ,\ \sup_{t>1}\{(s-\frac{1}{2})t + \frac{1}{2}\}\right\}.$$

It is easy to see that for $s > 1/2$, we have $f^*(s) = +\infty$, and for $s \leq -1/2$, we have $f^*(s) = -1/2$. For $s \in [-1/2, 1/2]$, we have

$$\begin{aligned} f^*(s) &= \max\Big\{\sup_{t\in[0,1]}\{(s+\frac{1}{2})t - \frac{1}{2}\}\ ,\ \sup_{t>1}(\{(s-\frac{1}{2})t + \frac{1}{2}\}\Big\} \\ &= \max\left\{((s+\frac{1}{2})\cdot 1 - \frac{1}{2}),\ ((s-\frac{1}{2})\cdot 1 + \frac{1}{2})\right\} = s. \end{aligned}$$

Thus, we have

$$f^*(s) = \begin{cases} -\frac{1}{2} & s \leq -\frac{1}{2}, \\ s & s \in [-\frac{1}{2}, \frac{1}{2}] \\ +\infty & s > \frac{1}{2}. \end{cases}.$$

From Proposition 12, we obtain

$$\begin{aligned}
\sup_{D_f(P\|P^o)\leq\rho} \mathbb{E}_P[l(X)] &= \inf_{\lambda>0,\eta\in\mathbb{R}} \mathbb{E}_{P^o}[\lambda f^*(\frac{l(X)-\eta}{\lambda})] + \lambda\rho + \eta \\
&= \inf_{\lambda,\eta:\lambda>0,\eta\in\mathbb{R},\frac{\sup_{x\in\mathcal{X}} l(x)-\eta}{\lambda}\leq\frac{1}{2}} \mathbb{E}_{P^o}[\lambda \max\{\frac{l(X)-\eta}{\lambda}, -\frac{1}{2}\}] + \lambda\rho + \eta \\
&= \inf_{\lambda,\eta:\lambda>0,\eta\in\mathbb{R},\frac{\sup_{x\in\mathcal{X}} l(x)-\eta}{\lambda}\leq\frac{1}{2}} \mathbb{E}_{P^o}[\max\{l(X)-\eta, -\lambda/2\}] + \lambda\rho + \eta \\
&= \inf_{\lambda,\eta:\lambda>0,\eta\in\mathbb{R},\frac{\sup_{x\in\mathcal{X}} l(x)-\eta}{\lambda}\leq\frac{1}{2}} \mathbb{E}_{P^o}[(l(X)-\eta+\lambda/2)_+] - \lambda/2 + \lambda\rho + \eta \\
&= \inf_{\lambda,\eta':\lambda>0,\eta'\in\mathbb{R},\frac{\sup_{x\in\mathcal{X}} l(x)-\eta'}{\lambda}\leq 1} \mathbb{E}_{P^o}[(l(X)-\eta')_+] + \lambda\rho + \eta'.
\end{aligned}$$

The second equality follows since $f^*(\frac{l(X)-\eta}{\lambda}) = +\infty$ whenever $\frac{l(X)-\eta}{\lambda} > \frac{1}{2}$, which can be ignored as we are minimizing over λ and η. The fourth equality follows form the fact that $\max\{x, y\} = (x-y)_+ + y$ for any $x, y \in \mathbb{R}$. Finally, the last equality follows by making the substitution $\eta' = \eta - \lambda/2$. Taking the optimal value of λ, i.e., $\lambda = (\sup_{x\in\mathcal{X}} l(x) - \eta')_+$, we get

$$\sup_{D_f(P\|P^o)\leq\rho} \mathbb{E}_P[l(X)] = \inf_{\eta\in\mathbb{R}} \mathbb{E}_{P^o}[(l(X)-\eta)_+] + (\sup_{x\in\mathcal{X}} l(x) - \eta)_+\rho + \eta.$$

Now,

$$\begin{aligned}
\inf_{D_f(P\|P^o)\leq\rho} \mathbb{E}_P[l(X)] &= -\sup_{D_f(P\|P^o)\leq\rho} \mathbb{E}_P[-l(X)] \\
&= -\inf_{\eta\in\mathbb{R}} \mathbb{E}_{P^o}[(-l(X)-\eta)_+] + (\sup_{x\in\mathcal{X}} -l(x) - \eta)_+\rho + \eta \\
&= -\inf_{\eta'\in\mathbb{R}} \mathbb{E}_{P^o}[(\eta' - l(X))_+] + (\eta' - \inf_{x\in\mathcal{X}} l(x))_+\rho - \eta',
\end{aligned}$$

which completes the proof. □

We are now ready to prove Proposition 6.

Proof of Proposition 6.

For each (s, a), the optimization problem in (4.3) is given by $\min_{P_{s,a} \in \mathcal{P}_{s,a}} \mathbb{E}_{s' \sim P_{s,a}}[V(s')]$, and our focus is on the setting where $\mathcal{P}_{s,a}$ is given by the TV uncertainty set. So, $\mathcal{P}_{s,a}$ can be equivalently defined using the f-divergence with $f(t) = |t-1|/2$ as $\mathcal{P}_{s,a} = \{P_{s,a} : D_f(P_{s,a}||P^o_{s,a}) \leq \rho\}$. We can now use the result of Lemma 27 to get

$$\inf_{P_{s,a} \in \mathcal{P}_{s,a}} \mathbb{E}_{s' \sim P_{s,a}}[V(s')] = - \inf_{\eta \in \mathbb{R}} \ \mathbb{E}_{s' \sim P^o_{s,a}}[(\eta - V(s'))_+] + (\eta - \inf_{s'' \in \mathcal{S}} V(s''))_+ \rho - \eta.$$

From Proposition 12, the function $h(\eta) = \mathbb{E}_{s' \sim P^o_{s,a}}[(\eta - V(s'))_+] + \rho(\eta - \inf_{s''} V(s''))_+ - \eta$ is convex in η. Now, solving $\mathrm{d}h(\eta)/\mathrm{d}\eta = 0$ yields

$$\mathbb{E}_{s' \sim P^o_{s,a}}[1_{[\eta^* > V(s')]}] + \rho \cdot 1_{[\eta^* > \inf_{s''} V(s'')]} = 1.$$

It now follows that we can limit our search space of η since the optimal $\eta^* \in [0, \frac{1}{(1-\gamma)}]$ according to the above equation.

Using these, we get

$$\begin{aligned}(TQ)(s,a) &= r(s,a) + \gamma \inf_{P_{s,a} \in \mathcal{P}_{s,a}} \mathbb{E}_{s' \sim P_{s,a}}[V(s')] \\ &= r(s,a) + \gamma \cdot -1 \cdot \inf_{\eta \in [0, \frac{1}{(1-\gamma)}]} \ \mathbb{E}_{s' \sim P^o_{s,a}}[(\eta - V(s'))_+] + (\eta - \inf_{s'' \in \mathcal{S}} V(s''))_+ \rho - \eta.\end{aligned}$$

This completes the proof of Proposition 6. □

C.3 Proof of Theorem 8

We start by proving Lemma 5 which mainly follows from Lemma 26 in Appendix C.1.

Proof of Lemma 5. Let $h((s,a),\eta) = \mathbb{E}_{s' \sim P^o_{s,a}}((\eta - \max_{a'} f(s',a'))_+ - (1-\rho)\eta)$. We note that $h((s,a),\eta)$ is $\Sigma(\mathcal{S}\times\mathcal{A})$-measurable in $(s,a) \in \mathcal{S}\times\mathcal{A}$ for each $\eta \in [0, 1/(1-\gamma)]$ and is continuous in η for each $(s,a) \in \mathcal{S}\times\mathcal{A}$. Now it follows that $h((s,a),\eta)$ is a normal integrand (see Definition 3 in Appendix C.1). We now note that $L^1(\mathcal{S}\times\mathcal{A}, \Sigma(\mathcal{S}\times\mathcal{A}), \mu)$ is a decomposable space (Remark 13 in Appendix C.1). Thus, this lemma now directly follows from Lemma 26. □

Now we state a result and provide its proof for the empirical risk minimization on the dual parameter.

Lemma 28 (Dual Optimization Error Bound)**.** *Let $\widehat{g}_f$ be the dual optimization parameter from the algorithm (Step 4) for the state-action value function f and let T_g be as defined in (4.9). With probability at least $1-\delta$, we have*

$$\sup_{f\in\mathcal{F}} \|Tf - T_{\widehat{g}_f} f\|_{1,\mu} \leq \frac{2\gamma(2-\rho)}{(1-\gamma)}\sqrt{\frac{2\log(|\mathcal{G}|)}{N}} + \frac{15\gamma}{(1-\gamma)}\sqrt{\frac{2\log(8|\mathcal{F}|/\delta)}{N}} + \gamma\varepsilon_{\textit{dual}}.$$

Proof. Fix an $f \in \mathcal{F}$. We will also invoke union bound for the supremum here. We recall from (4.8) that $\widehat{g}_f = \operatorname{argmin}_{g\in\mathcal{G}} \widehat{L}_{\mathrm{dual}}(g; f)$. From the robust Bellman equation, we directly obtain

$$\begin{aligned}
\|T_{\widehat{g}_f} f - Tf\|_{1,\mu} &= \gamma(\mathbb{E}_{s,a\sim\mu}|\mathbb{E}_{s'\sim P^o_{s,a}}((\widehat{g}_f(s,a) - \max_{a'} f(s',a'))_+ - (1-\rho)\widehat{g}_f(s,a)) \\
&\qquad - \inf_{\eta\in[0,1/(1-\gamma)]} \mathbb{E}_{s'\sim P^o_{s,a}}((\eta - \max_{a'} f(s',a'))_+ - (1-\rho)\eta)|) \\
&\overset{(a)}{=} \gamma(\mathbb{E}_{s,a\sim\mu}\mathbb{E}_{s'\sim P^o_{s,a}}((\widehat{g}_f(s,a) - \max_{a'} f(s',a'))_+ - (1-\rho)\widehat{g}_f(s,a)) \\
&\qquad - \mathbb{E}_{s,a\sim\mu}[\inf_{\eta\in[0,1/(1-\gamma)]} \mathbb{E}_{s'\sim P^o_{s,a}}((\eta - \max_{a'} f(s',a'))_+ - (1-\rho)\eta)]) \\
&\overset{(b)}{=} \gamma(\mathbb{E}_{s,a\sim\mu,s'\sim P^o_{s,a}}((\widehat{g}_f(s,a) - \max_{a'} f(s',a'))_+ - (1-\rho)\widehat{g}_f(s,a)) \\
&\qquad - \inf_{g\in L^1} \mathbb{E}_{s,a\sim\mu,s'\sim P^o_{s,a}}((g(s,a) - \max_{a'} f(s',a'))_+ - (1-\rho)g(s,a))) \\
&= \gamma(\mathbb{E}_{s,a\sim\mu,s'\sim P^o_{s,a}}((\widehat{g}_f(s,a) - \max_{a'} f(s',a'))_+ - (1-\rho)\widehat{g}_f(s,a)) \\
&\qquad - \inf_{g\in\mathcal{G}} \mathbb{E}_{s,a\sim\mu,s'\sim P^o_{s,a}}((g(s,a) - \max_{a'} f(s',a'))_+ - (1-\rho)g(s,a))) \\
&\qquad + \gamma(\inf_{g\in\mathcal{G}} \mathbb{E}_{s,a\sim\mu,s'\sim P^o_{s,a}}((g(s,a) - \max_{a'} f(s',a'))_+ - (1-\rho)g(s,a))
\end{aligned}$$

$$
\begin{aligned}
&\quad - \inf_{g \in L^1} \mathbb{E}_{s,a\sim\mu,s'\sim P^o_{s,a}}((g(s,a) - \max_{a'} f(s',a'))_+ - (1-\rho)g(s,a))) \\
&\overset{(c)}{\leq} \gamma(\mathbb{E}_{s,a\sim\mu,s'\sim P^o_{s,a}}((\widehat{g}_f(s,a) - \max_{a'} f(s',a'))_+ - (1-\rho)\widehat{g}_f(s,a)) \\
&\quad - \inf_{g \in \mathcal{G}} \mathbb{E}_{s,a\sim\mu,s'\sim P^o_{s,a}}((g(s,a) - \max_{a'} f(s',a'))_+ - (1-\rho)g(s,a))) + \gamma\varepsilon_{\text{dual}} \\
&\overset{(d)}{\leq} 2\gamma R(l \circ \mathcal{G} \circ \mathcal{D}) + \frac{15\gamma}{(1-\gamma)}\sqrt{\frac{2\log(8/\delta)}{N}} + \gamma\varepsilon_{\text{dual}} \\
&\overset{(e)}{\leq} \frac{2\gamma(2-\rho)}{(1-\gamma)}\sqrt{\frac{2\log(|\mathcal{G}|)}{N}} + \frac{15\gamma}{(1-\gamma)}\sqrt{\frac{2\log(8/\delta)}{N}} + \gamma\varepsilon_{\text{dual}}.
\end{aligned}
$$

(a) follows since $\inf_g h(g) \leq h(\widehat{g}_f)$. (b) follows from Lemma 5. (c) follows from the approximate dual realizability assumption (Assumption 7).

For (d), we consider the loss function $l(g,(s,a,s')) = (g(s,a) - \max_{a'} f(s',a'))_+ - (1-\rho)g(s,a)$ and dataset $\mathcal{D} = \{s_i, a_i, s'_i\}_{i=1}^N$. Note that $|l(g,(s,a,s'))| \leq 3/(1-\gamma)$ (since $f \in \mathcal{F}$ and $g \in \mathcal{G}$). Now, we can apply the empirical risk minimization result (C.1) in Lemma 25 to get (d), where $R(\cdot)$ is the Rademacher complexity.

Finally, (e) follows from (C.2) in Lemma 25 when combined with the facts that $l(g,(s,a,s'))$ is $(2-\rho)$-Lipschitz in g and $g(s,a) \leq 1/(1-\gamma)$, since $g \in \mathcal{G}$.

With union bound, with probability at least $1-\delta$, we finally get

$$
\sup_{f\in\mathcal{F}} \|Tf - T_{\widehat{g}_f} f\|_{1,\mu} \leq \frac{2\gamma(2-\rho)}{(1-\gamma)}\sqrt{\frac{2\log(|\mathcal{G}|)}{N}} + \frac{15\gamma}{(1-\gamma)}\sqrt{\frac{2\log(8|\mathcal{F}|/\delta)}{N}} + \gamma\varepsilon_{\text{dual}},
$$

which concludes the proof. □

We next prove the least-squares generalization bound for the RFQI algorithm.

Lemma 29 (Least squares generalization bound)**.** *Let $\widehat{f}_g$ be the least-squares solution from the algorithm (Step 5) for the state-action value function f and dual variable function g. Let T_g be as defined in* (4.9)*. Then, with probability at least $1-\delta$, we have*

$$
\sup_{f\in\mathcal{F}} \sup_{g\in\mathcal{G}} \|T_g f - \widehat{f}_g\|_{2,\mu} \leq \sqrt{6\varepsilon_c} + \frac{10}{(1-\gamma)}\sqrt{\frac{18\log(2|\mathcal{F}||\mathcal{G}|/\delta)}{N}}.
$$

Proof. We adapt the least-squares generalization bound given in [106, Lemma A.11] to our setting. We recall from (4.10) that $\widehat{f}_g = \operatorname{argmin}_{Q\in\mathcal{F}} \widehat{L}_{\mathrm{RFQI}}(Q; f, g)$. We first fix functions $f \in \mathcal{F}$ and $g \in \mathcal{G}$. For any function $f' \in \mathcal{F}$, we define random variables $z_i^{f'}$ as

$$z_i^{f'} = \left(f'(s_i, a_i) - y_i\right)^2 - \left((T_g f)(s_i, a_i) - y_i\right)^2,$$

where $y_i = r_i - \gamma(g(s_i, a_i) - \max_{a'} f(s_i', a'))_+ + \gamma(1-\rho)g(s_i, a_i)$, and $(s_i, a_i, s_i') \in \mathcal{D}$ with $(s_i, a_i) \sim \mu$, $s_i' \sim P^o_{s_i,a_i}$. It is straightforward to note that for a given (s_i, a_i), we have $\mathbb{E}_{s_i'\sim P^o_{s_i,a_i}}[y_i] = (T_g f)(s_i, a_i)$.

Also, since $g(s_i, a_i) \le 1/(1-\gamma)$ (because $g \in \mathcal{G}$) and $f(s_i, a_i), f'(s_i, a_i) \le 1/(1-\gamma)$ (because $f, f' \in \mathcal{F}$), we have $(T_g f)(s_i, a_i) \le 3/(1-\gamma)$. This also gives us that $y_i \le 3/(1-\gamma)$.

Using this, we obtain the first moment and an upper-bound for the second moment of $z_i^{f'}$ as follows:

$$\begin{aligned}
\mathbb{E}_{s_i'\sim P^o_{s_i,a_i}}[z_i^{f'}] &= \mathbb{E}_{s_i'\sim P^o_{s_i,a_i}}[(f'(s_i, a_i) - (T_g f)(s_i, a_i)) \cdot (f'(s_i, a_i) + (T_g f)(s_i, a_i) - 2y_i)] \\
&= (f'(s_i, a_i) - (T_g f)(s_i, a_i))^2, \\
\mathbb{E}_{s_i'\sim P^o_{s_i,a_i}}[(z_i^{f'})^2] &= \mathbb{E}_{s_i'\sim P^o_{s_i,a_i}}[(f'(s_i, a_i) - (T_g f)(s_i, a_i))^2 \cdot (f'(s_i, a_i) + (T_g f)(s_i, a_i) - 2y_i)^2] \\
&= (f'(s_i, a_i) - (T_g f)(s_i, a_i))^2 \cdot \mathbb{E}_{s_i'\sim P^o_{s_i,a_i}}[(f'(s_i, a_i) + (T_g f)(s_i, a_i) - 2y_i)^2] \\
&\le C_1 (f'(s_i, a_i) - (T_g f)(s_i, a_i))^2,
\end{aligned}$$

where $C_1 = 10^2/(1-\gamma)^2$. This immediately implies that

$$\begin{aligned}
\mathbb{E}_{s_i,a_i\sim\mu, s_i'\sim P^o_{s_i,a_i}}[z_i^{f'}] &= \|T_g f - f'\|_{2,\mu}^2, \\
\mathbb{E}_{s_i,a_i\sim\mu, s_i'\sim P^o_{s_i,a_i}}[(z_i^{f'})^2] &\le C_1 \|T_g f - f'\|_{2,\mu}^2.
\end{aligned}$$

From these calculations, it is also straightforward to see that $|z_i^{f'} - \mathbb{E}_{s_i,a_i\sim\mu, s_i'\sim P^o_{s_i,a_i}}[z_i^{f'}]| \le 2C_1$ almost surely.

Now, using the Bernstein's inequality (Lemma 24), together with a union bound over all $f' \in$

$\mathcal{F}$, with probability at least $1-\delta$, we have

$$|\|T_g f - f'\|_{2,\mu}^2 - \frac{1}{N}\sum_{i=1}^{N} z_i^{f'}| \leq \sqrt{\frac{2C_1\|T_g f - f'\|_{2,\mu}^2 \log(2|\mathcal{F}|/\delta)}{N}} + \frac{2C_1\log(2|\mathcal{F}|/\delta)}{3N}, \quad \text{(C.5)}$$

for all $f' \in \mathcal{F}$. Setting $f' = \widehat{f}_g$, with probability at least $1-\delta/2$, we have

$$\|T_g f - \widehat{f}_g\|_{2,\mu}^2 \leq \frac{1}{N}\sum_{i=1}^{N} z_i^{\widehat{f}_g} + \sqrt{\frac{2C_1\|T_g f - \widehat{f}_g\|_{2,\mu}^2 \log(4|\mathcal{F}|/\delta)}{N}} + \frac{2C_1\log(4|\mathcal{F}|/\delta)}{3N}. \quad \text{(C.6)}$$

Now we upper-bound $(1/N)\sum_{i=1}^{N} z_i^{\widehat{f}_g}$ in the following. Consider a function $\widetilde{f} \in \operatorname{argmin}_{h\in\mathcal{F}} \|h - T_g f\|_{2,\mu}^2$. Note that $\widetilde{f}$ is independent of the dataset. We note that our earlier first and second moment calculations hold true for $\widetilde{f}$, replacing f', as well. Now, from (C.5) setting $f' = \widetilde{f}$, with probability at least $1-\delta/2$ we have

$$\frac{1}{N}\sum_{i=1}^{N} z_i^{\widetilde{f}} - \|T_g f - \widetilde{f}\|_{2,\mu}^2 \leq \sqrt{\frac{2C_1\|T_g f - \widetilde{f}\|_{2,\mu}^2 \log(4|\mathcal{F}|/\delta)}{N}} + \frac{2C_1\log(4|\mathcal{F}|/\delta)}{3N}. \quad \text{(C.7)}$$

Suppose $(1/N)\sum_{i=1}^{N} z_i^{\widetilde{f}} \geq 2C_1\log(4|\mathcal{F}|/\delta)/N$ holds, then from (C.7) we get

$$\frac{1}{N}\sum_{i=1}^{N} z_i^{\widetilde{f}} - \|T_g f - \widetilde{f}\|_{2,\mu}^2 \leq \sqrt{\|T_g f - \widetilde{f}\|_{2,\mu}^2 \cdot \frac{1}{N}\sum_{i=1}^{N} z_i^{\widetilde{f}}} + \frac{2C_1\log(4|\mathcal{F}|/\delta)}{N}. \quad \text{(C.8)}$$

We note the following algebra fact: Suppose $x^2 - ax + b \leq 0$ with $b > 0$ and $a^2 \geq 4b$, then we have $x \leq a$. Taking $x = (1/N)\sum_{i=1}^{N} z_i^{\widetilde{f}}$ in this fact, from (C.8) we get

$$\frac{1}{N}\sum_{i=1}^{N} z_i^{\widetilde{f}} \leq 3\|T_g f - \widetilde{f}\|_{2,\mu}^2 + \frac{4C_1\log(4|\mathcal{F}|/\delta)}{3N} \leq 3\|T_g f - \widetilde{f}\|_{2,\mu}^2 + \frac{2C_1\log(4|\mathcal{F}|/\delta)}{N}. \quad \text{(C.9)}$$

Now suppose $(1/N)\sum_{i=1}^{N} z_i^{\widetilde{f}} \leq 2C_1\log(4|\mathcal{F}|/\delta)/N$, then (C.9) holds immediately. Thus, (C.9) always holds with probability at least $1-\delta/2$. Furthermore, recall $\widetilde{f} \in \operatorname{argmin}_{h\in\mathcal{F}} \|h - T_g f\|_{2,\mu}^2$,

we have

$$\frac{1}{N}\sum_{i=1}^{N} z_i^{\widetilde{f}} \leq 3\|T_g f - \widetilde{f}\|_{2,\mu}^2 + \frac{2C_1 \log(4|\mathcal{F}|/\delta)}{N}$$
$$= 3\min_{h\in\mathcal{F}} \|h - T_g f\|_{2,\mu}^2 + \frac{2C_1 \log(4|\mathcal{F}|/\delta)}{N} \leq 3\varepsilon_{\mathrm{c}} + \frac{2C_1 \log(4|\mathcal{F}|/\delta)}{N}, \tag{C.10}$$

where the last inequality follows from the approximate robust Bellman completion assumption (Assumption 5).

We note that since $\widehat{f}_g$ is the least-squares regression solution, we know that $(1/N)\sum_{i=1}^{N} z_i^{\widehat{f}_g} \leq (1/N)\sum_{i=1}^{N} z_i^{\widetilde{f}}$. With this note in (C.10), from (C.6), with probability at least $1-\delta$, we have

$$\|T_g f - \widehat{f}_g\|_{2,\mu}^2 \leq 3\varepsilon_{\mathrm{c}} + \frac{2C_1 \log(4|\mathcal{F}|/\delta)}{N}$$
$$+ \sqrt{\frac{2C_1\|T_g f - \widehat{f}_g\|_{2,\mu}^2 \log(4|\mathcal{F}|/\delta)}{N}} + \frac{2C_1 \log(4|\mathcal{F}|/\delta)}{3N}$$
$$\leq 3\varepsilon_{\mathrm{c}} + \frac{3C_1 \log(4|\mathcal{F}|/\delta)}{N} + \sqrt{\frac{3C_1\|T_g f - \widehat{f}_g\|_{2,\mu}^2 \log(4|\mathcal{F}|/\delta)}{N}}.$$

From the earlier algebra fact, taking $x = \|T_g f - \widehat{f}_g\|_{2,\mu}^2$, with probability at least $1-\delta$, we have

$$\|T_g f - \widehat{f}_g\|_{2,\mu}^2 \leq 6\varepsilon_{\mathrm{c}} + \frac{9C_1 \log(4|\mathcal{F}|/\delta)}{N}.$$

From the fact $\sqrt{x+y} \leq \sqrt{x} + \sqrt{y}$, with probability at least $1-\delta$, we get

$$\|T_g f - \widehat{f}_g\|_{2,\mu} \leq \sqrt{6\varepsilon_{\mathrm{c}}} + \sqrt{\frac{9C_1 \log(4|\mathcal{F}|/\delta)}{N}}.$$

Using union bound for $f \in \mathcal{F}$ and $g \in \mathcal{G}$, with probability at least $1-\delta$, we finally obtain

$$\sup_{f\in\mathcal{F}}\sup_{g\in\mathcal{G}} \|T_g f - \widehat{f}_g\|_{2,\mu} \leq \sqrt{6\varepsilon_{\mathrm{c}}} + \sqrt{\frac{18C_1 \log(2|\mathcal{F}||\mathcal{G}|/\delta)}{N}},$$

which completes the least-squares generalization bound analysis. □

We are now ready to prove the main theorem.

Proof of Theorem 8. We let $V_k(s) = Q_k(s, \pi_k(s))$ for every $s \in \mathcal{S}$. Since π_k is the greedy policy w.r.t Q_k, we also have $V_k(s) = Q_k(s, \pi_k(s)) = \max_a Q_k(s, a)$. We recall that $V^* = V^{\pi^*}$ and $Q^* = Q^{\pi^*}$. We also recall from Section 4.2 that Q^{π^*} is a fixed-point of the robust Bellman operator T defined in (4.3). We also note that the same holds true for any stationary deterministic policy π from [33] that Q^π satisfies $Q^\pi(s,a) = r(s,a) + \gamma \min_{P_{s,a} \in \mathcal{P}_{s,a}} \mathbb{E}_{s' \sim P_{s,a}}[V^\pi(s')]$. We can now further use the dual form (4.5) under Assumption 6. We first characterize the performance decomposition between V^{π^*} and V^{π_K}. For a given $s_0 \in \mathcal{S}$, we observe that

$$
\begin{aligned}
&V^{\pi^*}(s_0) - V^{\pi_K}(s_0) = (V^{\pi^*}(s_0) - V_K(s_0)) - (V^{\pi_K}(s_0) - V_K(s_0)) \\
&= (Q^{\pi^*}(s_0, \pi^*(s_0)) - Q_K(s_0, \pi_K(s_0))) - (Q^{\pi_K}(s_0, \pi_K(s_0)) - Q_K(s_0, \pi_K(s_0))) \\
&\overset{(a)}{\leq} Q^{\pi^*}(s_0, \pi^*(s_0)) - Q_K(s_0, \pi^*(s_0)) + Q_K(s_0, \pi_K(s_0)) - Q^{\pi_K}(s_0, \pi_K(s_0)) \\
&= Q^{\pi^*}(s_0, \pi^*(s_0)) - Q_K(s_0, \pi^*(s_0)) + Q_K(s_0, \pi_K(s_0)) - Q^{\pi^*}(s_0, \pi_K(s_0)) \\
&\qquad + Q^{\pi^*}(s_0, \pi_K(s_0)) - Q^{\pi_K}(s_0, \pi_K(s_0)) \\
&\overset{(b)}{\leq} Q^{\pi^*}(s_0, \pi^*(s_0)) - Q_K(s_0, \pi^*(s_0)) + Q_K(s_0, \pi_K(s_0)) - Q^{\pi^*}(s_0, \pi_K(s_0)) \\
&\qquad + \gamma \sup_\eta (\mathbb{E}_{s_1 \sim P^o_{s_0, \pi_K(s_0)}}((\eta - V^{\pi_K}(s_1))_+ - (\eta - V^{\pi^*}(s_1))_+)) \\
&\overset{(c)}{\leq} |Q^{\pi^*}(s_0, \pi^*(s_0)) - Q_K(s_0, \pi^*(s_0))| + |Q^{\pi^*}(s_0, \pi_K(s_0)) - Q_K(s_0, \pi_K(s_0))| \\
&\qquad + \gamma \mathbb{E}_{s_1 \sim P^o_{s_0, \pi_K(s_0)}}(|V^{\pi^*}(s_1) - V^{\pi_K}(s_1)|).
\end{aligned}
$$

(a) follows from the fact that π_K is the greedy policy with respect to Q_K. (b) follows from the Bellman optimality equations and the fact $|\sup_x f(x) - \sup_x g(x)| \leq \sup_x |f(x) - g(x)|$. Finally, (c) follows from the facts $(x)_+ - (y)_+ \leq (x-y)_+$ and $(x)_+ \leq |x|$ for any $x, y \in \mathbb{R}$.

We now recall the initial state distribution d_0. Thus, we have

$$
\mathbb{E}_{s_0 \sim d_0}[V^{\pi^*}] - \mathbb{E}_{s_0 \sim d_0}[V^{\pi_K}] \leq
$$

$$\mathbb{E}_{s_0\sim d_0}\Big[|Q^{\pi^*}(s_0,\pi^*(s_0)) - Q_K(s_0,\pi^*(s_0))| + |Q^{\pi^*}(s_0,\pi_K(s_0)) - Q_K(s_0,\pi_K(s_0))| + \gamma\mathbb{E}_{s_1\sim P^o_{s_0,\pi_K(s_0)}}(|V^{\pi^*}(s_1) - V^{\pi_K}(s_1)|)\Big].$$

Since $V^{\pi^*}(s) \geq V^{\pi_K}(s)$ for any $s \in \mathcal{S}$, by telescoping we get

$$\mathbb{E}_{s_0\sim d_0}[V^{\pi^*}] - \mathbb{E}_{s_0\sim d_0}[V^{\pi_K}] \leq \sum_{h=0}^{\infty}\gamma^h \times \Big(\mathbb{E}_{s\sim d_{h,\pi_K}}[|Q^{\pi^*}(s,\pi^*(s)) - Q_K(s,\pi^*(s))| + |Q^{\pi^*}(s,\pi_K(s)) - Q_K(s,\pi_K(s))|]\Big), \quad \text{(C.11)}$$

where $d_{h,\pi_K} \in \Delta(\mathcal{S})$ for all natural numbers $h \geq 0$ is defined as

$$d_{h,\pi_K} = \begin{cases} d_0 & \text{if } h = 0, \\ P^o_{s',\pi_K(s')} & \text{otherwise, with } s' \sim d_{h-1,\pi_K}. \end{cases}$$

We emphasize that the state distribution d_{h,π_K}'s are different from the discounted state-action occupancy distributions. We note that a similar state distribution proof idea is used in [106].

Recall $\|f\|^2_{p,\nu} = (\mathbb{E}_{s,a\sim\nu}|f(s,a)|^p)^{1/p}$, where $\nu \in \Delta(\mathcal{S}\times\mathcal{A})$. With this we have

$$\mathbb{E}_{s_0\sim d_0}[V^{\pi^*}] - \mathbb{E}_{s_0\sim d_0}[V^{\pi_K}] \leq \sum_{h=0}^{\infty}\gamma^h\Big(\|Q^{\pi^*} - Q_K\|_{1,d_{h,\pi_K}\circ\pi^*} + \|Q^{\pi^*} - Q_K\|_{1,d_{h,\pi_K}\circ\pi_K}\Big), \quad \text{(C.12)}$$

where the state-action distributions $d_{h,\pi_K}\circ\pi^*(s,a) \propto d_{h,\pi_K}(s)\mathbb{1}\{a = \pi^*(s)\}$ and $d_{h,\pi_K}\circ\pi_K(s,a) \propto d_{h,\pi_K}(s)\mathbb{1}\{a = \pi_K(s)\}$ directly follows by comparing with (C.11).

We now bound one of the RHS terms above by bounding for any state-action distribution ν satisfying Assumption 4 (in particular the following bound is true for $d_{h,\pi_K}\circ\pi^*$ or $d_{h,\pi_K}\circ\pi_K$ in (C.11)):

$$\|Q^{\pi^*} - Q_K\|_{1,\nu} \leq \|Q^{\pi^*} - TQ_{K-1}\|_{1,\nu} + \|TQ_{K-1} - Q_K\|_{1,\nu}$$

$$
\begin{aligned}
&\overset{(a)}{\leq} \|Q^{\pi^*} - TQ_{K-1}\|_{1,\nu} + \sqrt{C}\|TQ_{K-1} - Q_K\|_{1,\mu} \\
&= (\mathbb{E}_{s,a\sim\nu}|Q^{\pi^*}(s,a) - TQ_{K-1}(s,a)|) + \sqrt{C}\|TQ_{K-1} - Q_K\|_{1,\mu} \\
&\overset{(b)}{\leq} (\mathbb{E}_{s,a\sim\nu}\gamma \sup_{\eta} |\mathbb{E}_{s'\sim P^o_{s,a}}((\eta - \max_{a'} Q_{K-1}(s',a'))_+ - (\eta - \max_{a'} Q^{\pi^*}(s',a'))_+)|) \\
&\qquad\qquad + \sqrt{C}\|TQ_{K-1} - Q_K\|_{1,\mu} \\
&\overset{(c)}{\leq} (\mathbb{E}_{s,a\sim\nu}|\mathbb{E}_{s'\sim P^o_{s,a}}(\max_{a'} Q^{\pi^*}(s',a') - \max_{a'} Q_{K-1}(s',a'))_+|) + \sqrt{C}\|TQ_{K-1} - Q_K\|_{1,\mu} \\
&\overset{(d)}{\leq} \gamma(\mathbb{E}_{s,a\sim\nu}\mathbb{E}_{s'\sim P^o_{s,a}} \max_{a'} |Q^{\pi^*}(s',a') - Q_{K-1}(s',a')|) + \sqrt{C}\|TQ_{K-1} - Q_K\|_{1,\mu} \\
&\overset{(e)}{\leq} \gamma\|Q^{\pi^*} - Q_{K-1}\|_{1,\nu'} + \sqrt{C}\|TQ_{K-1} - Q_K\|_{1,\mu} \\
&\overset{(f)}{\leq} \gamma\|Q^{\pi^*} - Q_{K-1}\|_{1,\nu'} + \sqrt{C}\|T_{g_{K-1}}Q_{K-1} - Q_K\|_{2,\mu} + \sqrt{C}\|TQ_{K-1} - T_{g_{K-1}}Q_{K-1}\|_{1,\mu},
\end{aligned}
\tag{C.13}
$$

where (a) follows by the concentratability assumption (Assumption 4), (b) from Bellman equation, operator T, and the fact $|\sup_x p(x) - \sup_x q(x)| \leq \sup_x |p(x) - q(x)|$, (c) from the fact $|(x)_+ - (y)_+| \leq |(x-y)_+|$ for any $x, y \in \mathbb{R}$, (d) follows by Jensen's inequality and by the facts $|\sup_x p(x) - \sup_x q(x)| \leq \sup_x |p(x) - q(x)|$ and $(x)_+ \leq |x|$ for any $x, y \in \mathbb{R}$, and (e) by defining the distribution ν' as $\nu'(s',a') = \sum_{s,a} \nu(s,a) P^o_{s,a}(s') \mathbb{1}\{a' = \operatorname{argmax}_b |Q^{\pi^*}(s',b) - Q_{K-1}(s',b)|\}$, and (f) using the fact that $\|\cdot\|_{1,\mu} \leq \|\cdot\|_{2,\mu}$.

Now, by recursion until iteration 0, we get

$$
\begin{aligned}
\|Q^{\pi^*} - Q_K\|_{1,\nu} &\leq \gamma^K \sup_{\bar{\nu}} \|Q^{\pi^*} - Q_0\|_{1,\bar{\nu}} + \sqrt{C}\sum_{t=0}^{K-1} \gamma^t \|TQ_{K-1-t} - T_{g_{K-1-t}}Q_{K-1-t}\|_{1,\mu} \\
&\qquad + \sqrt{C}\sum_{t=0}^{K-1} \gamma^t \|T_{g_{K-1-t}}Q_{K-1-t} - Q_{K-t}\|_{2,\mu} \\
&\overset{(a)}{\leq} \frac{\gamma^K}{1-\gamma} + \sqrt{C}\sum_{t=0}^{K-1} \gamma^t \|TQ_{K-1-t} - T_{g_{K-1-t}}Q_{K-1-t}\|_{1,\mu} \\
&\qquad + \sqrt{C}\sum_{t=0}^{K-1} \gamma^t \|T_{g_{K-1-t}}Q_{K-1-t} - Q_{K-t}\|_{2,\mu} \\
&\overset{(b)}{\leq} \frac{\gamma^K}{1-\gamma} + \frac{\sqrt{C}}{1-\gamma} \sup_{f\in\mathcal{F}} \|Tf - T_{\widehat{g}_f} f\|_{1,\mu} + \frac{\sqrt{C}}{1-\gamma} \sup_{f\in\mathcal{F}} \|T_{\widehat{g}_f} f - \widehat{f}_{\widehat{g}_f}\|_{2,\mu}
\end{aligned}
$$

$$\leq \frac{\gamma^K}{1-\gamma} + \frac{\sqrt{C}}{1-\gamma} \sup_{f \in \mathcal{F}} \|Tf - T_{\widehat{g}_f} f\|_{1,\mu} + \frac{\sqrt{C}}{1-\gamma} \sup_{f \in \mathcal{F}} \sup_{g \in \mathcal{G}} \|T_g f - \widehat{f}_g\|_{2,\mu}. \tag{C.14}$$

where (a) follows since $|Q^{\pi^*}(s,a)| \leq 1/(1-\gamma), Q_0(s,a) = 0$, and (b) follows since $\widehat{g}_f$ is the dual variable function from the algorithm for the state-action value function f and $\widehat{f}_g$ as the least squares solution from the algorithm for the state-action value function f and dual variable function g pair.

The proof is now complete combining (C.12) and (C.14) with Lemma 28 and Lemma 29. □

C.4 Related Works

Here we provide a more detailed description of the related work to complement what we listed in the introduction (Section 4.1).

Offline RL: The problem of learning the optimal policy only using an offline dataset is first addressed under the generative model assumption [75, 76, 77, 78, 79, 4, 80]. This assumption requires generating the same uniform number of next-state samples for each and every state-action pairs. To account for large state spaces, there are number of works [89, 63, 90, 91, 92, 93, 94] that utilize function approximation under similar assumption, concentratability assumption [91] in which the data distribution μ sufficiently covers the discounted state-action occupancy. There is rich literature [68, 102, 69, 91, 103, 94] in the conquest of identifying and improving these necessary and sufficient assumptions for offline RL that use variations of Fitted Q-Iteration (FQI) algorithm [100, 101]. There is also rich literature [95, 96, 97, 98, 99] that develop offline deep RL algorithms focusing on the algorithmic and empirical aspects and propose multitude heuristic approaches to advance the field. All these results assume that the offline data is generated according to a single model and the goal is to find the optimal policy for the MDP with the same model. In particular, none of these works consider the *offline robust RL problem* where the offline data is generated according to a (training) model which can be different from the one in testing, and the goal is to learn a policy that is robust w.r.t. an uncertainty set.

Robust RL: To address the parameter uncertainty problem, [33] and [34] introduced the RMDP

framework. [33] showed that the optimal robust value function and policy can be computed using the robust counterparts of the standard value iteration and policy iteration algorithms. To tackle the parameter uncertainty problem, other works considered distributionally robust setting [37], modified policy iteration [57], and more general uncertainty set [36]. These initial works mainly focused on the planning problem (known transition probability dynamics) in the tabular setting. [39] proposed linear function approximation method to solve large RMDPs. Though this work suggests a sampling based approach, a general model-free learning algorithm and analysis was not included. [40] 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. [48] introduced soft-robust actor-critic algorithms using neural networks, but does not provide any global convergence guarantees for the learned policy. [58] proposed a min-max game framework to address the robust learning problem focusing on the tabular setting. [59] proposed a kernel-based RL algorithm for finding the robust value function in a batch learning setting. [46] employed an entropy-regularized policy optimization algorithm for continuous control using neural network, but does not provide any provable guarantees for the learned policy. [1] proposed least-squares policy iteration method to handle large state-action space in robust RL, but only provide asymptotic policy evaluation convergence guarantees whereas [1] provide finite time convergence for the policy iteration to optimal robust value.

Other robust RL related works: Robust control is a well-studied area in the classical control theory [84, 85]. Recently, there are some interesting works that address the robust RL problem using this framework, especially focusing on the linear quadratic regulator setting [86]. Risk sensitive RL algorithms [87, 170, 171] and adversarial RL algorithms [45, 172, 173] also address the robustness problem implicitly under different frameworks which are independent from RMDPs. Our framework and approach of robust MDP is significantly different from these line of works.

The works that are closest to ours are by [82, 83, 2] that address the robust RL problem in a

tabular setting under the generative model assumption. Due to the generative model assumption, the offline data has the same uniform number of samples corresponding to each and every state-action pair, and tabular setting allows the estimation of the uncertainty set followed by solving the planning problem. Our work is significantly different from these in the following way: (i) we consider a robust RL problem with arbitrary large state space, instead of the small tabular setting, (ii) we consider a true offline RL setting where the state-action pairs are sampled according to an arbitrary distribution, instead of using the generative model assumption, (iii) we focus on a function approximation approach where the goal is to directly learn optimal robust value/policy using function approximation techniques, instead of solving the tabular planning problem with the estimated model. *To the best of our knowledge, this is the first work that addresses the offline robust RL problem with arbitrary large state space using function approximation, with provable guarantees on the performance of the learned policy.*

C.5 Experiment Details

We provide more detailed and practical version of our RFQI algorithm (Algorithm 3) in this section. We also provide more experimental results evaluated on *Cartpole*, *Hopper*, and *Half-Cheetah* OpenAI Gym Mujoco [71] environments.

We provide our code in a **github repository** https://github.com/zaiyan-x/RFQI containing instructions to reproduce all results in this paper. We implemented our RFQI algorithm based on the architecture of Batch Constrained deep Q-learning (BCQ) algorithm [95] [*] and Pessimistic Q-learning (PQL) algorithm [103] [†]. We note that PQL algorithm (with $b = 0$ filtration thresholding [103]) and BCQ algorithm are the practical versions of FQI algorithm with neural network architecture.

C.5.1 RFQI Practical Algorithm

We provide the practical version of our RFQI algorithm in Algorithm 7 and highlight the difference with BCQ and PQL algorithms in blue (steps 8 and 9).

[*] Available at https://github.com/sfujim/BCQ
[†] Available at https://github.com/yaoliucs/PQL

Algorithm 7 RFQI Practical Algorithm

Input: Offline dataset $\mathcal{D}$, radius of robustness ρ, maximum perturbation Φ, target update rate τ, mini-batch size N, maximum number of iterations K, number of actions u.
Initialize: Two state-action neural networks Q_{θ_1} and Q_{θ_2}, one dual neural network g_{θ_3} policy (perturbation) model: $\xi_\varphi \in [-\Phi, \Phi]$), and action VAE G^a_ω, with random parameters θ_1, θ_2, φ, ω, and target networks $Q_{\theta'_1}, Q_{\theta'_2}, \xi_{\varphi'}$ with $\theta'_1 \leftarrow \theta_1, \theta'_2 \leftarrow \theta_2$, $\varphi' \leftarrow \varphi$.
for $k = 1, \cdots, K$ **do**
  Sample a minibatch B with N samples from $\mathcal{D}$.
  Train $\omega \leftarrow \text{argmin}_\omega \, ELBO(B; G^a_\omega)$. Sample u actions a'_i from $G^a_\omega(s')$ for each s'.
  Perturb u actions $a'_i = a'_i + \xi_\varphi(s', a'_i)$.
  Compute next-state value target for each s' in B:

$$V_t = \max_{a'_i}(0.75 \cdot \min\{Q_{\theta'_1}, Q_{\theta'_2}\} + 0.25 \cdot \max\{Q_{\theta'_1}, Q_{\theta'_2}\}).$$

  $\theta_3 \leftarrow \text{argmin}_\theta \sum[\max\{g_\theta(s,a) - V_t(s'), 0\} - (1-\rho)g_\theta(s,a)]$.
  Compute next-state Q target for each (s, a, r, s') pair in B:

$$Q_t(s,a) = r - \gamma \cdot \max\{g_{\theta_3}(s,a) - V_t(s'), 0\} + \gamma(1-\rho)g_{\theta_3}(s,a).$$

  $\theta \leftarrow \text{argmin}_\theta \sum(Q_t(s,a) - Q_\theta(s,a))^2$.
  Sample u actions a_i from $G^a_\omega(s)$ for each s.
  $\varphi \leftarrow \text{argmax}_\varphi \sum \max_{a_i} Q_{\theta_1}(s, a_i + \xi_\varphi(s, a_i))$.
  Update target network: $\theta' = (1-\tau)\theta' + \tau\theta, \varphi' = (1-\tau)\varphi' + \tau\varphi$.
end for
Output policy: Given s, sample u actions a_i from $G^a_\omega(s)$. Select action $a = \text{argmax}_{a_i} Q_{\theta_1}(s, a_i + \xi_\varphi(s, a_i))$.

RFQI algorithm implementation details: The Variational Auto-Encoder (VAE) G^a_ω [174] is defined by two networks, an encoder $E_{\omega_1}(s,a)$ and decoder $D_{\omega_2}(s,z)$, where $\omega = \{\omega_1, \omega_2\}$. The encoder outputs mean and standard deviation, $(\mu, \sigma) = E_{\omega_1}(s,a)$, of a normal distribution. A latent vector z is sampled from the standard normal distribution and for a state s, the decoder maps them to an action $D_{\omega_2} : (s,z) \mapsto \tilde{a}$. Then the evidence lower bound ($ELBO$) of VAE is given by $ELBO(B; G^a_\omega) = \sum_B (a - \tilde{a})^2 + D_{\text{KL}}(\mathcal{N}(\mu, \sigma), \mathcal{N}(0,1))$, where $\mathcal{N}$ is the normal distribution with mean and standard deviation parameters. We refer to [95] for more details on VAE. We also use the default VAE architecture from BCQ algorithm [95] and PQL algorithm [103] in our RFQI algorithm.

We now focus on the additions described in blue (steps 8 and 9) in Algorithm 7. For all the other networks we use default architecture from BCQ algorithm [95] and PQL algorithm [103] in our RFQI algorithm.

(1) In each iteration k, we solve the dual variable function g_θ optimization problem (step 4 in Algorithm 3, step 8 in Algorithm 7) implemented by ADAM [175] on the minibatch B with the learning rate l_1 mentioned in Table C.1.

(2) Our state-action value target function corresponds to the robust state-action value target function described in (4.10). This is reflected in step 9 of Algorithm 7. The state-action value function Q_θ optimization problem (step 5 in Algorithm 3, step 9 in Algorithm 7) is implemented by ADAM [175] on the minibatch B with the learning rate l_2 mentioned in Table C.1.

Environment	Discount γ	Learning rates $[l_1, l_2]$	Q Neural nets $\theta_1 = \theta_2 = [h_1, h_2]$	Dual Neural nets $\theta_3 = [h_1, h_2]$
CartPole	0.99	$[10^{-3}, 10^{-3}]$	$[400, 300]$	$[64, 64]$
Hopper	0.99	$[10^{-3}, 8 \times 10^{-4}]$ $[3 \times 10^{-4}, 6 \times 10^{-4}]$	$[400, 300]$	$[64, 64]$
Half-Cheetah	0.99	$[10^{-3}, 8 \times 10^{-4}]$ $[3 \times 10^{-4}, 6 \times 10^{-4}]$	$[400, 300]$	$[64, 64]$

Table C.1: Details of hyper-parameters in FQI and RFQI algorithms experiments.

Hyper-parameters details: We now give the description of hyper-parameters used in our codebase in Table C.1. We use same hyper-parameters across different algorithms. Across all learning algorithms we use $\tau = 0.005$ for the target network update, $K = 5 \times 10^5$ for the maximum iterations, $|\mathcal{D}| = 10^6$ for the offline dataset, $|B| = 1000$ for the minibatch size. We used grid-search for ρ in $\{0.2, 0.3, \cdots, 0.6\}$. We also picked best of the two sets of learning rates mentioned in Table

C.1. For all the other hyper-parameters we use default values from BCQ algorithm [95] and PQL algorithm [103] in our RFQI algorithm that can be found in our code.

Offline datasets: Now we discuss the offline dataset used in the our training of FQI and RFQI algorithms. For the fair comparison in every plot, we train both FQI and RFQI algorithms on same offline datasets.

Cartpole dataset $\mathcal{D}_{\text{c}}$: We first train proximal policy optimization (PPO) [176] algorithm, under default RL baseline zoo [177] parameters. We then generate the Cartpole dataset $\mathcal{D}_{\text{c}}$ with 10^5 samples using an ε-greedy ($\varepsilon = 0.3$) version of this PPO trained policy. We note that this offline dataset contains non-expert behavior meeting the richness of the data-generating distribution assumption in practice.

Mixed dataset $\mathcal{D}_{\text{m}}$: For the MuJoCo environments, *Hopper* and *Half-Cheetah*, we increase the richness of the dataset since these are high dimensional problems. We first train soft actor-critic (SAC) [178] algorithm, under default RL baseline zoo [177] parameters, with replay buffer updated by a fixed ε-greedy ($\varepsilon = 0.1$) policy with the model parameter *actuator_ctrlrange* set to $[-0.85, 0.85]$. We then generate the mixed dataset $\mathcal{D}_{\text{m}}$ with 10^6 samples from this ε-greedy ($\varepsilon = 0.3$) SAC trained policy. We note that such a dataset generation gives more diverse set of observations than the process of $\mathcal{D}_{\text{c}}$ generation for fair comparison between FQI and RFQI algorithms.

D4RL dataset $\mathcal{D}_{\text{d}}$: We consider the *hopper-medium* and *halfcheetah-medium* offline datasets in [146] which are benchmark datasets in offline RL literature [146, 93, 103]. These 'medium' datasets are generated by first training a policy online using Soft Actor-Critic [178], early-stopping the training, and collecting 10^6 samples from this partially-trained policy. We refer to [146] for more details.

We end this section by mentioning the software and hardware configurations used. The training and evaluation is done using three computers with the following configuration. Operating system is Ubuntu 18.04 and Lambda Stack; main softwares are PyTorch, Caffe, CUDA, cuDNN, Numpy, Matplotlib; processor is AMD Threadripper 3960X (24 Cores, 3.80 GHz); GPUs are 2x RTX 2080

Ti; memory is 128GB RAM; Operating System Drive is 1 TB SSD (NVMe); and Data Drive is 4TB HDD.

C.5.2 More Experimental Results

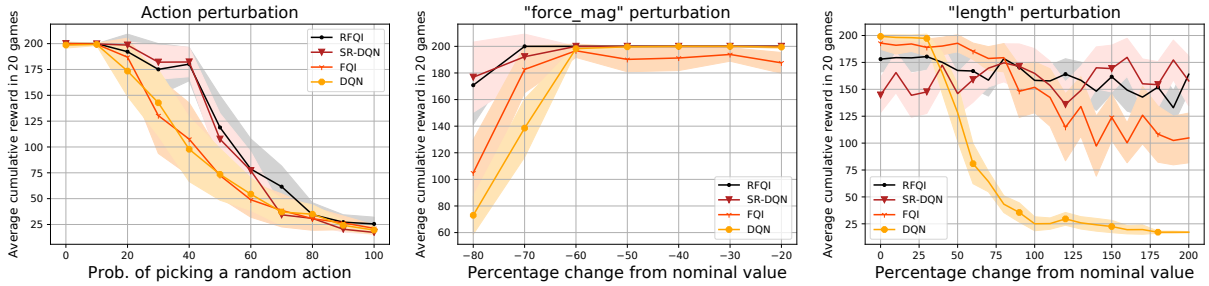

Figure C.1: *Cartpole simulation results on offline dataset* $\mathcal{D}_c$. Average cumulative reward in 20 episodes versus different model parameter perturbations mentioned in the respective titles.

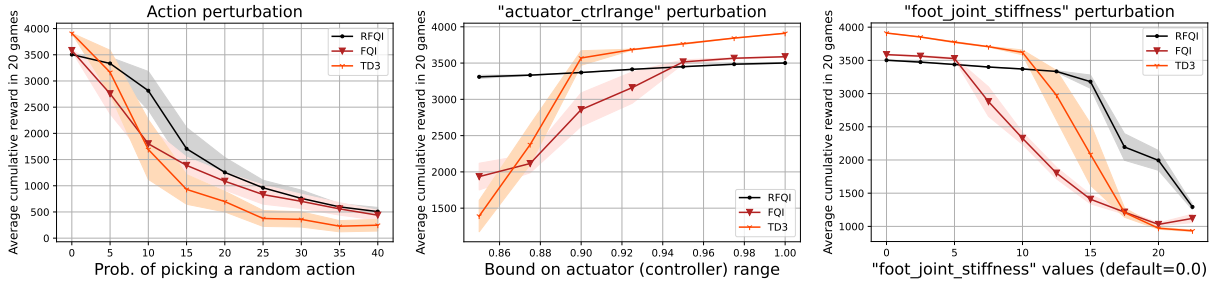

Figure C.2: *Hopper simulation results on offline dataset* $\mathcal{D}_m$. Average cumulative reward in 20 episodes versus different model parameter perturbations mentioned in the respective titles.

Here we provide more experimental results and details in addition to Fig. 4.1-4.3 in Section 4.5.

For the *Cartpole*, we compare RFQI algorithm against the non-robust RL algorithms FQI and DQN, and the soft-robust RL algorithm proposed in [48]. We trained FQI and RFQI algorithms on the dataset $\mathcal{D}_c$ (a detailed description of data set is provided in Appendix C.5.1). We test the robustness of the algorithms by changing the parameters *force_mag* (to model external force disturbance), *length* (to model change in pole length), and also by introducing action perturbations (to model actuator noise). The nominal value of *force_mag* and *length* parameters are 10 and 0.5 respectively. Fig. C.1 shows superior robust performance of RFQI compared to the non-robust

FQI and DQN. For example, consider the action perturbation performance plot in Fig. C.1 where RFQI algorithm improves by 75% compared to FQI algorithm in average cumulative reward for a 40% chance of action perturbation. We note that we found $\rho = 0.5$ is the best from grid-search for RFQI algorithm. The RFQI performance is similar to that of soft-robust DQN. We note that soft-robust DQN algorithm is an online deep RL algorithm (and not an offline RL algorithm) and has no provable performance guarantee. Moreover, soft-robust DQN algorithm requires generating online data according a number of models in the uncertainty set, whereas RFQI only requires offline data according to a single nominal training model.

Before we proceed to describe our results on the OpenAI Gym MuJoCo [71] environments *Hopper* and *Half-Cheetah*, we first mention their model parameters and its corresponding nominal values in Table C.2. The model parameter names are self-explanatory, for example, stiffness control on the leg joint is the *leg_joint_stiffness*, range of actuator values is the *actuator_ctrlrange*. The front and back parameters in Half-Cheetah are for the front and back legs. We refer to the perturbed environments provided in our code and the *hopper.xml, halfcheetah.xml* files in the environment assets of OpenAI Gym MuJoCo [71] for more information regarding these model parameters.

Environment	Model parameter	Nominal range/value
Hopper	*actuator_ctrlrange*	$[-1, 1]$
	foot_joint_stiffness	0
	leg_joint_stiffness	0
	thigh_joint_stiffness	0
	joint_damping	1
	joint_frictionloss	0
Half-Cheetah	*joint_frictionloss*	0
	front *actuator_ctrlrange*	$[-1, 1]$
	back *actuator_ctrlrange*	$[-1, 1]$
	front *joint_stiffness* = (*thigh_joint_stiffness*, *shin_joint_stiffness*, *foot_joint_stiffness*)	$(180, 120, 60)$
	back *joint_stiffness* = (*thigh_joint_stiffness*, *shin_joint_stiffness*, *foot_joint_stiffness*)	$(240, 180, 120)$
	front *joint_damping* = (*thigh_joint_damping*, *shin_joint_damping*, *foot_joint_damping*)	$(4.5, 3.0, 1.5)$
	back *joint_damping* = (*thigh_joint_damping*, *shin_joint_damping*, *foot_joint_damping*)	$(6.0, 4.5, 3.0)$

Table C.2: Details of model parameters for *Hopper* and *Half-Cheetah* environments.

For the *Hopper*, we compare RFQI algorithm against the non-robust RL algorithms FQI and TD3 [110]. We trained FQI and RFQI algorithms on the mixed dataset $\mathcal{D}_{\text{m}}$ (a detailed description of dataset provided in Appendix C.5.1). We note that we do not compare with soft robust RL algorithms because of its poor performance on MuJoCo environments in the rest of our figures. We test the robustness of the algorithm by introducing action perturbations, and by changing the model

parameters *actuator_ctrlrange*, *foot_joint_stiffness*, and *leg_joint_stiffness*. Fig. 4.3 and Fig. C.2 shows RFQI algorithm is consistently robust compared to the non-robust algorithms. We note that we found $\rho = 0.5$ is the best from grid-search for RFQI algorithm. The average episodic reward of RFQI remains almost the same initially, and later decays much less and gracefully when compared to FQI and TD3 algorithms. For example, in plot 3 in Fig. C.2, at the *foot_joint_stiffness* parameter value 15, the episodic reward of FQI is only around 1400 whereas RFQI achieves an episodic reward of 3200. Similar robust performance of RFQI can be seen in other plots as well. We also note that TD3 [95] is a powerful off-policy policy gradient algorithm that relies on large 10^6 replay buffer of online data collection, unsurprisingly performs well initially with less perturbation near the nominal models.

In order to verify the effectiveness and consistency of our algorithm across different offline dataset, we repeat the above experiments, on additional OpenAI Gym MuJoCo [71] environment *Half-Cheetah*, using D4RL dataset $\mathcal{D}_{\mathrm{d}}$ (a detailed description of dataset provided in Appendix C.5.1) which are benchmark in offline RL literature [146, 93, 103] than our mixed dataset $\mathcal{D}_{\mathrm{m}}$. Since D4RL dataset is a benchmark dataset for offline RL algorithms, here we focus only on the comparison between the two offline RL algorithms we consider, our RFQI algorithm and its non-robust counterpart FQI algorithm. We now showcase the results on *Hopper* and *Half-Cheetah* for this setting.

For the *Hopper*, we test the robustness by changing the model parameters *gravity*, *joint_damping*, and *joint_frictionloss*. Fig. C.3 shows RFQI algorithm is consistently robust compared to the non-robust FQI algorithm. We note that we found $\rho = 0.5$ is the best from grid-search for RFQI algorithm. The average episodic reward of RFQI remains almost the same initially, and later decays much less and gracefully when compared to FQI algorithm. For example, in plot 2 in Fig. C.3, for the 30% change in *joint_damping* parameter, the episodic reward of FQI is only around 1400 whereas RFQI achieves an episodic reward of 3000 which is almost the same as for unperturbed model. Similar robust performance of RFQI can be seen in other plots as well.

For the *Half-Cheetah*, we test the robustness by changing the model parameters *joint_stiffness*

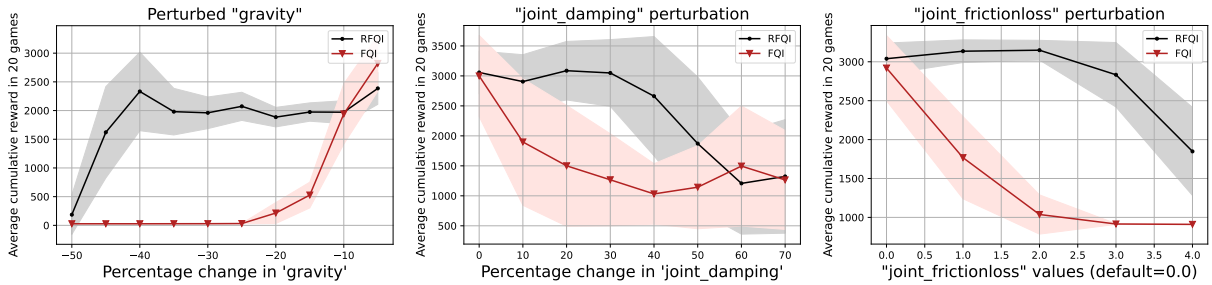

Figure C.3: *Hopper evaluation simulation results on offline dataset $\mathcal{D}_d$.* Average cumulative reward in 20 episodes versus different model parameter perturbations mentioned in the respective titles.

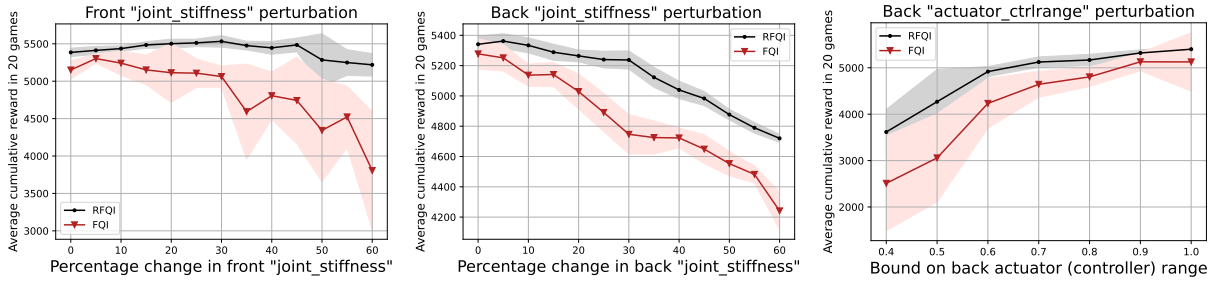

Figure C.4: *Half-Cheetah evaluation simulation results on offline dataset $\mathcal{D}_d$.* Average cumulative reward in 20 episodes versus different model parameter perturbations mentioned in the respective titles.

of front and back joints, and *actuator_ctrlrange* of back joint. Fig. C.4 shows RFQI algorithm is consistently robust compared to the non-robust FQI algorithm. We note that we found $\rho = 0.3$ is the best from grid-search for RFQI algorithm. For example, in plot 1 in Fig. C.4, RFQI episodic reward stays at around 5500 whereas FQI drops faster to 4300 for more than 50% change in the nominal value. Similar robust performance of RFQI can be seen in other plots as well.

As part of discussing the limitations of our work, we also provide two instances where RFQI and FQI algorithm behave similarly. RFQI and FQI algorithms trained on the D4RL dataset $\mathcal{D}_{\text{d}}$ perform similarly under the perturbations of the *Hopper* model parameters *actuator_ctrlrange* and *thigh_joint_stiffness* as shown in Fig. C.5. We also make similar observations under the perturbations of the *Half-Cheetah* model parameters *joint_damping* (both front *joint_damping* and back *joint_damping*) and *joint_frictionloss* as shown in Fig. C.6. We observed that the robustness performance can depend on the offline data available, which was also observed for non-robust offline RL algorithms [103, 146, 93]. Also, perturbing some parameters may make the problem

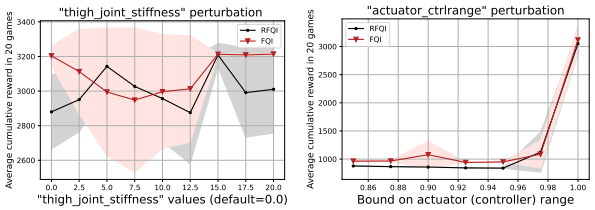

Figure C.5: Similar performance of RFQI and FQI in Hopper on dataset $\mathcal{D}_{\text{d}}$ w.r.t. parameters *actuator_ctrlrange* and *thigh_joint_stiffness*.

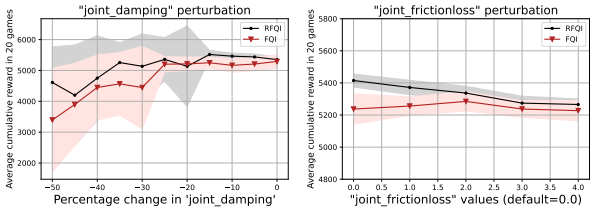

Figure C.6: Similar performance of RFQI and FQI in Half-Cheetah on dataset $\mathcal{D}_{\text{d}}$ w.r.t. parameters *joint_damping* and *joint_frictionloss*.

really hard especially if the data is not representative with respect to that parameter. We believe that this is the reason for the similar performance of RFQI and FQI w.r.t. some parameters. We believe that this opens up an exciting area of research on developing online policy gradient algorithms for robust RL, which may be able to overcome the restriction and challenges due to offline data. We plan to pursue this goal in our future work.

APPENDIX D

APPENDIX FOR CHAPTER 5*

☕ Supplementary Materials ☕

In this appendix, we include complete proofs, all experiments, and all supporting details for the corresponding chapter.

D.1 Useful Technical Results

The following result from [139, Lemma 26] is useful for characterizing the entropy regularized value performance with respect to the policy.

Lemma 30 (Soft sub-optimality lemma)**.** *For any policy π and $\tau > 0$,*

$$\tilde{V}_{\pi_\tau^*} - \tilde{V}_\pi = \frac{\tau}{1-\gamma} \mathbb{E}_{s\sim d^\pi}[D_{\mathrm{KL}}(\pi(\cdot \mid s) \| \pi_\tau^*(\cdot \mid s))].$$

The following lemma is a generalization from [179] for relating entropy regularized and classic value functions.

Lemma 31. *For any policy π and $\tau > 0$, $V_\pi - \tau \log(|\mathcal{A}|)/(1-\gamma) \leq \tilde{V}_\pi \leq V_\pi$.*

Proof. Recall the *entropy regularized* value function and value function definitions. That is, for any policy π and $\tau > 0$,

$$\tilde{V}_\pi = \mathbb{E}_{s_0\sim\mu, a_t\sim\pi(\cdot|s_t), s_{t+1}\sim P^o_{s_t,a_t}} \sum_{t=0}^{\infty} \gamma^t (r(s_t,a_t) - \tau \log \pi(a_t \mid s_t)) \quad \text{and}$$
$$V_\pi = \mathbb{E}_{s_0\sim\mu, a_t\sim\pi(\cdot|s_t), s_{t+1}\sim P^o_{s_t,a_t}} \sum_{t=0}^{\infty} \gamma^t r(s_t,a_t).$$

Since $r(s_t,a_t) - \tau \log \pi(a_t \mid s_t) \leq r(s_t,a_t)$, it directly follows $\tilde{V}_\pi \leq V_\pi$. Note that the entropy of

*Reprinted with permission from Kishan Panaganti, Zaiyan Xu, Dileep Kalathil, Mohammad Ghavamzadeh, "Improving Behavioral Cloning with Distributionally Robust Optimization." Preprint, 2023.

any discrete distribution is bounded above by the entropy of the uniform distribution in its support [168]. Now, the other side follows since $\sum_{t=0}^{\infty} \gamma^t \tau \log \pi(a_t \mid s_t) \leq \tau \log(1/|\mathcal{A}|)/(1-\gamma)$. □

We state a result which shows that the state-distributions are close when two policies are close under the same model.

Lemma 32. *Consider any model P and policies π, π'. We have $D_{\mathrm{TV}}(d^{\pi}, d^{\pi'}) \leq \frac{\gamma}{(1-\gamma)} \sqrt{\mathbb{E}_{s\sim d^{\pi}}[D_{\mathrm{KL}}(\pi'(\cdot|s)\|\pi(\cdot|s)}$*

Proof. From the proof analyses in [106, Lemma 14.1] and Lemma 35, we get

$$\left\| d^{\pi} - d^{\pi'} \right\|_1 \leq \frac{\gamma}{(1-\gamma)} \mathbb{E}_{s\sim d^{\pi}}[\|\pi'(\cdot|s) - \pi(\cdot|s)\|_1].$$

From Pinsker's and Jensen's inequality [180], we get the required result. □

Let P^o be a distribution on the space $\mathcal{X}$. We now state a result from [3] based on DRO methodology.

Lemma 33 ([3, Lemma 5])**.** *Eqs.* (5.3) *and* (5.6) *is with $D_{\mathrm{TV}}(\cdot,\cdot)$. Let $l : \mathcal{X} \to \mathbb{R}$ be a loss function. Then*

$$\sup_{D_{\mathrm{TV}}(P,P^o)\leq\rho} \mathbb{E}_{x\sim P}[l(x)] = \inf_{\eta\in\mathbb{R}} \left\{ \mathbb{E}_{x\sim P^o}[(l(x)-\eta)_+] + (\sup_{x\in\mathcal{X}} l(x) - \eta)_+ \cdot \rho + \eta \right\}. \tag{D.1}$$

We now specialize [109, Corollary 2] for the total variation distance.

Lemma 34. *Let $\Theta \subseteq \mathbb{R}^d$, $l : \mathcal{X} \times \Theta \mapsto [0, M]$ and fix any $\rho \in (0, 1]$. We have*

$$\sup_{D_{\mathrm{TV}}(P,P^o)\leq\rho} \mathbb{E}_P[l(X,\widehat{\theta})] \leq \inf_{\theta\in\Theta} \sup_{D_{\mathrm{TV}}(P,P^o)\leq\sqrt{\rho}} \mathbb{E}_P[l(X,\theta)] + cM(1+\rho)\sqrt{\frac{\log(1/\delta) + 2d\log(N)}{N}},$$

which holds with probability at least $1-\delta$, where $c > 0$ is some universal constant.

Proof. First set $\rho = 2\rho'$. We first note that from Pinsker's inequality [180], [181, Theorem 5], and [182, Lemma 11.1], we have $2D_{\chi^2}(p,q) \leq D_{\mathrm{TV}}(p,q) \leq 2\sqrt{D_{\chi^2}(p,q)}$ for any two distributions. From this, it is easy to see that $\{p : D_{\mathrm{TV}}(p,q) \leq 2\rho'\} \subseteq \{p : D_{\chi^2}(p,q) \leq \rho'\} \subseteq \{p : D_{\mathrm{TV}}(p,q) \leq 2\sqrt{\rho'}\}$. This completes the proof by applying [109, Corollary 2] for D_{χ^2} with radius $\rho/2$. □

D.2 Proof of Results in Section 5.2

D.2.1 Proof of Theorem 10

Proof. We recall $\pi_e = \pi^*_\tau$. Now substituting $\pi = \pi_{\mathrm{il}}$ in Lemma 30 we get

$$\tilde{V}_{\pi_e} - \tilde{V}_{\pi_{\mathrm{il}}} = \frac{\tau}{1-\gamma}\mathbb{E}_{s\sim d^{\pi_{\mathrm{il}}}}[D_{\mathrm{KL}}(\pi_{\mathrm{il}}(\cdot \mid s)\|\pi_e(\cdot \mid s))] = \frac{\tau L_{\mathrm{il}}(\pi_{\mathrm{il}})}{1-\gamma}, \tag{D.2}$$

where the last equality follows from Eq. (5.2). We note that $L_{\mathrm{il}}(\pi_{\mathrm{il}}) = \varepsilon_{\mathrm{il}}$ and $\tau' = \tau\log(|\mathcal{A}|)$. The proof of this theorem is now complete by combining Eq. (D.2) with Lemma 31. □

D.2.2 Proof of Proposition 7

We provide the formal statement and its proof here.

Proposition 13. *Fix the underlying model P^o. For all $\pi \in \Pi$ and $\rho_{\mathrm{c}} > 0$,*

$$\begin{aligned}
&\max_{\pi'\in\Pi_{\rho_{\mathrm{c}}}} \mathbb{E}_{s\sim d^{\pi'}_{Po}}[l(\pi_e(\cdot|s),\pi(\cdot|s))] \\
&= \min_{\eta\in\mathbb{R}} \mathbb{E}_{s\sim d^{\pi_e}_{Po}}[(l(\pi_e(\cdot|s),\pi(\cdot|s)) - \eta)_+] + (\sup_{s\in\mathcal{S}: d^{\pi_e}_{Po}(s)>0} l(\pi_e(\cdot|s),\pi(\cdot|s)) - \eta)_+ \cdot \rho + \eta.
\end{aligned}$$

Proof. We rewrite $\max_{\pi'\in\Pi_{\rho_{\mathrm{c}}}} \mathbb{E}_{s\sim d^{\pi'}_{Po}}[l(\pi_e(\cdot|s),\pi(\cdot|s))]$ as

$$\max_{d^{\pi'}_{Po}: D_{\mathrm{TV}}(d^{\pi'}_{Po}, d^{\pi_e}_{Po})\le\rho_{\mathrm{c}}} \mathbb{E}_{s\sim d^{\pi'}_{Po}}[l(\pi_e(\cdot|s),\pi(\cdot|s))]$$

since $\Pi_{\rho_{\mathrm{c}}} = \{\pi : D_{\mathrm{TV}}(d^{\pi}_{Po}, d^{\pi_e}_{Po}) \le \rho_{\mathrm{c}}\}$. The result immediately follows from Lemma 33. □

D.2.3 Proof of Theorem 11

Proof. Firstly, notice that from Lemma 32 we have

$$D_{\mathrm{TV}}(d^{\pi}, d^{\pi_e}) \le \frac{\gamma}{1-\gamma}\sqrt{L_{\mathrm{bc}}(\pi)} \le \frac{\gamma}{1-\gamma}\sqrt{L_{\mathrm{drbc}}(\pi,\rho_{\mathrm{c}})}, \tag{D.3}$$

where in the last inequality we have used that the fact that $L_{\mathrm{bc}}(\pi) \leq L_{\mathrm{drbc}}(\pi)$ for any π. Now, for $\pi = \pi_{\mathrm{drbc}}$, we further get

$$D_{\mathrm{TV}}(d^{\pi_{\mathrm{drbc}}}, d^{\pi_e}) \leq \frac{\gamma}{1-\gamma}\sqrt{L_{\mathrm{drbc}}(\pi_{\mathrm{drbc}}, \rho_{\mathrm{c}})} \overset{(a)}{=} \frac{\gamma\sqrt{\varepsilon_{\mathrm{drbc}}(\rho_{\mathrm{c}})}}{1-\gamma} \overset{(b)}{\leq} \rho_{\mathrm{c}}, \tag{D.4}$$

where (a) follows since $L_{\mathrm{drbc}}(\pi_{\mathrm{drbc}}, \rho_{\mathrm{c}}) = \varepsilon_{\mathrm{drbc}}(\rho_{\mathrm{c}})$ and (b) since ρ_{c} is in the feasibility set $\mathcal{U} = \{x \in (0,1] : \gamma^2\varepsilon_{\mathrm{drbc}}(x) \leq x(1-\gamma)^2\}$.

We recall $\pi_e = \pi_\tau^*$. Letting $\pi = \pi_{\mathrm{drbc}}$, we get

$$\begin{aligned}
L_{\mathrm{il}}(\pi_{\mathrm{drbc}}) &= \sum_{s\in\mathcal{S}} d^{\pi_{\mathrm{drbc}}}(s) D_{\mathrm{KL}}(\pi_{\mathrm{drbc}}(\cdot \mid s) \| \pi_e(\cdot \mid s)) \\
&= \mathbb{1}(D_{\mathrm{TV}}(d^{\pi_{\mathrm{drbc}}}, d^{\pi_e}) \leq \rho_{\mathrm{c}}) \sum_{s\in\mathcal{S}} d^{\pi_{\mathrm{drbc}}}(s) D_{\mathrm{KL}}(\pi_{\mathrm{drbc}}(\cdot \mid s) \| \pi_e(\cdot \mid s)) \\
&\qquad + \mathbb{1}(D_{\mathrm{TV}}(d^{\pi_{\mathrm{drbc}}}, d^{\pi_e}) > \rho_{\mathrm{c}}) \sum_{s\in\mathcal{S}} \frac{d^{\pi_{\mathrm{drbc}}}(s)}{d^{\pi_e}(s)} d^{\pi_e}(s) D_{\mathrm{KL}}(\pi_{\mathrm{drbc}}(\cdot \mid s) \| \pi_e(\cdot \mid s)) \\
&\overset{(c)}{\leq} L_{\mathrm{drbc}}(\pi_{\mathrm{drbc}}, \rho_{\mathrm{c}}) \overset{(d)}{=} \varepsilon_{\mathrm{drbc}}(\rho_{\mathrm{c}}),
\end{aligned} \tag{D.5}$$

where (c) holds by definition of L_{drbc} Eq. (5.3) and Eq. (D.4), and (d) follows by $L_{\mathrm{drbc}}(\pi_{\mathrm{drbc}}, \rho) = \varepsilon_{\mathrm{drbc}}(\rho)$. Now substituting $\pi = \pi_{\mathrm{drbc}}$ in Lemma 30 we get

$$\tilde{V}_{\pi_e} - \tilde{V}_{\pi_{\mathrm{drbc}}} = \frac{\tau}{1-\gamma}\mathbb{E}_{s\sim d^{\pi_{\mathrm{drbc}}}}[D_{\mathrm{KL}}(\pi_{\mathrm{drbc}}(\cdot \mid s) \| \pi_e(\cdot \mid s))] = \frac{\tau L_{\mathrm{il}}(\pi_{\mathrm{drbc}})}{1-\gamma}, \tag{D.6}$$

where the last equality follows from Eq. (5.2). We note that $\tau' = \tau\log(|\mathcal{A}|)$. The proof of this theorem is now complete by combining Eqs. (D.5) and (D.6) with Lemma 31. □

D.2.4 Proof of Theorem 13

Proof. Firstly, from Lemma 34, we observe that $L_{\mathrm{drbc}}(\widehat{\pi}_{\mathrm{drbc}}, \rho_{\mathrm{c}}) \leq \widehat{\varepsilon}_{\mathrm{drbc}}(\rho_{\mathrm{c}})$ holds with probability at least $1-\delta$ with

$$\widehat{\varepsilon}_{\mathrm{drbc}}(\rho_{\mathrm{c}}) = \varepsilon_{\mathrm{drbc}}(\rho_{\mathrm{c}}) + c(1+\rho_{\mathrm{c}})\sqrt{\frac{\log(1/\delta) + 2\,|\mathcal{A}|\log(N)}{\mathrm{e}_{\min}N}},$$

where $\mathrm{e}_{\min} = \min_{s,a:\pi_e(a|s)>0} \pi_e(a|s)$ and $c > 0$ is some universal constant.

For $\pi = \widehat{\pi}_{\mathrm{drbc}}$ in Eq. (D.3), we get

$$D_{\mathrm{TV}}(d^{\widehat{\pi}_{\mathrm{drbc}}}, d^{\pi_e}) \leq \frac{\gamma}{1-\gamma}\sqrt{L_{\mathrm{drbc}}(\widehat{\pi}_{\mathrm{drbc}}, \rho_{\mathrm{c}})} \overset{(a)}{\leq} \frac{\gamma\sqrt{\widehat{\varepsilon}_{\mathrm{drbc}}(\rho_{\mathrm{c}})}}{1-\gamma} \overset{(b)}{\leq} \rho_{\mathrm{c}},$$

where (a) follows with probability at least $1-\delta$ since $L_{\mathrm{drbc}}(\widehat{\pi}_{\mathrm{drbc}}, \rho_{\mathrm{c}}) \leq \widehat{\varepsilon}_{\mathrm{drbc}}(\rho_{\mathrm{c}})$ and (b) since ρ_{c} is in the feasibility set $\mathcal{U}_N = \{x \in (0,1] : \gamma^2\widehat{\varepsilon}_{\mathrm{drbc}}(x) \leq x(1-\gamma)^2\}$. The proof is now complete by following the proof analysis of Theorem 11. □

D.2.5 Proof of Theorem 12

Proof. We consider a class of deterministic MDPs $(\mathcal{S}, \mathcal{A}, P^o, \gamma, r)$ shown in Fig.D.1 for any $\gamma \in (0,1)$. We fix the initial state $s_0 = 0$. The state space is $\mathcal{S} = \{0,1,2\}$ and action space is $\mathcal{A} = \{a_1, a_2\}$. Reward for state 1 and action a_2 pair is $r(1,a_2) = 1$ and the reward is 0 for all other (s,a). We note that any positive scalar multiple of this reward function can be considered for this class of MDPs. Transition function P^o is deterministic, as indicated by the arrows.

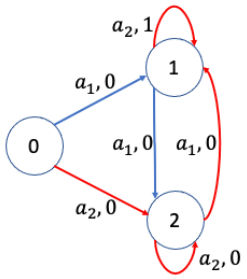

Figure D.1: Transitions and rewards for model P^o. The states $\{0,1,2\}$ are given inside the circles, and the actions $\{a_1, a_2\}$ and associated rewards are given on the corresponding transitions.

We denote state-distribution of model P^o for policy π starting at fixed state s_0 as $d^{\pi}_{s_0,P^o}$.

It is easy to see that taking action a_1 in states $0, 2$ and action a_2 in state 1 is the optimal policy π^*. This is obvious since action a_2 at state 1 is the only rewarded pair among all state-actions.

So, the optimal value we get is $V_{\pi^*}(0) = \gamma/(1-\gamma)$. We consider this optimal policy as the expert policy π_e. Recall the arbitrarily small $\varepsilon > 0$ from the theorem statement.

For simplicity with action space size 2, since KL loss function is an upper bound on 0-1 loss, we consider 0-1 loss for solving the optimization of all imitation learning algorithms in this theorem proof.

Recalling the definition of state-distribution, $d^{\pi}_{s_0,P}(s) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t \mathrm{Pr}_t(s \mid \pi, s_0, P)$, it is easy to see that $d^{\pi_e}_{s_0,P^o}(s=1) = \gamma = 1 - d^{\pi_e}_{s_0,P^o}(s=0)$ and $d^{\pi_e}_{s_0,P^o}(s=2) = 0$.

From BC algorithm Eq. (5.1), we have $\pi_{\mathrm{bc}} = \min_{\pi}(1-\gamma)\pi(a_2|s=0)$. This implies taking

$$\pi_{\mathrm{bc}}(\cdot|s=0) = \begin{cases} \varepsilon/(1-\gamma) & \text{if action } a_2 \\ 1-\varepsilon/(1-\gamma) & \text{if action } a_1 \end{cases}, \ \pi_{\mathrm{bc}}(a_2|s=1) = 1, \ \pi_{\mathrm{bc}}(a_2|s=2) = 1,$$

yields BC loss Eq. (5.1) as ε.

We consider $\rho = \gamma$ and the distance metric l_1 to find discrepancy between the state-distribution of policy π' and the expert policy. Since we are on a probability simplex of dimention 2, we have the following five valid vertices for the state-distribution:

$$d^{\pi}_{s_0,P^o}(s) = \begin{cases} (1,0,0) \\ (1-2\gamma, 2\gamma, 0) \\ (1-\gamma, 0, \gamma) \\ (1-\gamma, \gamma, 0) \\ (1-2\gamma, \gamma, \gamma) \end{cases},$$

where the tuple notation follows as $d^{\pi}_{s_0,P^o}(s) = (d^{\pi}_{s_0,P^o}(s=0), d^{\pi}_{s_0,P^o}(s=1), d^{\pi}_{s_0,P^o}(s=2))$. But, from Fig.D.1, it is obvious that $d^{\pi}_{s_0,P^o}(s=0) = 1-\gamma$ for any policy π, since after the time step 1 we never reach back state 0. So, the only two valid choices are $d^{\pi_1}_{s_0,P^o}(s) = (1-\gamma, 0, \gamma)$ and $d^{\pi_2}_{s_0,P^o}(s) = (1-\gamma, \gamma, 0)$ for some policy π_1 and π_2 (note that there can be multiple stochastic

policies satisfying this).

Now, from DR-BC algorithm Eq. (5.3), we have

$$\pi_{\text{drbc}} = \min_{\pi} \max\{(1-\gamma)\pi(a_2|s=0) + \gamma\pi(a_1|s=1), (1-\gamma)\pi(a_2|s=0) + \gamma\pi(a_2|s=2)\}.$$

This implies taking

$$\pi_{\text{drbc}}(\cdot|s=0) = \begin{cases} \varepsilon/(1-\gamma) & \text{if action } a_2 \\ 1-\varepsilon/(1-\gamma) & \text{if action } a_1 \end{cases}, \ \pi_{\text{drbc}}(a_2|s=1) = 1, \ \pi_{\text{drbc}}(a_1|s=2) = 1,$$

yields DR-BC loss Eq. (5.3) as ε.

Now, calculating the discounted value [6] we get

$$V_{\pi_{\text{bc}}}(s_0=0) = (1 - \frac{\varepsilon}{1-\gamma})\frac{\gamma}{1-\gamma} + \frac{\varepsilon}{1-\gamma} \cdot 0 \text{ and } V_{\pi_{\text{drbc}}}(s_0=0) = (1 - \frac{\varepsilon}{1-\gamma})\frac{\gamma}{1-\gamma} + \frac{\varepsilon}{1-\gamma} \cdot \frac{\gamma^2}{1-\gamma}.$$

Since $V_{\pi_e}(0) = \gamma/(1-\gamma)$, we can finally write

$$V_{\pi_{\text{bc}}}(0) = V_{\pi_e}(0) - \frac{\varepsilon\gamma}{(1-\gamma)^2} \text{ and } V_{\pi_{\text{drbc}}}(0) = V_{\pi_e}(0) - \frac{\varepsilon\gamma}{1-\gamma}.$$

This completes the proof of this theorem. □

D.3 Proof of Results in Section 5.3

D.3.1 Proof of Proposition 8

We provide the formal statement and its proof here.

Proposition 14. *For a fixed expert policy $\pi_e \in \Pi$, we have, for all $\pi \in \Pi$ and $\rho_{\text{r}} \in (0, 1]$,*

$$\max_{P \in \mathcal{M}} \mathbb{E}_{s \sim d_P^{\pi_e}}[D_{\text{KL}}(\pi(\cdot|s) \| \pi_e(\cdot|s))]$$
$$= \min_{\eta \in \mathbb{R}} \ \mathbb{E}_{s \sim d_{Po}^{\pi_e}}[(D_{\text{KL}}(\pi(\cdot|s) \| \pi_e(\cdot|s)) - \eta)_+] + (\sup_{s \in \mathcal{S}} D_{\text{KL}}(\pi(\cdot|s) \| \pi_e(\cdot|s)) - \eta)_+ \cdot \rho + \eta.$$

Proof. We rewrite $\max_{P\in\mathcal{M}} \mathbb{E}_{s\sim d_P^{\pi_e}}[D_{\mathrm{KL}}(\pi(\cdot|s)\|\pi_e(\cdot|s))]$ as

$$\max_{d_P^{\pi_e}: D_{\mathrm{TV}}(d_P^{\pi_e}, d_{Po}^{\pi_e})\le\rho_{\mathrm{r}}} \mathbb{E}_{s\sim d_P^{\pi_e}}[D_{\mathrm{KL}}(\pi(\cdot|s)\|\pi_e(\cdot|s))]$$

since $\mathcal{M} = \{P \in \mathcal{P} : D_{\mathrm{TV}}(d_P^{\pi_e}, d_{Po}^{\pi_e}) \le \rho_{\mathrm{r}}\}$. The result immediately follows from Lemma 33. □

D.3.2 Proof of Theorem 14

We present a few results needed for proving Theorem 14.

First, we formally show that when two models are close, then their state-distributions are close under the same roll-out policy.

Lemma 35. *Consider any policy π and $P \in \mathcal{P}$. We have $D_{\mathrm{TV}}(d_P^\pi, d_{P^o}^\pi) \le \gamma\rho_r'/(1-\gamma)$.*

Proof. By definition, since $P \in \mathcal{P}$, we have $D_{\mathrm{TV}}(P_{s,a}, P^o_{s,a}) \le \rho_r'$ for all $(s,a) \in \mathcal{S}\times\mathcal{A}$. We denote matrices $\mathbb{P}_\pi, \mathbb{P}^o_\pi \colon \mathcal{S}\times\mathcal{S} \to [0,1]$ with $\mathbb{P}_\pi(s',s) = \sum_{a\in\mathcal{A}} \pi(a|s) P_{s,a}(s')$ and $\mathbb{P}^o_\pi(s',s) = \sum_{a\in\mathcal{A}} \pi(a|s) P^o_{s,a}(s')$. Now, we can write

$$d_P^\pi = (1-\gamma)\sum_{t=0}^\infty \gamma^t \mathrm{Pr}_t(s \mid \pi, s_0\sim\mu, P) = (1-\gamma)\sum_{t=0}^\infty (\gamma\mathbb{P}_\pi)^t\mu \quad \text{and}$$
$$d_{P^o}^\pi = (1-\gamma)\sum_{t=0}^\infty \gamma^t \mathrm{Pr}_t(s \mid \pi, s_0\sim\mu, P^o) = (1-\gamma)\sum_{t=0}^\infty (\gamma\mathbb{P}^o_\pi)^t\mu.$$

Denoting $\mathbb{P}_{t,\pi} = \mathbb{P}_\pi^t\mu, \mathbb{P}^o_{t,\pi} = (\mathbb{P}^o_\pi)^t\mu$, from triangle inequality we further get

$$\|d_P^\pi - d_{P^o}^\pi\|_1 \le (1-\gamma)\sum_{t=0}^\infty \gamma^t \left\|\mathbb{P}_{t,\pi} - \mathbb{P}^o_{t,\pi}\right\|_1. \tag{D.7}$$

Intuitively, $\mathbb{P}_{t,\pi}(\mathbb{P}^o_{t,\pi})$ is state distribution resulting from π evolving in the model $P(P^o)$ at time step t with μ as the initial state distribution. We now bound $\left\|\mathbb{P}_{t,\pi} - \mathbb{P}^o_{t,\pi}\right\|_1$ for $t \ge 0$ in a recursive approach. From basic Markov chain theory [183] for any $t \ge 0$, we have

$$\left\|\mathbb{P}_{t,\pi} - \mathbb{P}^o_{t,\pi}\right\|_1 = \sum_{s'} |\mathbb{P}_{t,\pi}(s') - \mathbb{P}^o_{t,\pi}(s')|$$

$$
\begin{aligned}
&= \sum_{s'} |\sum_{s,a} (\mathbb{P}_{t-1,\pi}(s) P(s'|s,a) - \mathbb{P}^o_{t-1,\pi}(s) P^o(s'|s,a)) \pi(a|s)| \\
&= \sum_{s'} |\sum_{s,a} (P(s'|s,a)(\mathbb{P}_{t-1,\pi}(s) - \mathbb{P}^o_{t-1,\pi}(s)) + \mathbb{P}^o_{t-1,\pi}(s)(P(s'|s,a) - P^o(s'|s,a))) \pi(a|s)| \\
&\leq \sum_{s} |\mathbb{P}_{t-1,\pi}(s) - \mathbb{P}^o_{t-1,\pi}(s)| \sum_{a} \pi(a|s) \sum_{s'} P(s'|s,a) \\
&\qquad + \sum_{s} \mathbb{P}^o_{t-1,\pi}(s) \sum_{a} \pi(a|s) \sum_{s'} |P(s'|s,a) - P^o(s'|s,a)| \\
&= \left\| \mathbb{P}_{t-1,\pi} - \mathbb{P}^o_{t-1,\pi} \right\|_1 + \sum_{s} \mathbb{P}^o_{t-1,\pi}(s) \sum_{a} \pi(a|s) \sum_{s'} |P(s'|s,a) - P^o(s'|s,a)| \\
&\leq \left\| \mathbb{P}_{t-1,\pi} - \mathbb{P}^o_{t-1,\pi} \right\|_1 + 2\rho'_r,
\end{aligned}
$$

where the last inequality holds since $D_{\mathrm{TV}}(P_{s,a}, P^o_{s,a}) = (1/2) \left\| P_{s,a} - P^o_{s,a} \right\|_1 \leq \rho'_r$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$. By recursion, we have $\left\| \mathbb{P}_{t,\pi} - \mathbb{P}^o_{t,\pi} \right\|_1 \leq 2\rho'_r t$. Recall from algebra that $\sum_{t=0}^{\infty} \gamma^t t = \gamma/(1-\gamma)^2$. Combining this with Eq. (D.7) completes the proof. □

Now we state a result which extends the performance difference lemma [106, Lemma 1.16] notion for robust MDPs.

Lemma 36 (Robust Performance Difference Lemma)**.** *For any π', π policies, we get*

$$
V^{\mathrm{rob}}_{\pi} - V^{\mathrm{rob}}_{\pi'} \leq \frac{1}{1-\gamma} \cdot \max_{P \,:\, D(d^{\pi}_{P}, d^{\pi}_{Po}) \leq \rho_r} \mathbb{E}_{s \sim d^{\pi}_{P}} \left[\sum_{a} (\pi(a|s) - \pi'(a|s)) Q^{\mathrm{rob}}_{\pi'}(s,a) \right].
$$

Proof. We first define few useful notations for this proof. The *robust model* $P^{\mathrm{rob},\pi}$ for every π is as follows:

$$
P^{\mathrm{rob},\pi}_{s,a} = \operatorname*{argmin}_{P_{s,a} \in \mathcal{P}_{s,a}} P_{s,a}^{\top} V^{\mathrm{rob}}_{\pi}, \quad (s,a) \in \mathcal{S} \times \mathcal{A}.
$$

We call V^P_{π} as the value function for policy π under the model P. Now we can write $V^{\mathrm{rob}}_{\pi'} = V^{P^{\mathrm{rob},\pi'}}_{\pi'}$.

Fix $s_0 \sim \mu$. For any π', π policies, we have

$$
V^{\mathrm{rob}}_{\pi}(s_0) - V^{\mathrm{rob}}_{\pi'}(s_0) \overset{(a)}{\leq} V^{P^{\mathrm{rob},\pi'}}_{\pi}(s_0) - V^{\mathrm{rob}}_{\pi'}(s_0)
$$

$$\overset{(b)}{=} V_{\pi}^{P^{\mathrm{rob},\pi'}}(s_0) - V_{\pi'}^{P^{\mathrm{rob},\pi'}}(s_0)$$
$$\overset{(c)}{=} \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_{s_0}^{\pi,P^{\mathrm{rob},\pi'}}}\left[\sum_a(\pi(a|s)-\pi'(a|s))Q_{\pi'}^{\mathrm{rob}}(s,a)\right],$$

where (a) follows since by definition of $V_{\pi}^{\mathrm{rob}}(s_0)$ we have $V_{\pi}^{\mathrm{rob}}(s_0) \leq V_{\pi}^{P^{\mathrm{rob},\pi'}}(s_0)$, and (b) follows from definition of $P^{\mathrm{rob},\pi'}$ yielding $V_{\pi'}^{\mathrm{rob}}(s_0) = V_{\pi'}^{P^{\mathrm{rob},\pi'}}(s_0)$. Observe $P^{\mathrm{rob},\pi'} \in \mathcal{P}$. Now taking expectation on $s_0 \sim \mu$ with Lemma 35 completes the proof of this result and it only remains to show (c).

For (c), first denote $\mathcal{T}_{\pi,\pi'}(s) = (s_t, a_t)_{t\geq 0}$ trajectory generated from rolling policy π from the initial state s_0 under the robust model $P^{\mathrm{rob},\pi'}$. Now,

$$\begin{aligned}
V_{\pi}^{P^{\mathrm{rob},\pi'}}(s_0) - V_{\pi'}^{P^{\mathrm{rob},\pi'}}(s_0) &= \mathbb{E}_{\mathcal{T}_{\pi,\pi'}(s)}[\sum_t \gamma^t r(s_t,a_t)] - V_{\pi'}^{P^{\mathrm{rob},\pi'}}(s_0) \\
&\overset{(d)}{=} \mathbb{E}_{\mathcal{T}_{\pi,\pi'}(s)}[\sum_t \gamma^t (r(s_t,a_t) + \gamma V_{\pi'}^{P^{\mathrm{rob},\pi'}}(s_{t+1}) - V_{\pi'}^{P^{\mathrm{rob},\pi'}}(s_t)] \\
&\overset{(e)}{=} \mathbb{E}_{\mathcal{T}_{\pi,\pi'}(s)}[\sum_t \gamma^t (Q_{\pi'}^{\mathrm{rob}}(s_t,a_t) - V_{\pi'}^{\mathrm{rob}}(s_t)] \\
&\overset{(f)}{=} \frac{1}{1-\gamma}\mathbb{E}_{s'\sim d_{s_0}^{\pi',P^{\mathrm{rob},\pi'}}} \sum_{a'} \pi(a'|s')(Q_{\pi'}^{\mathrm{rob}}(s',a')) \\
&\quad - \frac{1}{1-\gamma}\mathbb{E}_{s'\sim d_{s_0}^{\pi',P^{\mathrm{rob},\pi'}}} \sum_{a'} \pi'(a'|s')(Q_{\pi'}^{\mathrm{rob}}(s',a')) \\
&= \frac{1}{1-\gamma}\mathbb{E}_{s'\sim d_{s_0}^{\pi',P^{\mathrm{rob},\pi'}}}\left[\sum_{a'}(\pi(a'|s')-\pi'(a'|s'))(Q_{\pi'}^{\mathrm{rob}}(s',a'))\right],
\end{aligned}$$

where (d) follows by recursion, (e) follows since $Q_{\pi'}^{\mathrm{rob}}(s,a) = r(s,a) + \gamma(P_{s,a}^{\mathrm{rob},\pi'})^{\top} V_{\pi'}^{P^{\mathrm{rob},\pi'}}$, and (f) from $V_{\pi'}^{\mathrm{rob}}(s) = \sum_a \pi'(a|s) Q_{\pi'}^{\mathrm{rob}}(s,a)$. This finally proves (c). □

Proof of Theorem 14. We start by Lemma 36 with $\pi' = \pi_{\mathrm{drbc}}$ and $\pi = \pi_e$. We get

$$\begin{aligned}
V_{\pi_e}^{\mathrm{rob}} - V_{\pi'}^{\mathrm{rob}} &\leq \frac{1}{1-\gamma} \max_{P\,:\,D(d_P^{\pi_e}, d_{Po}^{\pi_e})\leq \rho_r} \mathbb{E}_{s\sim d_P^{\pi_e}}\left[\sum_a(\pi_e(a|s)-\pi'(a|s))Q_{\pi'}^{\mathrm{rob}}(s,a)\right] \\
&\overset{(a)}{\leq} \frac{1}{1-\gamma} \max_{P\,:\,D(d_P^{\pi_e}, d_{Po}^{\pi_e})\leq \rho_r} \mathbb{E}_{s\sim d_P^{\pi_e}}\left[\|\pi_e(\cdot|s)-\pi'(\cdot|s)\|_1 \left\|Q_{\pi'}^{\mathrm{rob}}(s,\cdot)\right\|_{\infty}\right]
\end{aligned}$$

$$\overset{(b)}{\leq} \frac{2}{(1-\gamma)^2} \max_{P\,:\,D(d^{\pi_e}_{P}, d^{\pi_e}_{P^o}) \leq \rho_r} \mathbb{E}_{s\sim d^{\pi_e}_{P}} \left[\sqrt{D_{\mathrm{KL}}(\pi'(\cdot|s) \| \pi_e(\cdot|s))} \right]$$
$$\overset{(c)}{\leq} \frac{2}{(1-\gamma)^2} \sqrt{\max_{P\,:\,D(d^{\pi_e}_{P}, d^{\pi_e}_{P^o}) \leq \rho_r} \mathbb{E}_{s\sim d^{\pi_e}_{P}} \left[D_{\mathrm{KL}}(\pi'(\cdot|s) \| \pi_e(\cdot|s)) \right]},$$

where (a) follows from Holder's inequality, (b) from Pinsker's inequality [180] and the fact that $\left\| Q^{\mathrm{rob}}_{\pi'}(s,\cdot) \right\|_\infty \leq 1/(1-\gamma)$ for any π', and (c) from Jensen's inequality [180]. The proof of this result is complete since $L_{\mathrm{drbc}}(\pi_{\mathrm{drbc}}, \rho_r) = \varepsilon_{\mathrm{drbc}}(\rho_r)$. □

D.3.3 Proof of Theorem 15

Proof. We consider a class of deterministic MDPs $(\mathcal{S}, \mathcal{A}, P^o, \gamma, r)$ shown in Fig.D.2 to be the nominal model for any $\gamma \in (0.01, 1)$. We fix an initial state $s_0 = 0$. The state space is $\mathcal{S} = \{0, 1\}$ and action space is $\mathcal{A} = \{a_l, a_r\}$, where a_l denotes 'move left' and a_r denotes 'move right' action. Reward for state 1 and action a_r pair is $r(1, a_r) = 1$, for state 0 and action a_r pair is $r(0, a_r) = -100\gamma/99$, and the reward is 0 for all other (s, a). We note that any positive scalar multiple of this reward function can be considered for this class of MDPs. Transition function P^o is deterministic, as indicated by the arrows. We consider deterministic model P', as shown in Fig.D.3, as the perturbed model of P^o. Thus the uncertainty set $\mathcal{P}$ is $\{P^o, P'\}$.

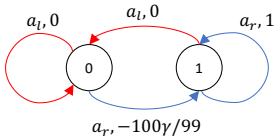

Figure D.2: Transitions and rewards corresponding to the nominal model P^o. The states $\{0, 1\}$ are given inside the circles, and the actions $\{a_l, a_r\}$ and associated rewards are given on the corresponding transitions.

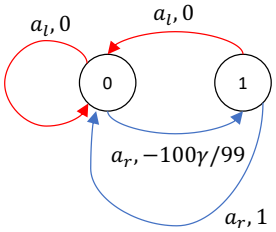

Figure D.3: Transitions and rewards corresponding to the model P'.

We denote state-distribution of model P^o for policy π starting at fixed state s_0 as $d^{\pi}_{s_0, P^o}$. Similarly for model P' it is $d^{\pi}_{s_0, P'}$.

It is easy to see that taking action a_r in any state is the non-robust optimal policy π^o corresponding to the nominal model P^o. This is obvious if for state $s = 1$. For $s = 0$, notice that taking action a_l will give a value zero and taking action a_r will give a value $\frac{\gamma}{1-\gamma} - \frac{100\gamma}{99}$. Since $\gamma > 0.01$, taking action a_r will give a positive value and hence is optimal. So, we get

$$V_{\pi^o,P^o}(0) = \frac{\gamma}{1-\gamma} - \frac{100\gamma}{99}.$$

Let $\varepsilon > 0$ be arbitrarily small. Let us consider the following expert policy

$$\pi_e(\cdot|s=0) = \pi_e(\cdot|s=1) = \begin{cases} \varepsilon/2 & \text{if action } a_l, \\ 1-\varepsilon/2 & \text{if action } a_r. \end{cases}$$

Recall Eq. (5.2). For simplicity with action space size 2, since KL loss function is an upper bound on 0-1 loss, we consider 0-1 loss for solving the optimization of all imitation learning algorithms in this theorem proof. It is easy to see that $L_{\text{il}}(\pi^o) = \varepsilon$ and for all other policies π (including stochastic) we have $L_{\text{il}}(\pi) > \varepsilon$. Thus, the imitation learning algorithm selects $\pi_{\text{il}} = \pi^o$ as its candidate.

We will now compute the optimal non-robust value from state 0 of model P'. Since it is enough to consider deterministic policies [6], we have

$$\begin{aligned} \max_{\pi} V_{\pi,P'}(0) = \max\{&V_{(\pi(0)=a_r,\pi(1)=a_r),P'}(0),\ V_{(\pi(0)=a_l,\pi(1)=a_l),P'}(0), \\ &V_{(\pi(0)=a_r,\pi(1)=a_l),P'}(0),\ V_{(\pi(0)=a_l,\pi(1)=a_r),P'}(0)\} \\ \overset{(a)}{=} \max\{&-\frac{\gamma}{99(1-\gamma^2)},\ 0,\ -\frac{100\gamma}{99(1-\gamma^2)},\ 0\} = 0, \end{aligned} \tag{D.8}$$

where (a) follows by using the Bellman recursive equations [6]:

$$V_{\pi^o,P'}(0) = -\frac{100\gamma}{99} + \gamma + \gamma^2 V_{\pi^o,P'}(0) \text{ and } V_{(\pi(0)=a_r,\pi(1)=a_l),P'}(0) = -\frac{100\gamma}{99} + \gamma^2 V_{(\pi(0)=a_r,\pi(1)=a_l),P'}(0).$$

We finally have

$$\max_{\pi} V_{\pi,P'}(0) - V_{\pi_{\text{il}},P'}(0) = \frac{\gamma}{99(1-\gamma^2)} \geq \frac{\gamma}{198(1-\gamma)},$$

where the inequality follows since $1+\gamma \leq 2$. Thus, setting $c = \gamma^2/198$ and $\gamma_o = 0.01$, completes first part of the theorem.

Now we prove the second part of the theorem. We now consider $s_0 = 1$ and expert policy π_e as taking action a_l in state 0 and a_r in state 1. Recall the arbitrarily small $\varepsilon > 0$ from the theorem statement.

Recalling the definition of state-distribution, $d^{\pi}_{s_0,P}(s) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t \text{Pr}_t(s \mid \pi, s_0, P)$, it is easy to see that $d^{\pi_e}_{s_0,P^o}(s=0) = 0 = 1 - d^{\pi_e}_{s_0,P^o}(s=1)$ and $d^{\pi_e}_{s_0,P'}(s=0) = \gamma = 1 - d^{\pi_e}_{s_0,P'}(s=1)$.

From BC algorithm Eq. (5.1), we have $\pi_{\text{bc}} = \min_{\pi}\pi(a_l|s=1)$. This implies π_{bc} taking action a_r in both states yields zero BC loss. Similarly, from DR-BC algorithm Eq. (5.6), we have $\pi_{\text{drbc}} = \min_{\pi}\max\{\pi(a_l|s=1), \gamma\pi(a_r|s=0) + (1-\gamma)\pi(a_l|s=1)\}$. This implies taking

$$\pi_{\text{drbc}}(a_l|s=0) = 1,\ \pi_{\text{drbc}}(\cdot|s=1) = \begin{cases} \varepsilon & \text{if action } a_l, \\ 1-\varepsilon & \text{if action } a_r, \end{cases}$$

yields DR-BC loss Eq. (5.6) as ε.

Following similar steps as in Eq. (D.8), the robust value of π_{bc} at state $s_0 = 1$ is given by

$$V^{\text{rob}}_{\pi_{\text{bc}}}(1) = \min\{V_{\pi_{\text{bc}},P^o}(0), V_{\pi_{\text{bc}},P'}(0)\} = 1 - \gamma^2/(99(1-\gamma^2)),$$

the robust value of π_{drbc} at state $s_0 = 1$ is given by

$$V^{\text{rob}}_{\pi_{\text{drbc}}}(1) = \min\{V_{\pi_{\text{drbc}},P^o}(0), V_{\pi_{\text{drbc}},P'}(0)\} = \min\{\frac{1-\varepsilon}{1-(1-\varepsilon)\gamma}, 1-\varepsilon\} = 1-\varepsilon,$$

and the robust value of π_e at state $s_0 = 1$ is given by

$$V^{\text{rob}}_{\pi_e}(1) = \min\{V_{\pi_e,P^o}(0), V_{\pi_e,P'}(0)\} = \min\{\frac{1}{1-\gamma}, 1\} = 1.$$

We finally have $V^{\text{rob}}_{\pi_e}(1) - V^{\text{rob}}_{\pi_{\text{drbc}}}(1) \leq \varepsilon/(1-\gamma)$ and

$$V^{\text{rob}}_{\pi_e}(1) - V^{\text{rob}}_{\pi_{\text{bc}}}(1) = \frac{\gamma^2}{99(1-\gamma^2)} \geq \frac{\gamma^2}{198(1-\gamma)},$$

where the inequality follows since $1 + \gamma \leq 2$. Thus, setting $c = \gamma^2/198$ and $\gamma_o = 0.01$, completes the proof of this theorem. □

D.4 Experiment Details

In this section, we provide more details of our DR-BC algorithm. We include the exact derivation of our empirical algorithm. We also present all the simulation configurations, including dataset generation, model hyperparameters, experiment equipment, etc. We provide our code in an **anonymous GitHub repository**: https://github.com/ferocious-cheetah/DRBC.

We would like to mention the software and hardware used in simulations. We used three Lambda Stack computers with the the operating system Ubuntu 18.04.6 LTS. The hardware configurations include: processor is AMD Threadripper 3960X (24 Cores, 3.80 GHz); GPUs are 2x RTX 2080 Ti; memory is 128GB RAM; Operating System Drive is 1 TB SSD (NVMe); and Data Drive is 4TB HDD. The software are but not limited to PyTorch, Caffe, CUDA, cuDNN, Numpy, Matplotlib. A detailed list of software dependencies will be included in our GitHub repository.

D.4.1 DR-BC Practical Algorithm

The practical version of our DR-BC algorithm can be found in Algorithm 8. We now explain how we derived the practical algorithm.

(1) Specializing Algorithm 4 to total variation uncertainty set. We rely on the Lemma 33.

(2) Solving the optimization problem in Equation D.1. The following proposition shows that with careful analysis, we can further reduce the total variation dual formulation to a scalar opti-

Algorithm 8 DR-BC Practical Algorithm With Total Variation Uncertainty Set

1: **Input:** Expert demonstration data $\mathcal{D}_e = \{(s_i, a_i)_{i=1}^N\}$, radius of robustness ρ, minibatch size $N_{\mathcal{B}}$.
2: **Initialize:** Policy (actor) neural network π_θ with random parameters θ.
3: **for** $k = 1, \ldots, K$ **do**
4: Sample a minibatch $\mathcal{B}$ of size $N_{\mathcal{B}}$ from $\mathcal{D}_e$.
5: Calculate the empirical DRO loss:

$$L_{\mathrm{drbc}}(\pi_\theta, \rho) = \inf_{\eta \in [0,(1+\rho)\overline{L}]} \left\{ \left(\frac{1}{N_{\mathcal{B}}} \sum_{(s,a)\in\mathcal{B}} (\|a - \pi_\theta(s)\|_2^2 - \eta)_+ \right) + \left(\sup_{(s,a)\in\mathcal{B}} \|\pi_e(s) - \pi_\theta(s)\|_2^2 - \eta \right)_+ \cdot \rho + \eta \right\}.$$

6: $\theta \leftarrow \mathrm{argmin}_\theta \, L_{\mathrm{drbc}}(\pi_\theta, \rho)$.
7: **end for**
8: **Output policy:** π_θ

mization over a finite interval.

Proposition 15. *Suppose that we have deterministic policies π_e, π and a bounded action space $\mathcal{A}$. Let the loss l be chosen as the squared L2 loss:*

$$l(\pi(s), \pi_e(s)) = \|\pi(s) - \pi_e(s)\|_2^2 .$$

Further, denote $\overline{L} = \sup_{s\in\mathcal{S}: d^{\pi_e}(s)>0} \|\pi(s) - \pi_e(s)\|_2^2$. Then the dual reformulation in Lemma 33 can be further rewritten as

$$L_{\mathrm{drbc}}(\pi, \rho) = \inf_{\eta \in [0,(1+\rho)\overline{L}]} \left\{ \left(\frac{1}{N_{\mathcal{B}}} \sum_{(s,a)\in\mathcal{B}} (\|a - \pi_\theta(s)\|_2^2 - \eta)_+ \right) + \left(\sup_{s\in\mathcal{S}: d^{\pi_e}(s)>0} \|\pi_e(s) - \pi_\theta(s)\|_2^2 - \eta \right)_+ \cdot \rho + \eta \right\}. \tag{D.9}$$

Proof. Denote the function

$$h(\eta) = \mathbb{E}_{s\sim d^{\pi_e}}[(\|\pi(s) - \pi_e(s)\|_2^2 - \eta)_+] + (\overline{L} - \eta)_+ \cdot \rho + \eta.$$

First, assuming $\eta < 0$, we have

$$h(\eta) = \mathbb{E}_{s\sim d^{\pi_e}}[(\|\pi(s) - \pi_e(s)\|_2^2 - \eta)] + (\overline{L} - \eta) \cdot \rho + \eta$$
$$= \mathbb{E}_{s\sim d^{\pi_e}}[(\|\pi(s) - \pi_e(s)\|_2^2)] + (\overline{L} - \eta) \cdot \rho.$$

Now, assuming $\eta = 0$, we then have

$$h(0) = \mathbb{E}_{s\sim d^{\pi_e}}[(\|\pi(s) - \pi_e(s)\|_2^2)] + \overline{L}\rho.$$

Lastly, we need to find an upper bound. Consider the following:

$$h((1+\rho)\overline{L}) = \mathbb{E}_{s\sim d^{\pi_e}}[(\|\pi(s) - \pi_e(s)\|_2^2 - (1+\rho)\overline{L})_+] + (\overline{L} - (1+\rho)\overline{L})_+ \cdot \rho + (1+\rho)\overline{L} = (1+\rho)\overline{L}.$$

It is clear that $h(0) \leq h(\eta)$, $\forall \eta < 0$, and $h(0) \leq h((1+\rho)\overline{L})$. Since h is convex in the dual variable η, it is sufficient to consider $\eta \in [0, (1+\rho)\overline{L}]$ for the above optimization problem. Note that since the action space $\mathcal{A}$ is assumed to be bounded, we have $\overline{L} < \infty$. Lastly, approximating the expectation using sample mean over minibatches gives us the result. □

Lemma 33 and Proposition 15 greatly simplify the optimization problem in the Equation 5.4 of Algorithm 4 into its practical form (Step 5 in Algorithm 8). Now we focus on describing our practical algorithm (Algorithm 8). For all the hyperparameters mentioned in the description below, we provide detailed records in the next section.

1. At initialization, we need an expert dataset $\mathcal{D}_e$ of size N, some radius of our uncertainty set ρ, and a prescribed minibatch size. All of the three are treated as hyperparameters by us. We also need to initialize a neural network π_θ which is our policy (actor) with random parameters θ.

2. In each iteration k, we need to minimize a single-variable η scalar function where the variable η only takes values in a finite real interval. Note that all OpenAI Gym MuJoCo environments

have bounded action space. This suggests that the $\overline{L}$ can be further replaced by squared L2 distance between the upper bound and lower bound on the actions. Such specifications can be found on their website [71]. We use the minimization solver from the powerful optimization libraries in SciPy [88]. In particular, we use the `SLSQP` method [184] with the bounds as prescribed in Proposition 15. Note that in Step 5 of Algorithm 8, we estimate the supremum over the whole state space $\mathcal{S}$ using the supremum over the current minibatch.

3. Our policy (actor) is optimized based on the L_{drbc} loss using ADAM [175] in Step 6 with the learning rate l. We also provide the option to decay the learning rate. This creates three new hyperparameters: a boolean flag `lr_decay` signaling to decay or not, decay rate of the learning rate scheduler γ and decay frequency κ.

4. In all our simulations, we keep the maximum train steps K (Step 4 in Algorithm 8) to be two millions.

D.4.2 DR-BC Model Details

Now we specify all the hyperparameters used in our simulations in Table D.1. Note that "lr" is short for learning rate and the robust parameter ρ only applies to the DR-BC algorithm. We want to point out that the size of the expert dataset is determined by the difficulty and complexity of an environment. `Hopper-v3` is the easiest among the three. With only 2000 expert samples, which is roughly two expert trajectories, we are able to show the mitigation of covariate shift and the robustness under model perturbations in DR-BC. The hyperparameters included above are picked according to the results of extensive grid searches over the hyperparameter space.

D.4.3 Expert Demonstration Generation

We select the state-of-art, non-robust, online RL algorithm TD3 [110] to generate the expert demonstrations. To further facilitate reproducibility, we choose to use the pre-trained policies from the RL Baselines3 Zoo repositories [144] which is well-regarded in the reinforcement learning research community. The data generation utilities are also included in our repository (see Section D.4.2 for the link).

Environment	Expert dateset size N	Minibatch size $N_{\mathcal{B}}$	Policy π_θ hidden layers	Activation function	Max train steps K
`Hopper-v3`	2000	256	(256, 256)	Tanh	2000000
`HalfCheetah-v3`	3000	256	(256, 256)	Tanh	2000000
`Walker2d-v3`	6000	256	(256, 256)	Tanh	2000000

Environment	Robustness parameter ρ	Learning rate (lr) l	lr decay `lr_decay`	lr decay rate γ	lr decay freq. κ
`Hopper-v3`	0.2	1e-4	`True`	0.9	10000
`HalfCheetah-v3`	0.3	1e-5	`True`	0.9	10000
`Walker2d-v3`	0.5	1e-4	`True`	0.95	10000

Table D.1: Hyperparameters used in training the BC and DR-BC algorithms.

D.4.4 More Simulation Results

In this section, we continue the discussion we have in Section 5.4. Also, we include all validation results here. We perturb `Hopper-v3` by changing the model parameter *'gravity'*, *'thigh_joint_stiffness'*, *'joint_damping'*, and *'joint_frictionloss'*. Fig. D.4 shows that DR-BC is tenacious under model perturbations. For example, in the bottom left figure, when the *'thigh_joint_stiffness'* parameter is positive and increasing, a joint spring is created in the thigh of hopper and becomes stiffer. A non-robust policy such as BC cannot withstand such mismatch between the training and testing environments. Meanwhile, our DR-BC agent refuses to drop in performance.

In Fig. D.5, we perturb `HalfCheetah-v3` by changing: the model parameter *'gravity'*, the parameter *'joint_stiffness'* of all three joints in the back of the cheetah, the parameter *'joint_damping'* of all six joints, and finally the parameter *'joint_frictionloss'* of all six joints. BC performance precipitates facing severe perturbations in all joints at once. Meanwhile, DR-BC refuses to lose performance in wide ranges of perturbations.

We continue from the discussion in Section 5.4. In Fig. D.6, we preturb `Walker2d-v3` by changing: *'gravity'*, *'actuator_ctrlrange'* of all joints, *'foot_joint_damping'* of both foot joints, *'leg_joint_stiffness'* of both leg joints, and *'foot_joint_stiffness'* of both foot joints. We can see that DR-BC is exceptionally robust when *'joint_stiffness'* is perturbed in both foot joints. For such

wide range of perturbation, the performance of DR-BC only drops 500 in episodic reward averaged over 10 differently seeded games.

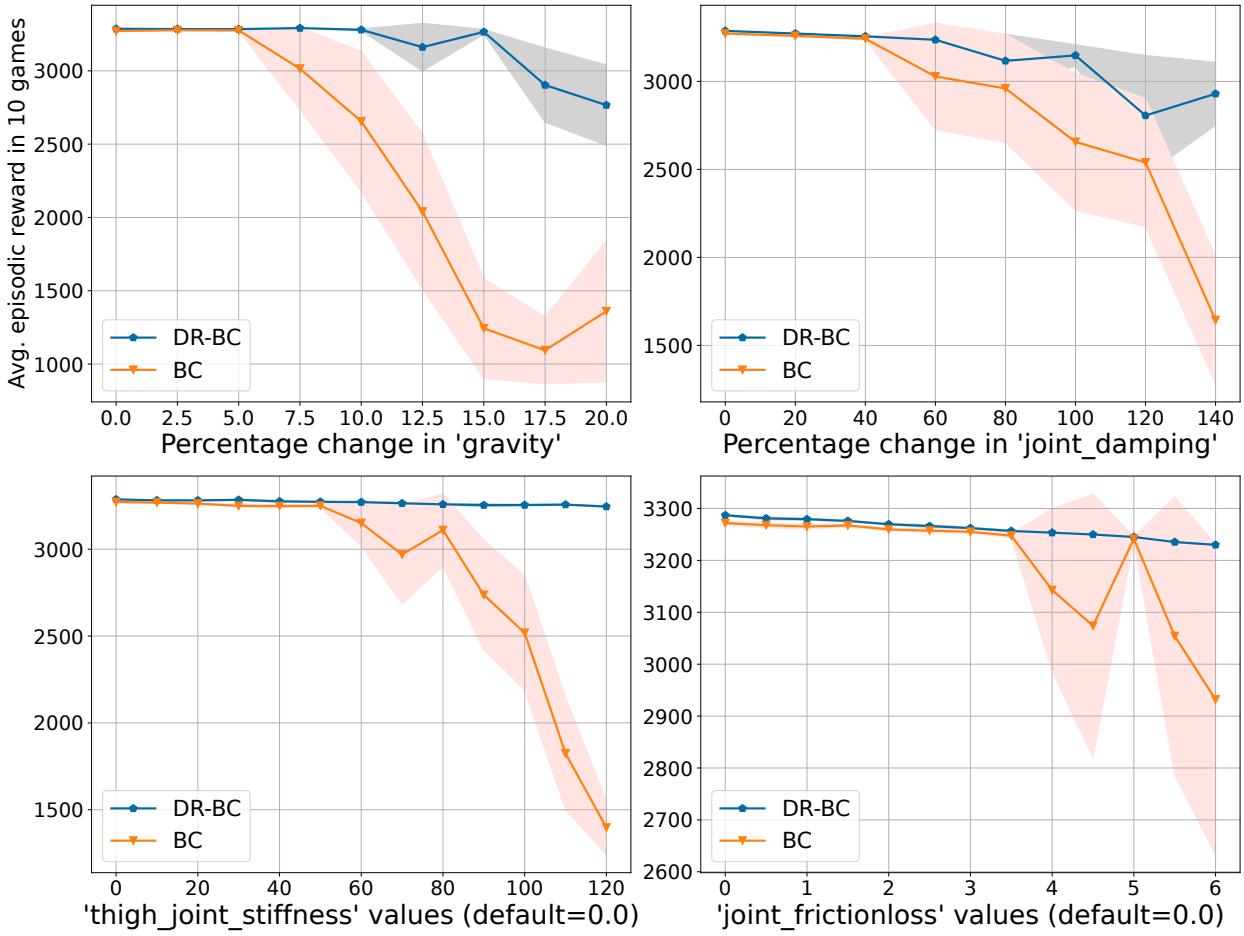

Figure D.4: `Hopper-v3` *perturbation results.* Average episodic reward on 10 differently seeded episodes. From left to right, the perturbations are in: *'gravity'*, *'joint_damping'*, *'thigh_joint_stiffness'*, and *'joint_frictionloss'* of all joints.

D.4.5 Details of Environment Perturbations

We include all the model parameters we used and their corresponding nominal values in Table D.2. The model parameter names should be self-explanatory. For example, *foot_joint_stiffness* means the parameter that controls the stiffness on the joint on the foot. Note that the parameter *actuator_ctrlrange* is actually a tuple in the form of $(-x, x)$. Hence, we describe them as the

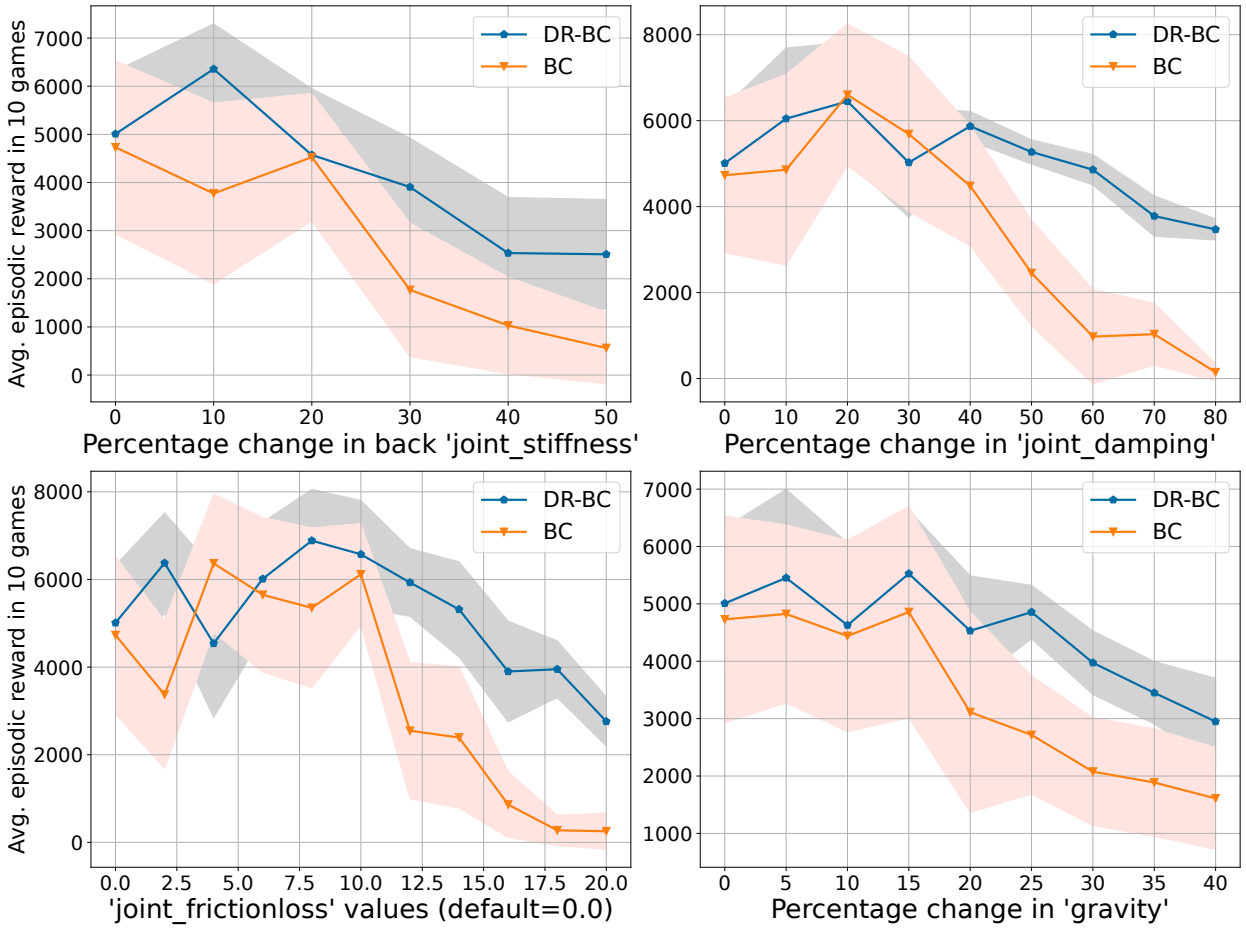

Figure D.5: `HalfCheetah-v3` *perturbation results.* Average episodic reward on 10 differently seeded episodes. From left to right and top to bottom, the perturbations are in: back *'joint_stiffness'*, *'joint_damping'* of all joints, *'joint_frictionloss'* of all joints, and the model parameter *'gravity'*.

range of actuator (controller). We provide all the source files containing the implementations of the perturbed environments through our GitHub repository. They are readily usable if one wants to use them to quantify the robustness of their policies. For more information regarding these model parameters or any other information regarding these environments, including those not covered in this paper, please refer to the *hopper.xml*, *halfcheetah.xml*, and *walker2d.xml* files in the environment assets of OpenAI Gym [71] MuJoCo which is accessible at OpenAI Gym repository MuJoCo section.

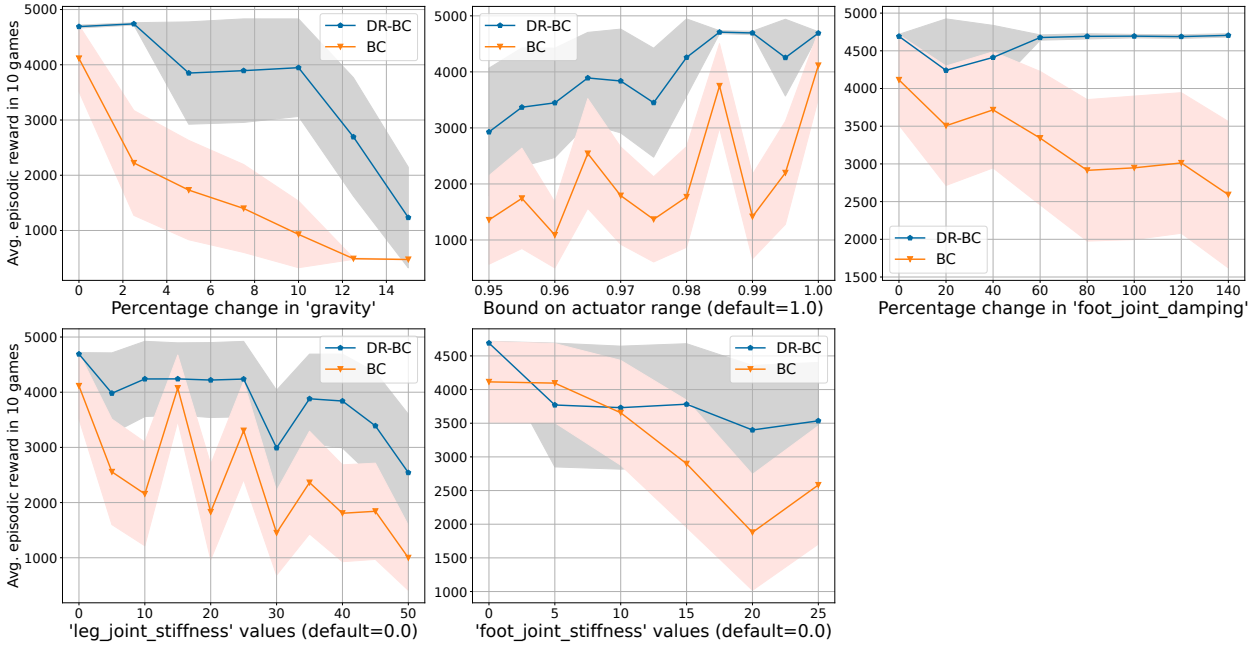

Figure D.6: `Walker2d-v3` *perturbation results.* Average episodic reward on 10 differently seeded episodes. From left to right and top to bottom, the perturbations are in: *'gravity'*, *'actuator_ctrlrange'* of all joints, *'foot_joint_damping'* of both foot joints, *'leg_joint_stiffness'* of both leg joints, and *'foot_joint_stiffness'* of both foot joints.

Environment	Model parameter	Nominal range/value
`Hopper-v3`	*gravity*	-9.81
	all *joint_frictionloss*	0
	all *actuator_ctrlrange*	$[-1, 1]$
	foot_joint_stiffness	0
	leg_joint_stiffness	0
	thigh_joint_stiffness	0
	all *joint_damping*	1.0
`HalfCheetah-v3`	*gravity*	-9.81
	all *joint_frictionloss*	0
	front *actuator_ctrlrange*	$[-1, 1]$
	back *actuator_ctrlrange*	$[-1, 1]$
	front *joint_stiffness* = (*fthigh_joint_stiffness*, *fshin_joint_stiffness*, *ffoot_joint_stiffness*)	$(180, 120, 60)$
	back *joint_stiffness* = (*bthigh_joint_stiffness*, *bshin_joint_stiffness*, *bfoot_joint_stiffness*)	$(240, 180, 120)$
	front *joint_damping* = (*fthigh_joint_damping*, *fshin_joint_damping*, *ffoot_joint_damping*)	$(4.5, 3.0, 1.5)$
	back *joint_damping* = (*bthigh_joint_damping*, *bshin_joint_damping*, *bfoot_joint_damping*)	$(6.0, 4.5, 3.0)$
`Walker2d-v3`	*gravity*	-9.81
	all *joint_frictionloss*	0
	left/right *thigh_actuator_ctrlrange*	$[-1, 1]$
	left/right *leg_actuator_ctrlrange*	$[-1, 1]$
	left/right *foot_actuator_ctrlrange*	$[-1, 1]$
	left/right *thigh_joint_stiffness*	0
	left/right *leg_joint_stiffness*	0
	left/right *foot_joint_stiffness*	0
	left/right *thigh_joint_damping*	0.1
	left/right *leg_joint_damping*	0.1
	left/right *foot_joint_damping*	0.1

Table D.2: Model parameters of `Hopper-v3`, `HalfCheetah-v3`, and `Walker2d-v3`.