Robust Reinforcement Learning using Offline Data

Kishan Panaganti[1], **Zaiyan Xu**[1], **Dileep Kalathil**[1], **Mohammad Ghavamzadeh**[2]
[1]Texas A&M University, [2]Google Research.
Emails: {kpb, zxu43 ,dileep.kalathil}@tamu.edu, ghavamza@google.com

Abstract

The goal of robust reinforcement learning (RL) is to learn a policy that is robust against the uncertainty in model parameters. Parameter uncertainty commonly occurs in many real-world RL applications due to simulator modeling errors, changes in the real-world system dynamics over time, and adversarial disturbances. Robust RL is typically formulated as a max-min problem, where the objective is to learn the policy that maximizes the value against the worst possible models that lie in an uncertainty set. In this work, we propose a robust RL algorithm called Robust Fitted Q-Iteration (RFQI), which uses only an offline dataset to learn the optimal robust policy. Robust RL with offline data is significantly more challenging than its non-robust counterpart because of the minimization over all models present in the robust Bellman operator. This poses challenges in offline data collection, optimization over the models, and unbiased estimation. In this work, we propose a systematic approach to overcome these challenges, resulting in our RFQI algorithm. We prove that RFQI learns a near-optimal robust policy under standard assumptions and demonstrate its superior performance on standard benchmark problems.

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 (Sünderhauf et al., 2018; Tobin et al., 2017; Peng et al., 2018).

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) (Iyengar, 2005; Nilim and El Ghaoui, 2005). 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 the uncertainty set makes the robust MDP and robust RL problems significantly more challenging than their non-robust counterparts.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

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 D.

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 (Antos et al., 2008; Bertsekas, 2011; Lange et al., 2012; Chen and Jiang, 2019; Xie and Jiang, 2020; Levine et al., 2020; Xie et al., 2021). Many recent works develop deep RL algorithms and heuristics for the offline RL problem, focusing on the algorithmic and empirical aspects (Fujimoto et al., 2019; Kumar et al., 2019, 2020; Yu et al., 2020; Zhang and Jiang, 2021). A number of theoretical work focus on analyzing the variations of Fitted Q-Iteration (FQI) algorithm (Gordon, 1995; Ernst et al., 2005), by identifying the necessary and sufficient conditions for the learned policy to be approximately optimal and characterizing the performance in terms of sample complexity (Munos and Szepesvári, 2008; Farahmand et al., 2010; Lazaric et al., 2012; Chen and Jiang, 2019; Liu et al., 2020; Xie et al., 2021). 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 Iyengar (2005); Nilim and El Ghaoui (2005). The RMDP problem has been analyzed extensively in the literature (Xu and Mannor, 2010; Wiesemann et al., 2013; Yu and Xu, 2015; Mannor et al., 2016; Russel and Petrik, 2019) providing computationally efficient algorithms, but these works are limited to the planning problem. Robust RL algorithms with provable guarantees have also been proposed (Lim et al., 2013; Tamar et al., 2014; Roy et al., 2017; Panaganti and Kalathil, 2021; Wang and Zou, 2021), 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 (Pinto et al., 2017; Derman et al., 2018, 2020; Mankowitz et al., 2020; Zhang et al., 2020a). 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 Zhou et al. (2021); Yang et al. (2021); Panaganti and Kalathil (2022) 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 P. However, generating data according to all models in 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 P?*
(ii) Optimization over the uncertainty set P: The robust Bellman operator (defined in (3)) involves a minimization over P, which is a significant computational challenge. Moreover, the uncertainty set P itself is unknown in the RL setting. *How do we solve the optimization over 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 P with an ex-

pectation 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 P. Following the same reformulation, we then show that the optimization problem over 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 X, we denote its cardinality as $|X|$. The set of probability distribution over X is denoted as $\Delta(X)$, and its power set sigma algebra as $\sigma(X)$. For any $x \in \mathbb{R}$, we denote $\max\{x, 0\}$ as $(x)_+$. For any function $f : S \times A \to \mathbb{R}$, state-action distribution $\rho \in \Delta(S \times A)$, and real number $p \geq 1$, the ρ-weighted p-norm of f is defined as $\|f\|_{p,\rho} = \mathbb{E}_{s,a \sim \rho}[|f(s,a)|^p]^{1/p}$.

2 Preliminaries

A Markov Decision Process (MDP) is a tuple $(S, A, r, P, \gamma, d_0)$, where S is the state space, A is the action space, $r : S \times A \to \mathbb{R}$ is the reward function, $\gamma \in (0, 1)$ is the discount factor, and $d_0 \in \Delta(S)$ 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 $|S|$ and $|A|$ are finite but can be arbitrarily large. We will also assume that $r(s,a) \in [0, 1]$, for all $(s,a) \in S \times A$, without loss of generality. A policy $\pi : S \to \Delta(A)$ is a conditional distribution over actions given a state. The value function V^{π}_P and the state-action value function Q^{π}_P of a policy π for an MDP with model P are defined as

$$V^{\pi}_P(s) = \mathbb{E}_{\pi,P}\Big[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \,|\, s_0 = s\Big], \quad Q^{\pi}_P(s,a) = \mathbb{E}_{\pi,P}\Big[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \,|\, s_0 = s, a_0 = a\Big],$$

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 $\pi^*_P = \arg\max_\pi V^{\pi}_P$. The optimal state-action value function is given by $Q^*_P = \max_\pi Q^{\pi}_P$. The optimal policy can be obtained as $\pi^*_P(s) = \arg\max_a Q^*_P(s,a)$. The discounted state-action occupancy of a policy π for an MDP with model P, denoted as $d^{\pi}_P \in \Delta(S \times A)$, 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* (Iyengar, 2005). 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 $M = (S, A, r, \mathcal{P}, \gamma, d_0)$, where the uncertainty set $\mathcal{P}$ is typically defined as

$$\mathcal{P} = \bigotimes_{(s,a) \in S \times A} \mathcal{P}_{s,a}, \quad \text{where } \mathcal{P}_{s,a} = \{P_{s,a} \in \Delta(S) : D(P_{s,a}, P^o_{s,a}) \leq \epsilon\}, \tag{1}$$

$P^o = (P^o_{s,a}; (s,a) \in S \times A)$ is the *nominal model*, $D(\cdot, \cdot)$ is a distance metric between two probability distributions, and $\epsilon > 0$ 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 $\mathcal{P}$ (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 (Iyengar, 2005), where the neighborhood is defined using the distance metric $D(\cdot, \cdot)$ and radius ϵ. 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 corresponding to a policy

and the *optimal robust value function* V are defined as (Iyengar, 2005; Nilim and El Ghaoui, 2005)

$$V = \inf_{P \in \mathcal{P}} V_{;P}; \qquad V = \sup \inf_{P \in \mathcal{P}} V_{;P}: \tag{2}$$

The *optimal robust policy* is such that the robust value function corresponding to it matches the optimal robust value function, i.e., $V = V$. It is known that there exists a deterministic optimal policy (Iyengar, 2005) for the RMDP. The *robust Bellman operator* is defined as (Iyengar, 2005)

$$(TQ)(s;a) = r(s;a) + \inf_{P_{s;a} \in \mathcal{P}_{s;a}} \mathbb{E}_{s' \sim P_{s;a}}[\max_b Q(s';b)]: \tag{3}$$

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 $(s) = \arg\max_a Q(s;a)$ (Iyengar, 2005). 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 in the tabular setting with a known 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 $_P$ 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 $(s_i;a_i)$, $\in (\mathcal{S} \times \mathcal{A})$ 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 $\mathcal{F} = \{f : \mathcal{S} \times \mathcal{A} \to [0; 1/(1-)]\}$ to approximate Q_P. The typical FQI update is given by $f_{k+1} = \arg\min_{f \in \mathcal{F}} \sum_{i=1}^N (r(s_i;a_i) + \max_b f_k(s_i';b) - f(s_i;a_i))^2$, 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 (Szepesvári and Munos, 2005; Chen and Jiang, 2019; Liu et al., 2020).

3 Offline Robust Reinforcement Learning

The goal of an offline robust RL algorithm is to learn the optimal robust policy 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 $(s_i;a_i) \sim ; \in (\mathcal{S} \times \mathcal{A})$ 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 (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 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 $\mathcal{F} = \{f : \mathcal{S} \times \mathcal{A} \to [0; 1/(1-)]\}$ and $\mathcal{G} = \{g : \mathcal{S} \times \mathcal{A} \to [0; 2/(1-)]\}$. 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 1. 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 and the representation power of $\mathcal{F}$.

Assumption 1 (Concentratability)**.** *There exists a finite constant $C > 0$ such that for any $\in (\mathcal{S} \times \mathcal{A})$, we have $\| d_{;P^o} / \| \le C$.*

Assumption 1 states that the ratio of the distribution and the data generating distribution , $(s;a)/(s;a)$, is uniformly bounded. This assumption is widely used in the offline RL literature (Munos, 2003; Agarwal et al., 2019; Chen and Jiang, 2019; Wang et al., 2021; Xie et al., 2021) in many different forms. We borrow this assumption from Chen and Jiang (2019), where they used it for

non-robust offline RL. In particular, we note that the distribution is in the collection of discounted state-action occupancies on model P^o alone for the robust RL.

Assumption 2 (Approximate completeness)**.** *Let* 2 $(S$ $A)$ *be the data distribution. Then,* $\sup_{f\,2\,F} \inf_{f'\,2\,F} \|f' - \mathcal{T} f\|_{2;}^2$ $"_c$*:*

Assumption 2 states that the function class F is approximately closed under the robust Bellman operator $\mathcal{T}$. This assumption has also been widely used in the offline RL literature (Agarwal et al., 2019; Chen and Jiang, 2019; Wang et al., 2021; Xie et al., 2021).

One of the most important properties that the function class F should have is that there must exist a function $f' \,2\, F$ which well-approximates Q . This assumption is typically called *approximate realizability* in the offline RL literature. This is typically formalized by assuming $\inf_{f\,2\,F} \|f - \mathcal{T} f\|_{2;}^2$ $"_r$ (Chen and Jiang, 2019). It is known that the approximate completeness assumption and the concentratability assumption imply the realizability assumption (Chen and Jiang, 2019; Xie et al., 2021).

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 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.

4.1 Dual Reformulation of Robust Bellman Operator

One key challenge in directly using the standard definition of the optimal robust value function given in (2) or of the robust Bellman operator given in (3) for developing and analyzing robust RL algorithms is that both involve computing an expectation w.r.t. each model $P \,2\, \mathcal{P}$. 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 1. *Let* M *be an RMDP with the uncertainty set* $\mathcal{P}$ *specified by* (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* $Q : S$ A $!$ $[0; 1=(1$ $)]$*, the robust Bellman operator* $\mathcal{T}$ *given in* (3) *can be equivalently written as*

$$(\mathcal{T}Q)(s;a) = r(s;a) \inf_{2\,[0;\,\frac{2}{(1\,\,)}]} (\mathsf{E}_{s'\,\,P^o_{s;a}}[(\,\,V(s'))_+] + (\,\,\inf_{s''} V(s''))_+); \quad (4)$$

where $V(s) = \max_{a\,2\,A} Q(s;a)$*. Moreover, the inner optimization problem in* (4) *is convex in* .

This result mainly relies on Shapiro (2017, Section 3.2) and Duchi and Namkoong (2018, Proposition 1). Note that in (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) 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 3 (Fail-state)**.** *The RMDP* M *has a 'fail-state'* s_f*, such that* $r(s_f; a) = 0$ *and* $P_{s_f;a}(s_f) = 1;\ 8a\ 2\ A;\ 8P\ 2\ \mathcal{P}$.

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 3 immediately implies that $V_{;P}(s_f) = 0;\ 8P\ 2\ \mathcal{P}$, and hence V $(s_f) = 0$ and Q $(s_f; a) = 0;\ 8a\ 2\ A$. It is also straightforward to see that $Q_{k+1}(s_f; a) = 0;\ 8a\ 2\ A$, where Q_k's are the RQI iterates given by the robust Bellman update $Q_{k+1} = \mathcal{T} Q_k$ with the initialization $Q_0 = 0$. By the contraction property of $\mathcal{T}$, we have Q_k $!$ Q . So, under Assumption 3, without loss of generality, we can always keep $Q_k(s_f; a) = 0;\ 8a\ 2\ A$ 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 $f : S \quad A \; ! \; [0; 1=(1 \quad)]$ with $f(s_f; a) = 0$, the robust Bellman operator T is now given by

$$(T f)(s; a) = r(s; a) \quad \inf_{2[0; \frac{2}{(1 \quad)}]} (E_{s'} \quad P^{o}_{s;a}[(\quad \max_{a'} f(s'; a'))_+] \quad (1 \quad)): \quad (5)$$

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 (2) or of the robust Bellman operator given in (3) for developing and analyzing robust RL algorithms is that both involve an optimization over P. The dual reformulation given in (5) partially overcomes this challenge also, as the optimization over P is now replaced by a convex optimization over a scalar $2 [0; 2=((1 \quad))]$. However, this still requires solving an optimization for each $(s; a) \; 2 \; S \quad A$, 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 $(S \quad A; \quad (S \quad A); \;)$ and let $L^1(S \quad A; \quad (S \quad A); \;)$ be the set of all absolutely integrable functions defined on this space.[1] In other words, L^1 is the set of all functions $g : S \quad A \; ! \; C \quad \mathbb{R}$, such that $kgk_{1;}$ is finite. We set $C = [0; 2= \; (1 \quad)]$, anticipating the solution of the dual optimization problem (5). We also note is the data generating distribution which is a -finite measure.

For any given function $f : S \quad A \; ! \; [0; 1=(1 \quad)]$, we define the loss function $L_{\text{dual}}(; f)$ as

$$L_{\text{dual}}(g; f) = E_{s;a} \quad [E_{s'} \quad P^{o}_{s;a}[(g(s; a) \quad \max_{a'} f(s'; a'))_+] \quad (1 \quad)g(s; a)]; \quad 8g \; 2 \; L^1: \quad (6)$$

In the following lemma, we show that the scalar optimization over for each $(s; a)$ pair in (5) can be replaced by a single functional optimization w.r.t. the loss function L_{dual}.

Lemma 1. *Let L_{dual} be the loss function defined in (6). Then, for any function $f : S \quad A \; !$ $[0; 1=(1 \quad)]$, we have*

$$\inf_{g \, 2 \, L^1} L_{\text{dual}}(g; f) = E_{s;a} \quad \left[\inf_{2[0; \frac{2}{(1 \quad)}]} E_{s'} \quad P^{o}_{s;a} \quad \max_{a'} f(s'; a') \right]_+ \quad (1 \quad) \quad : \quad (7)$$

Note that the RHS of (7) has minimization over for each $(s; a)$ pair and minimization is inside the expectation $E_{s;a} \quad [ ]$. However, the LHS of (7) has a single functional minimization over $g \; 2 \; L^1$ 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 Rockafellar and Wets (2009, 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 (Shapiro, 2017; Duchi and Namkoong, 2018) (see Appendix A for more details).

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

$$\hat{b}_{\text{dual}}(g; f) = \frac{1}{N} \sum_{i=1}^{N} (g(s_i; a_i) \quad \max_{a'} f(s_i'; a'))_+ \quad (1 \quad)g(s_i; a_i): \quad (8)$$

Now, for any given f, we can find an approximately optimal dual function through the *empirical risk minimization* approach as $\inf_{g \, 2 \, L^1} \hat{b}_{\text{dual}}(g; f)$.

As we mentioned in Section 3, our offline robust RL algorithm is given an input function class $G = fg : S \quad A \; ! \; [0; 2=((1 \quad))]g$ 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 G. For this to be meaningful, G should have sufficient representation power. In particular, the result in Lemma 1 should hold approximately even if we replace the infimum over L^1 with infimum over G. One can see that this is similar to the realizability requirement for the function class F as described in Section 3. We formalize the representation power of G in the following assumption.

[1] 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.

Assumption 4 (Approximate dual realizability). *For all $f \in \mathcal{F}$, there exists a uniform constant ε_{dual} such that* $\inf_{g \in \mathcal{G}} L_{\mathrm{dual}}(g; f) \leq \inf_{g \in L^1} L_{\mathrm{dual}}(g; f) + \varepsilon_{dual}$

Using the above assumption, for any given $f \in \mathcal{F}$, we can find an approximately optimal dual function $\hat{g}_f \in \mathcal{G}$ through the *empirical risk minimization* approach as $\hat{g}_f = \arg\min_{g \in \mathcal{G}} \hat{L}_{\mathrm{dual}}(g; f)$. In order to characterize the performance of this approach, consider the operator $\mathcal{T}_g$ for any $g \in \mathcal{G}$ as

$$(\mathcal{T}_g f)(s,a) = r(s,a) + \gamma \mathbb{E}_{s' \sim P^o_{s,a}}[(g(s,a) - \max_{a'} f(s',a'))_+] + (1-\gamma) g(s,a), \tag{9}$$

for all $f \in \mathcal{F}$ and $(s,a) \in \mathcal{S} \times \mathcal{A}$. We will show in Lemma 6 in Appendix C that the error $\sup_{f \in \mathcal{F}} \|\mathcal{T} f - \mathcal{T}_{\hat{g}_f} f\|_{1,\nu}$ is $O(\log(|\mathcal{F}|/\delta)/N)$ with probability at least $1-\delta$.

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 $Q_{k+1} = \mathcal{T} Q_k$ 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 $Q_{k+1} = \arg\min_{f \in \mathcal{F}} \|\mathcal{T} Q_k - f\|^2_{2,\nu}$. It is still infeasible to perform this update as it requires to exactly compute the expectation (w.r.t. P^o and ν) 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 $\mathcal{T}$ and the presence of the dual optimization problem within $\mathcal{T}$.

Given a dataset $\mathcal{D}$, we also follow the standard non-robust offline RL choice of least-squares residual minimization (Chen and Jiang, 2019; Xie et al., 2021; Wang et al., 2021). Define the empirical loss of f given f' (which represents the Q-function from the last iteration) and dual variable function g as

$$\hat{L}_{\mathrm{RFQI}}(f; f'; g) = \frac{1}{N} \sum_{i=1}^{N} \Big[r(s_i,a_i) + \gamma \big[(g(s_i,a_i) - \max_{a'} f'(s_i',a'))_+ + (1-\gamma) g(s_i,a_i) - f(s_i,a_i) \big] \Big]^2. \tag{10}$$

The correct dual variable function to be used in (10) is the optimal dual variable $g_{f'} = \arg\min_{g \in \mathcal{G}} L_{\mathrm{dual}}(g; f')$ corresponding to the last iterate f', which we will approximate it by $\hat{g}_{f'} = \arg\min_{g \in \mathcal{G}} \hat{L}_{\mathrm{dual}}(g; f')$. The RFQI update is then obtained as $\arg\min_{f \in \mathcal{F}} \hat{L}_{\mathrm{RFQI}}(f; f'; \hat{g}_{f'})$. Summarizing the individual steps described above, we formally give our RFQI algorithm below.

Algorithm 1 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:** $Q_0 = 0 \in \mathcal{F}$.
3: **for** $k = 0, \ldots, K-1$ **do**
4: **Dual variable function optimization:** Compute the dual variable function corresponding to Q_k through empirical risk minimization as $g_k = \hat{g}_{Q_k} = \arg\min_{g \in \mathcal{G}} \hat{L}_{\mathrm{dual}}(g; Q_k)$ (see (8)).
5: **Robust Q-update:** Compute the next iterate Q_{k+1} through least-squares regression as $Q_{k+1} = \arg\min_{Q \in \mathcal{F}} \hat{L}_{\mathrm{RFQI}}(Q; Q_k; g_k)$ (see (10)).
6: **end for**
7: **Output:** $\pi_K = \arg\max_a Q_K(s,a)$

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

Theorem 1. *Let Assumptions 1-4 hold. Let π_K be the output of the RFQI algorithm after K iterations. Denote $J^* = \mathbb{E}_{s \sim d_0}[V^*(s)]$ where d_0 is initial state distribution. Then, for any $\delta \in (0,1)$, with probability at least $1-\delta$, we have*

$$J^* - J^{\pi_K} \leq \frac{\gamma^K}{(1-\gamma)^2} + \frac{\sqrt{C}\left(\sqrt{6\varepsilon_c} + \varepsilon_{dual}\right)}{(1-\gamma)^2} + \frac{16}{(1-\gamma)^3} \sqrt{\frac{18C \log(2|\mathcal{F}||\mathcal{G}|/\delta)}{N}}.$$

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

Remark 2. The known sample complexity of robust-RL in the tabular setting is $\Theta\left(\frac{|\mathcal{S}|^2|\mathcal{A}|}{(\varepsilon(1-\gamma)^2(1-\gamma)^4}\right)$ (Yang et al., 2021; Panaganti and Kalathil, 2022). Considering $\log(|\mathcal{F}||\mathcal{G}|)$ to be $\Theta(|\mathcal{S}||\mathcal{A}|)$, we can recover the same bound as in the tabular setting (we lose a $|\mathcal{S}|$ due to the use of Bernstein inequality).

Remark 3. Under similar Bellman completeness and concentratability assumptions, RFQI sample complexity is comparable to that of a non-robust of ine RL algorithm, $O\left(\frac{1}{\varepsilon^2(1-\gamma)^4}\log\frac{|\mathcal{F}|}{\delta}\right)$ (Chen and Jiang, 2019). As a consequence of robustness, we have an $|\mathcal{A}|^2$ and $\log(|\mathcal{G}|)$ factors in our bound.

4.4 Proof Sketch

Here we brie y explain the key ideas used in the analysis of RFQI for obtaining the optimality gap bound in Theorem 1. The complete proof is provided in Appendix C.

Step 1: To bound $J^* - J^{\pi_K}$, we connect it to the error $\|Q^* - Q_K\|_{1,\nu}$ 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 $J^* - J^{\pi_K} \le 2\|Q^* - Q_K\|_{1,\nu}/(1-\gamma)$ (see (22) and the steps before in Appendix C).

Step 2: To bound $\|Q^* - Q_K\|_{1,\nu}$ for any state-action distribution ν such that $\nu = \nu_1 \sqrt{C}$, we decompose it to get a recursion, with approximation terms based on the least-squares regression and empirical risk minimization. Recall that g_f is the dual variable function from the algorithm for state-action value function $f \in \mathcal{F}$. Denote $\hat{f}_g$ 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., $\hat{f}_g = \arg\min_{Q\in\mathcal{F}} \hat{L}_{\text{RFQI}}(Q; f; g)$. By recursive use of the obtained inequality (23) (see Appendix C) and using uniform bound, we get

$$\|Q^* - Q_K\|_{1,\nu} \le \frac{\gamma^K}{1-\gamma} + \frac{\sqrt{C}}{1-\gamma}\sup_{f\in\mathcal{F}}\|\mathcal{T}f - \mathcal{T}_{g_f} f\|_{1,\nu} + \frac{\sqrt{C}}{1-\gamma}\sup_{f\in\mathcal{F}}\sup_{g\in\mathcal{G}}\|\mathcal{T}_g f - \hat{f}_g\|_{2,\nu}.$$

Step 3: We recognize that $\sup_{f\in\mathcal{F}}\|\mathcal{T}f - \mathcal{T}_{g_f} f\|_{1,\nu}$ is an empirical risk minimization error term. Using Rademacher complexity based bounds, we show in Lemma 6 that this error is $O(\log(|\mathcal{F}|/\delta)/N)$ with high probability.

Step 4: Similarly, we also recognize that $\sup_{f\in\mathcal{F}}\sup_{g\in\mathcal{G}}\|\mathcal{T}_g f - \hat{f}_g\|_{2,\nu}$ is a least-squares regression error term. We also show that this error is $O(\log(|\mathcal{F}||\mathcal{G}|/\delta)/N)$ with high probability. We adapt the generalized least squares regression result to accommodate the modi ed target functions resulting from the robust Bellman operator to obtain this bound (see Lemma 7).

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

5 Experiments

Figure 1: CartPole Figure 2: CartPole Figure 3: Hopper

Here, we demonstrate the robust performance of our RFQI algorithm by evaluating it on the CartPole and Hopper environments in OpenAI Gym (Brockman et al., 2016). In all the gures 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. A more detailed description of the experiments, and results on additional experiments, are deferred to Appendix F. We provide our code in github webpage https://github.com/zaiyan-x/RFQI containing instructions to reproduce all results in this paper.

For the Cartpole, we compare RFQI algorithm against the non-robust RL algorithms FQI and DQN, and the soft-robust RL algorithm proposed in Derman et al. (2018). We test the robustness of the algorithms by changing the parameter force_mag (to model external force disturbance), and also by introducing action perturbations (to model actuator noise). Fig. 1 and Fig. 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 of ine 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 of ine data according to a single nominal training model.

For the Hopper, we compare RFQI algorithm against the non-robust RL algorithms FQI and TD3 (Fujimoto et al., 2018), and the soft-robust RL (here soft-robust DDPG) algorithm proposed in Derman et al. (2018). We test the robustness of the algorithms by changing the parameter joint_stiffness. Fig. 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.

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 of ine 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 de ned with respect to the total variation distance. In future work, we will consider uncertainty sets de ned 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.

7 Acknowledgements

This work was supported in part by the National Science Foundation (NSF) grants NSF-CAREER-EPCN-2045783 and NSF ECCS 2038963. Any opinions, ndings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily re ect the views of the sponsoring agencies.

References

Agarwal, A., Jiang, N., Kakade, S. M., and Sun, W. (2019). Reinforcement learning: Theory and algorithms. CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep. 4, 5, 19, 21

Agarwal, A., Kakade, S., and Yang, L. F. (2020). Model-based reinforcement learning with a generative model is minimax optimal. In Conference on Learning Theory, pages 67–83. 23

Antos, A., Szepesvári, C., and Munos, R. (2008). Learning near-optimal policies with Bellman-residual minimization based tted policy iteration and a single sample path. Machine Learning 71(1):89–129. 2, 23

Azar, M. G., Munos, R., and Kappen, H. J. (2013). Minimax PAC bounds on the sample complexity of reinforcement learning with a generative model. Mach. Learn., 91(3):325–349. 23

Bertsekas, D. P. (2011). Approximate policy iteration: A survey and some new methods. Journal of Control Theory and Applications, 9(3):310–335. 2, 23

Borkar, V. S. (2002). Q-learning for risk-sensitive control. Mathematics of operations research 27(2):294–311. 23

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. (2016). Openai gym. arXiv preprint arXiv:1606.01540. 8, 24, 27

Chen, J. and Jiang, N. (2019). Information-theoretic considerations in batch reinforcement learning. In International Conference on Machine Learning, pages 1042–1051. 2, 4, 5, 7, 8, 23

Csiszár, I. (1967). Information-type measures of difference of probability distributions and indirect observation. studia scientiarum Mathematicarum Hungarica, 2:229–318. 16

Derman, E., Mankowitz, D., Mann, T., and Mannor, S. (2020). A bayesian approach to robust reinforcement learning. In Uncertainty in Artificial Intelligence, pages 648–658. 2

Derman, E., Mankowitz, D. J., Mann, T. A., and Mannor, S. (2018). Soft-robust actor-critic policy-gradient. In AUAI press for Association for Uncertainty in Artificial Intelligence, pages 208–218. 2, 9, 23, 26

Duchi, J. and Namkoong, H. (2018). Learning models with uniform performance via distributionally robust optimization. arXiv preprint arXiv:1810.08750. 5, 6, 16

Dullerud, G. E. and Paganini, F. (2013). A course in robust control theory: a convex approach, volume 36. Springer Science & Business Media. 23

Ernst, D., Geurts, P., and Wehenkel, L. (2005). Tree-based batch mode reinforcement learning. Journal of Machine Learning Research, 6:503–556. 2, 23

Farahmand, A.-m., Szepesvári, C., and Munos, R. (2010). Error propagation for approximate policy and value iteration. Advances in Neural Information Processing Systems, 23. 2, 23

Fei, Y., Yang, Z., Chen, Y., and Wang, Z. (2021). Exponential bellman equation and improved regret bounds for risk-sensitive reinforcement learning. Annual Conference on Neural Information Processing Systems 2021, pages 20436–20446. 23

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. (2020). D4rl: Datasets for deep data-driven reinforcement learning. 26, 27, 28

Fujimoto, S., Hoof, H., and Meger, D. (2018). Addressing function approximation error in actor-critic methods. In International Conference on Machine Learning, pages 1582–1591. 9, 27

Fujimoto, S., Meger, D., and Precup, D. (2019). Off-policy deep reinforcement learning without exploration. In International Conference on Machine Learning pages 2052–2062. 2, 23, 24, 25, 27

Gordon, G. J. (1995). Stable function approximation in dynamic programming. In Machine learning proceedings 1995, pages 261–268. 2, 23

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. International conference on machine learning, pages 1861–1870. 25, 26

Haskell, W. B., Jain, R., and Kalathil, D. (2016). Empirical dynamic programming. Mathematics of Operations Research, 41(2):402–429. 23

Huang, P., Xu, M., Fang, F., and Zhao, D. (2022). Robust reinforcement learning as a stackelberg game via adaptively-regularized adversarial training. arXiv preprint arXiv:2202.09514. 23

Iyengar, G. N. (2005). Robust dynamic programming. Mathematics of Operations Research, 30(2):257–280. 1, 2, 3, 4, 21, 23

Kalathil, D., Borkar, V. S., and Jain, R. (2021). Empirical Q-Value Iteration. Stochastic Systems, 11(1):1–18. 23

Kaufman, D. L. and Schaefer, A. J. (2013). Robust modified policy iteration. INFORMS Journal on Computing, 25(3):396–410. 23

Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980. 24

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114. 24

Kumar, A., Fu, J., Soh, M., Tucker, G., and Levine, S. (2019). Stabilizing off-policy q-learning via bootstrapping error reduction. Advances in Neural Information Processing Systems, pages 11784–11794. 2, 23

Kumar, A., Zhou, A., Tucker, G., and Levine, S. (2020). Conservative q-learning for of ine reinforcement learning. Advances in Neural Information Processing Systems, 33:1179–1191. 2, 23

Lange, S., Gabel, T., and Riedmiller, M. (2012). Batch reinforcement learning. Reinforcement learning, pages 45–73. Springer. 2, 23

Lazaric, A., Ghavamzadeh, M., and Munos, R. (2012). Finite-sample analysis of least-squares policy iteration. Journal of Machine Learning Research, 13:3041–3074. 2, 23

Levine, S., Kumar, A., Tucker, G., and Fu, J. (2020). Of ine reinforcement learning: Tutorial, review, and perspectives on open problems. arXiv preprint arXiv:2005.01643. 2, 23, 26, 27, 28

Li, G., Wei, Y., Chi, Y., Gu, Y., and Chen, Y. (2020). Breaking the sample size barrier in model-based reinforcement learning with a generative model. Advances in Neural Information Processing Systems, volume 33, pages 12861–12872. 23

Lim, S. H. and Autef, A. (2019). Kernel-based reinforcement learning in robust Markov decision processes. In International Conference on Machine Learning, pages 3973–3981. 23

Lim, S. H., Xu, H., and Mannor, S. (2013). Reinforcement learning in robust Markov decision processes. In Advances in Neural Information Processing Systems, pages 701–709. 2

Liu, Y., Swaminathan, A., Agarwal, A., and Brunskill, E. (2020). Provably good batch off-policy reinforcement learning without great exploration. Neural Information Processing Systems 2, 4, 23, 24, 25, 26, 27, 28

Mankowitz, D. J., Levine, N., Jeong, R., Abdolmaleki, A., Springenberg, J. T., Shi, Y., Kay, J., Hester, T., Mann, T., and Riedmiller, M. (2020). Robust reinforcement learning for continuous control with model misspeci cation. In International Conference on Learning Representations. 2, 23

Mannor, S., Mebel, O., and Xu, H. (2016). Robust mdps with k-rectangular uncertainty. Mathematics of Operations Research, 41(4):1484–1509. 2

Moses, A. K. and Sundaresan, R. (2011). Further results on geometric properties of a family of relative entropies. In 2011 IEEE International Symposium on Information Theory Proceedings, pages 1940–1944. 16

Munos, R. (2003). Error bounds for approximate policy iteration. ICML, volume 3, pages 560–567. 4

Munos, R. and Szepesvári, C. (2008). Finite-time bounds for tted value iteration. Journal of Machine Learning Research, 9(27):815–857. 2, 23

Nilim, A. and El Ghaoui, L. (2005). Robust control of Markov decision processes with uncertain transition matrices. Operations Research, 53(5):780–798. 1, 2, 4, 23

Panaganti, K. and Kalathil, D. (2021). Robust reinforcement learning using least squares policy iteration with provable performance guarantees. Proceedings of the 38th International Conference on Machine Learning, pages 511–520. 2, 23

Panaganti, K. and Kalathil, D. (2022). Sample complexity of robust reinforcement learning with a generative model. In Proceedings of The 25th International Conference on Arti cial Intelligence and Statistics, pages 9582–9602. 2, 8, 24

Peng, X. B., Andrychowicz, M., Zaremba, W., and Abbeel, P. (2018). Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 3803–3810. IEEE. 1

Pinto, L., Davidson, J., Sukthankar, R., and Gupta, A. (2017). Robust adversarial reinforcement learning. In *International Conference on Machine Learning*, pages 2817–2826. 2, 23

Prashanth, L. A. and Ghavamzadeh, M. (2016). Variance-constrained actor-critic algorithms for discounted and average reward mdps. *Mach. Learn.*, 105(3):367–417. 23

Raf n, A. (2020). Rl baselines3 zoo. https://github.com/DLR-RM/rl-baselines3-zoo . 25

Rockafellar, R. T. and Wets, R. J.-B. (2009). *Variational analysis*, volume 317. Springer Science & Business Media. 6, 15, 16

Roy, A., Xu, H., and Pokutta, S. (2017). Reinforcement learning under model mismatch. In *Advances in Neural Information Processing Systems*, pages 3043–3052. 2, 23

Russel, R. H. and Petrik, M. (2019). Beyond con dence regions: Tight bayesian ambiguity sets for robust mdps. *Advances in Neural Information Processing Systems*. 2

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*. 25

Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms*. Cambridge university press. 15

Shapiro, A. (2017). Distributionally robust stochastic programming. *SIAM Journal on Optimization*, 27(4):2258–2275. 5, 6, 16

Sidford, A., Wang, M., Wu, X., Yang, L. F., and Ye, Y. (2018). 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*, pages 5192–5202. 23

Singh, S. P. and Yee, R. C. (1994). An upper bound on the loss from approximate optimal-value functions. *Machine Learning*, 16(3):227–233. 23

Sünderhauf, N., Brock, O., Scheirer, W., Hadsell, R., Fox, D., Leitner, J., Upcroft, B., Abbeel, P., Burgard, W., Milford, M., et al. (2018). The limits and potentials of deep learning for robotics. *The International Journal of robotics research*, 37(4-5):405–420. 1

Szepesvári, C. and Munos, R. (2005). Finite time bounds for sampling based tted value iteration. In *Proceedings of the 22nd international conference on Machine learning*, pages 880–887. 4

Tamar, A., Mannor, S., and Xu, H. (2014). Scaling up robust mdps using function approximation. In *International Conference on Machine Learning*, pages 181–189. 2, 23

Tessler, C., Efroni, Y., and Mannor, S. (2019). Action robust reinforcement learning and applications in continuous control. In *International Conference on Machine Learning*, pages 6215–6224. 23

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. (2017). 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)*, pages 23–30. 1

Vershynin, R. (2018). *High-Dimensional Probability: An Introduction with Applications in Data Science*, volume 47. Cambridge University press. 15

Wang, R., Foster, D., and Kakade, S. M. (2021). What are the statistical limits of of ine {rl} with linear function approximation? In *International Conference on Learning Representations*. 4, 5, 7

Wang, Y. and Zou, S. (2021). Online robust reinforcement learning with model uncertainty. *Advances in Neural Information Processing Systems*, 34. 2

Wiesemann, W., Kuhn, D., and Rustem, B. (2013). Robust Markov decision processes. *Mathematics of Operations Research*, 38(1):153–183. 2, 23

Xie, T., Cheng, C.-A., Jiang, N., Mineiro, P., and Agarwal, A. (2021). Bellman-consistent pessimism for of ine reinforcement learning. *Advances in neural information processing systems*, 34. 2, 4, 5, 7, 23

Xie, T. and Jiang, N. (2020). Q* approximation schemes for batch reinforcement learning: A theoretical comparison. In *Conference on Uncertainty in Arti cial Intelligence*, pages 550–559. 2, 23

Xu, H. and Mannor, S. (2010). Distributionally robust Markov decision processes. In *Advances in Neural Information Processing Systems*, pages 2505–2513. 2, 23

Yang, W., Zhang, L., and Zhang, Z. (2021). Towards theoretical understandings of robust markov decision processes: Sample complexity and asymptotics. *arXiv preprint arXiv:2105.03863*. 2, 8, 24

Yu, P. and Xu, H. (2015). Distributionally robust counterpart in Markov decision processes. *IEEE Transactions on Automatic Control*, 61(9):2538–2543. 2

Yu, T., Thomas, G., Yu, L., Ermon, S., Zou, J., Levine, S., Finn, C., and Ma, T. (2020). Mopo: Model-based of ine policy optimization. In *Advances in Neural Information Processing Systems*. 2, 23

Zhang, H., Chen, H., Xiao, C., Li, B., Liu, M., Boning, D., and Hsieh, C.-J. (2020a). Robust deep reinforcement learning against adversarial perturbations on state observations. *Advances in Neural Information Processing Systems*, 33:21024–21037. 2

Zhang, K., Hu, B., and Basar, T. (2020b). Policy optimization for $\mathcal{H}_2$ linear control with $\mathcal{H}_\infty$ robustness guarantee: Implicit regularization and global convergence. In *Proceedings of the 2nd Annual Conference on Learning for Dynamics and Control*, volume 120, pages 179–190. 23

Zhang, S. and Jiang, N. (2021). Towards hyperparameter-free policy selection for of ine reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 12864–12875. 2, 23

Zhang, X., Chen, Y., Zhu, X., and Sun, W. (2022). Corruption-robust of ine reinforcement learning. In *International Conference on Arti cial Intelligence and Statistics*, pages 5757–5773. PMLR. 23

Zhang, Y., Yang, Z., and Wang, Z. (2021). Provably ef cient actor-critic for risk-sensitive and robust adversarial rl: A linear-quadratic case. In *International Conference on Arti cial Intelligence and Statistics*, pages 2764–2772. 23

Zhou, K., Doyle, J. C., Glover, K., et al. (1996). *Robust and optimal control*, volume 40. Prentice hall New Jersey. 23

Zhou, Z., Bai, Q., Zhou, Z., Qiu, L., Blanchet, J., and Glynn, P. (2021). Finite-sample regret bound for distributionally robust of ine tabular reinforcement learning. In *International Conference on Arti cial Intelligence and Statistics*, pages 3331–3339. 2, 24

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately re ect the paper's contributions and scope? [Yes] See contributions in the Introduction.
   (b) Did you describe the limitations of your work? [Yes] The discussions on the assumptions describes the limitations.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Sections 3-4.3
   (b) Did you include complete proofs of all theoretical results? [Yes] We provide proof sketch 4.4 in main paper and the complete proof in Appendix with self-contained material.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Described in the Appendix.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Described in the main paper and the Appendix.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Mentioned in the Appendix.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identi able information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Appendix

A 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 2 (Bernstein's inequality (Vershynin, 2018, Theorem 2.8.4)) *Let $X_1, \ldots, X_T$ be independent random variables. Assume that $|X_t - \mathbb{E}[X_t]| \le M$, for all t. Then, for any $\varepsilon > 0$, we have*

$$\mathbb{P}\left(\frac{1}{T}\sum_{t=1}^{T}(X_t - \mathbb{E}[X_t]) \ge \varepsilon\right) \le 2\exp\left(\frac{-T^2\varepsilon^2}{2\sigma^2 + \frac{2MT\varepsilon}{3}}\right),$$

where $\sigma^2 = \sum_{t=1}^{T}\mathbb{E}[X_t^2]$. Furthermore, if $X_1, \ldots, X_T$ are independent and identically distributed random variables, then for any $\delta \in (0,1)$, we have

$$\mathbb{E}[X_1] \le \frac{1}{T}\sum_{t=1}^{T} X_t + \sqrt{\frac{2\mathbb{E}[X_1^2]\log(2/\delta)}{T}} + \frac{M\log(2/\delta)}{3T},$$

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 Shalev-Shwartz and Ben-David (2014, Theorem 26.5, Lemma 26.8, Lemma 26.9).

Lemma 3 (ERM generalization bound) *Let $\mathbb{P}$ 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 that $|l(h;x)| \le c_1$ for some positive constant $c_1 > 0$. Given a dataset $\mathcal{D} = \{X_i\}_{i=1}^N$, generated independently from $\mathbb{P}$, denote $\hat{h}$ as the ERM solution, i.e. $\hat{h} = \arg\min_{h\in\mathcal{H}} (1/N)\sum_{i=1}^N l(h;X_i)$. For any fixed $\delta \in (0,1)$ and $h^* \in \arg\min_{h\in\mathcal{H}} \mathbb{E}_{X\sim\mathbb{P}}[l(h;X)]$, we have*

$$\mathbb{E}_{X\sim\mathbb{P}}[l(\hat{h};X)] \le \mathbb{E}_{X\sim\mathbb{P}}[l(h^*;X)] \le 2\mathcal{R}(l\circ\mathcal{H}\circ\mathcal{D}) + 5c_1\sqrt{\frac{2\log(8/\delta)}{N}}, \tag{11}$$

with probability at least $1-\delta$, where $\mathcal{R}(\cdot)$ is the Rademacher complexity of $\mathcal{H}$ given by

$$\mathcal{R}(l\circ\mathcal{H}\circ\mathcal{D}) = \frac{1}{N}\mathbb{E}_{\{\sigma_i\}_{i=1}^N}\left[\sup_{g\in l\circ\mathcal{H}}\sum_{i=1}^{N}\sigma_i g(X_i)\right];$$

in which σ_i's are independent from X_i's and are independently and identically distributed according to the Rademacher random variable, 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 $|h(x)| \le c_2$ for all $h \in \mathcal{H}$ and $x \in \mathcal{X}$, and $l(h;x)$ is c_3-Lipschitz in h, then we have

$$\mathbb{E}_{X\sim\mathbb{P}}[l(\hat{h};X)] \le \mathbb{E}_{X\sim\mathbb{P}}[l(h^*;X)] \le 2c_2c_3\sqrt{\frac{2\log(|\mathcal{H}|)}{N}} + 5c_1\sqrt{\frac{2\log(8/\delta)}{N}}, \tag{12}$$

with probability at least $1-\delta$.

We now mention two important concepts from variational analysis (Rockafellar and Wets, 2009) literature that is useful to relate minimization of integrals and the integrals of pointwise minimization under special class of functions.

Definition 1 (Decomposable spaces and Normal integrands (Rockafellar and Wets, 2009, Definition 14.59, Example 14.29)) *A space $\mathcal{X}$ of measurable functions is a decomposable space relative to an underlying measure space $(\Omega, \mathcal{A}, \nu)$, if for every function $x_0 \in \mathcal{X}$, every set $A \in \mathcal{A}$ with $\nu(A) < \infty$, and any bounded measurable function $x_1 : A \to \mathbb{R}$, the function $x(\omega) = x_0(\omega)1(\omega \notin A) + x_1(\omega)1(\omega \in A)$ 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 4. A few examples of decomposable spaces are $L^p(S, \mathcal{A}, \mu; \mathbb{R})$ for any $p \geq 1$ and $M(S, \mathcal{A}, \mu; \mathbb{R})$, the space of all $(S, \mathcal{A})$-measurable functions.

Lemma 4 (Rockafellar and Wets, 2009, Theorem 14.60). Let X be a space of measurable functions from Ω to $\mathbb{R}$ that is decomposable relative to a 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

$$\inf_{x \in X} \int_{\omega \in \Omega} f(\omega, x(\omega)) \, \mu(d\omega) = \int_{\omega \in \Omega} \inf_{x \in \mathbb{R}} f(\omega, x) \, \mu(d\omega).$$

Moreover, as long as the above in mum is not $-\infty$, we have that

$$x^0 \in \arg\min_{x \in X} \int_{\omega \in \Omega} f(\omega, x(\omega)) \, \mu(d\omega),$$

if and only if $x^0(\omega) \in \arg\min_{x \in \mathbb{R}} f(\omega, x)$ almost surely.

We now give one result from distributionally robust optimization. The ϕ-divergence between the distributions P and P^o is defined as

$$D_\phi(P \| P^o) = \int \phi\left(\frac{dP}{dP^o}\right) dP^o, \tag{13}$$

where ϕ is a convex function (Csiszár, 1967; Moses and Sundaresan, 2011). We obtain different divergences for different forms of the function ϕ, including some well-known divergences. For example, $\phi(t) = |t - 1|/2$ gives Total Variation (TV), $\phi(t) = t \log t$ gives Kullback-Liebler (KL), $\phi(t) = (t-1)^2$ gives Chi-square, and $\phi(t) = (\sqrt{t} - 1)^2$ 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., Shapiro (2017, Section 3.2) and Duchi and Namkoong (2018, Proposition 1).

Proposition 2. Let D_ϕ be the ϕ-divergence as defined in (13). Then,

$$\sup_{D_\phi(P \| P^o)} E_P[l(X)] = \inf_{\lambda > 0, \, \gamma \in \mathbb{R}} E_{P^o}\left[\phi^*\left(\frac{l(X)}{\lambda} - \gamma\right) + \gamma\right] \lambda + \rho\lambda, \tag{14}$$

where $\phi^*(s) = \sup_{t \geq 0}\{st - \phi(t)\}$ is the Fenchel conjugate.

Note that on the right hand side of (14), 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.

B Proof of the Proposition 1

As the first step, we adapt the result given in Proposition 2 in two ways: (i) Since Proposition 1 considers the TV uncertainty set, we will derive the specific form of this result for the TV uncertainty set, (ii) Since Proposition 1 considers the minimization problem instead of the maximization problem, unlike in Proposition 2, we will derive the specific form of this result for minimization.

Lemma 5. Let D_ϕ be as defined in (13) with $\phi(t) = |t-1|/2$ corresponding to the TV uncertainty set. Then,

$$\inf_{D_\phi(P \| P^o)} E_P[l(X)] = \sup_{\gamma \in \mathbb{R}} E_{P^o}[(\gamma - l(X))_+] + (\gamma - \inf_{x \in X} l(x))_+ \rho;$$

Proof. First, we will compute the Fenchel conjugate of $\phi(t) = |t-1|/2$. We have

$$\phi^*(s) = \sup_{t \geq 0}\{st - \tfrac{1}{2}|t-1|\} = \max\left\{\sup_{t \in [0,1]} (s + \tfrac{1}{2})t - \tfrac{1}{2}, \sup_{t > 1} (s - \tfrac{1}{2})t + \tfrac{1}{2}\right\}.$$

It is easy to see that for $s > 1/2$, we have $\phi^*(s) = +\infty$, and for $s < -1/2$, we have $\phi^*(s) = -1/2$. For $s \in [-1/2, 1/2]$, we have

$$\phi^*(s) = \max\left\{\sup_{t \in [0,1]} (s + \tfrac{1}{2})t - \tfrac{1}{2}, \sup_{t > 1} (s - \tfrac{1}{2})t + \tfrac{1}{2}\right.$$

$$= \max \quad ((s + \frac{1}{2}) \quad 1 \quad \frac{1}{2}); ((s \quad \frac{1}{2}) \quad 1 + \frac{1}{2}) \quad = s:$$

Thus, we have

$$f \ (s) = \begin{cases} \frac{1}{2} & s \quad \frac{1}{2}; \\ s & s 2 [\ \frac{1}{2}; \frac{1}{2}] : \\ +1 & s > \frac{1}{2}: \end{cases}$$

From Proposition 2, we obtain

$$\sup_{D_f (P kP^o)} E_P [l(X)] = \inf_{> 0; \ 2R} E_{P^o}[f \ (\frac{l(X)}{\quad})] + \quad +$$

$$= \inf_{; \ : > 0; \ 2R; \frac{\sup_{x 2X} \ l(x)}{\quad} \ \frac{1}{2}} E_{P^o}[\ \max f \frac{l(X)}{\quad}; \ \frac{1}{2} g] + \quad +$$

$$= \inf_{; \ : > 0; \ 2R; \frac{\sup_{x 2X} \ l(x)}{\quad} \ \frac{1}{2}} E_{P^o}[\max f l(X) \quad ; \quad = 2g] + \quad +$$

$$= \inf_{; \ : > 0; \ 2R; \frac{\sup_{x 2X} \ l(x)}{\quad} \ \frac{1}{2}} E_{P^o}[(l(X) \quad + \ = 2)_+] \quad = 2 + \quad +$$

$$= \inf_{; \ ^0: > 0; \ ^0 2R; \frac{\sup_{x 2X} \ l(x) \quad ^0}{\quad} \ 1} E_{P^o}[(l(X) \quad ^0)_+] + \quad + \ ^0:$$

The second equality follows since $c(\frac{l(X)}{\quad}) = + \ 1$ whenever $\frac{l(X)}{\quad} > \frac{1}{2}$, which can be ignored as we are minimizing over and . The fourth equality follows form the fact that $\max f x; y g = (x \quad y)_+ + y$ for any $x; y \ 2 \ R$. Finally, the last equality follows by making the substitution $^0 = \quad = 2$. Taking the optimal value of , i.e., $= (\sup_{x 2X} \ l(x) \quad ^0)_+$, we get

$$\sup_{D_f (P kP^o)} E_P [l(X)] = \inf_{2R} E_{P^o}[(l(X) \quad)_+] + (\sup_{x 2X} l(x) \quad)_+ \quad + :$$

Now,

$$\inf_{D_f (P kP^o)} E_P [l(X)] = \sup_{D_f (P kP^o)} E_P [\ l(X)]$$

$$= \inf_{2R} E_{P^o}[(\ l(X) \quad)_+] + (\sup_{x 2X} \quad l(x) \quad)_+ \quad +$$

$$= \inf_{^0 2R} E_{P^o}[(\ ^0 \quad l(X))_+] + (\ ^0 \quad \inf_{x 2X} l(x))_+ \quad ^0;$$

which completes the proof. □

We are now ready to prove Proposition 1.

Proof of Proposition 1 For each $(s; a)$, the optimization problem in (3) is given by $\min_{P_{s;a} \ 2P_{s;a}} E_{s^0 \ P_{s;a}} [V (s^0)]$, and our focus is on the setting where $P_{s;a}$ is given by the TV uncertainty set. So $P_{s;a}$ can be equivalently de ned using the f-divergence with $f (t) = jt \quad 1j=2$ as $P_{s;a} = f P_{s;a} : D_f (P_{s;a} jj P^o_{s;a}) \quad g$. We can now use the result of Lemma 5 to get

$$\inf_{P_{s;a} \ 2P_{s;a}} E_{s^0 \ P_{s;a}} [V (s^0)] = \inf_{2R} E_{s^0 \ P^o_{s;a}} [(\quad V (s^0))_+] + (\quad \inf_{s^{00} 2S} V (s^{00}))_+ \quad :$$

From Proposition 2, the function $h(\ ) = E_{s^0 \ P^o_{s;a}} [(\quad V (s^0))_+] + (\quad \inf_{s^{00}} V (s^{00}))_+$ is convex in . Since $V (s^0) \quad 0$, $h(\ ) = \quad 0$ when $\quad 0$. So, $\inf_{\ 2(1 \ ;0]} h(\ )$, achieved at $= 0$. Also, since $V (s) \quad 1=(1 \quad)$, we have

$$h(\frac{2}{(1 \quad)}) = E_{s^0 \ P^o_{s;a}} [\frac{2}{(1 \quad)} \quad V (s^0)] + (\frac{2}{(1 \quad)} \quad \inf_{s^{00}} V (s^{00})) \quad \frac{2}{(1 \quad)}$$

$$\frac{1}{(1 \quad)} + (\frac{2}{(1 \quad)} \quad \frac{1}{(1 \quad)}) = \frac{2}{(1 \quad)} \quad \frac{(1 + \ )}{(1 \quad)} \quad 0:$$

So, it is suf cient to consider $2 [0; \frac{2}{(1 \quad)}]$ for the above optimization problem.

Using these, we get

$$(TQ)(s;a) = r(s;a) + \inf_{P_{s;a} \in \mathcal{P}_{s;a}} \mathbb{E}_{s' \sim P_{s;a}}[V(s')]$$

$$= r(s;a) + \frac{1}{2} \inf_{\in [0; \frac{2}{(1-)}]} \mathbb{E}_{s' \sim P^o_{s;a}}[(\quad V(s'))_+] + (\quad \inf_{s'' \in S} V(s''))_+ \quad :$$

This completes the proof of Proposition 1. □

C Proof of Theorem 1

We start by proving Lemma 1 which mainly follows from Lemma 4 in Appendix A.

Proof of Lemma 1. Let $h((s;a);) = \mathbb{E}_{s' \sim P^o_{s;a}}((\quad \max_{a'} f(s';a'))_+ \quad (1 \quad))$. We note that $h((s;a);)$ is $(S \quad A)$-measurable in $(s;a) \in S \quad A$ for each $\in [0; 1=(1 \quad))]$ and is continuous in for each $(s;a) \in S \quad A$. Now it follows that $h((s;a);)$ is a normal integrand (see Definition 1 in Appendix A). We now note that $L^1(S \quad A; (S \quad A);)$ is a decomposable space (Remark 4 in Appendix A). Thus, this lemma now directly follows from Lemma 4. □

Now we state a result and provide its proof for the empirical risk minimization on the dual parameter.

Lemma 6 (Dual Optimization Error Bound) Let $\hat{g}_f$ 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 (9). With probability at least $1 \quad$, we have

$$\sup_{f \in F} \| T_{\hat{g}_f} f \quad T_{g_f} f \|_{1;} \quad \frac{4 \ (2 \quad)}{(1 \quad)} \sqrt{\frac{2 \log(|G|)}{N}} + \frac{25}{(1 \quad)} \sqrt{\frac{2 \log(8|F| =)}{N}} + "_{\text{dual}}:$$

Proof. Fix any $f \in F$. We will also invoke union bound for the supremum here. We recall from (8) that $\hat{g}_f = \arg\min_{g \in G} \hat{L}_{\text{dual}}(g; f)$. From the robust Bellman equation, we directly obtain

$$\| T_{\hat{g}_f} f \quad Tf \|_{1;} = (\mathbb{E}_{s;a} | \mathbb{E}_{s' \sim P^o_{s;a}}((\hat{g}_f(s;a) \quad \max_{a'} f(s';a'))_+ \quad (1 \quad)\hat{g}_f(s;a)) \quad \inf_{\in [0;2=(1 \quad))]} \mathbb{E}_{s' \sim P^o_{s;a}}((\quad \max_{a'} f(s';a'))_+ \quad (1 \quad))|)$$

$$\overset{(a)}{=} (\mathbb{E}_{s;a} \quad \mathbb{E}_{s' \sim P^o_{s;a}}((\hat{g}_f(s;a) \quad \max_{a'} f(s';a'))_+ \quad (1 \quad)\hat{g}_f(s;a)) \quad \mathbb{E}_{s;a}[\inf_{\in [0;2=(1 \quad))]} \mathbb{E}_{s' \sim P^o_{s;a}}((\quad \max_{a'} f(s';a'))_+ \quad (1 \quad))])$$

$$\overset{(b)}{=} (\mathbb{E}_{s;a;s' \sim P^o_{s;a}}((\hat{g}_f(s;a) \quad \max_{a'} f(s';a'))_+ \quad (1 \quad)\hat{g}_f(s;a)) \quad \inf_{g \in L^1} \mathbb{E}_{s;a;s' \sim P^o_{s;a}}((g(s;a) \quad \max_{a'} f(s';a'))_+ \quad (1 \quad)g(s;a)))$$

$$= (\mathbb{E}_{s;a;s' \sim P^o_{s;a}}((\hat{g}_f(s;a) \quad \max_{a'} f(s';a'))_+ \quad (1 \quad)\hat{g}_f(s;a)) \quad \inf_{g \in G} \mathbb{E}_{s;a;s' \sim P^o_{s;a}}((g(s;a) \quad \max_{a'} f(s';a'))_+ \quad (1 \quad)g(s;a))) + (\inf_{g \in G} \mathbb{E}_{s;a;s' \sim P^o_{s;a}}((g(s;a) \quad \max_{a'} f(s';a'))_+ \quad (1 \quad)g(s;a)) \quad \inf_{g \in L^1} \mathbb{E}_{s;a;s' \sim P^o_{s;a}}((g(s;a) \quad \max_{a'} f(s';a'))_+ \quad (1 \quad)g(s;a)))$$

$$\overset{(c)}{\leq} (\mathbb{E}_{s;a;s' \sim P^o_{s;a}}((\hat{g}_f(s;a) \quad \max_{a'} f(s';a'))_+ \quad (1 \quad)\hat{g}_f(s;a)) \quad \inf_{g \in G} \mathbb{E}_{s;a;s' \sim P^o_{s;a}}((g(s;a) \quad \max_{a'} f(s';a'))_+ \quad (1 \quad)g(s;a))) + "_{\text{dual}}$$

$$\overset{(d)}{\leq} 2 R(\mathcal{I} \quad G \quad D) + \frac{25}{(1 \quad)} \sqrt{\frac{2 \log(8=)}{N}} + "_{\text{dual}}$$

$$\overset{(e)}{\leq} \frac{4 \ (2 \quad)}{(1 \quad)} \sqrt{\frac{2 \log(|G|)}{N}} + \frac{25}{(1 \quad)} \sqrt{\frac{2 \log(8=)}{N}} + "_{\text{dual}}:$$

(a) follows since $\inf_g h(g) \le h(g_f)$. (b) follows from Lemma 1. (c) follows from the approximate dual realizability assumption (Assumption 4).

For (d), we consider the loss function $l(g;(s,a,s')) = (g(s,a) - \max_{a'} f(s',a'))_+ - (1-\gamma)g(s,a)$ and dataset $\mathcal{D} = \{s_i, a_i, s_i'\}_{i=1}^N$. Note that $|l(g;(s,a,s'))| \le 2/(1-\gamma)$ (since $f \in \mathcal{F}$ and $g \in \mathcal{G}$). Now, we can apply the empirical risk minimization result (11) in Lemma 3 to get (d), where $\mathcal{R}(\cdot)$ is the Rademacher complexity.

Finally, (e) follows from (12) in Lemma 3 when combined with the facts that $l(g;(s,a,s'))$ is $(2-\gamma)$-Lipschitz in g and $|g(s,a)| \le 2/(1-\gamma)$, since $g \in \mathcal{G}$.

With union bound, with probability at least $1-\delta$, we finally get

$$\sup_{f \in \mathcal{F}} \|T_{g_f} f - f\|_{1,\nu} \le \frac{4(2-\gamma)}{(1-\gamma)^2}\sqrt{\frac{2\log(|\mathcal{G}|/\delta)}{N}} + \frac{25}{(1-\gamma)^2}\sqrt{\frac{2\log(8|\mathcal{F}|/\delta)}{N}} + \varepsilon_{\text{dual}},$$

which concludes the proof. □

We next prove the least-squares generalization bound for the RFQI algorithm.

Lemma 7 (Least squares generalization bound) Let f_{g_f} 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 (9). Then, with probability at least $1-\delta$, we have

$$\sup_{f \in \mathcal{F}} \sup_{g \in \mathcal{G}} \|T_g f - f_{g_f}\|_{2,\nu} \le \sqrt{6\varepsilon_c} + \frac{16}{(1-\gamma)^2}\sqrt{\frac{18\log(2|\mathcal{F}||\mathcal{G}|/\delta)}{N}}.$$

Proof. We adapt the least-squares generalization bound given in Agarwal et al. (2019, Lemma A.11) to our setting. We recall from (10) that $f_{g_f} = \arg\min_{Q \in \mathcal{F}} L_{\text{RFQI}}(Q; f, g)$. 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

$$z_i^{f'} = (f'(s_i,a_i) - y_i)^2 - ((T_g f)(s_i,a_i) - y_i)^2,$$

where $y_i = r_i - (g(s_i,a_i) - \max_{a'} f(s_i',a'))_+ + (1-\gamma)g(s_i,a_i)$, and $(s_i,a_i,s_i') \in \mathcal{D}$ with $(s_i,a_i) \sim \nu$, $s_i' \sim P^o_{s_i,a_i}$. 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 $g(s_i,a_i) \le 2/(1-\gamma)$ (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:

$$\begin{aligned}
\mathbb{E}_{s_i' \sim P^o_{s_i,a_i}}[z_i^{f'}] &= \mathbb{E}_{s_i' \sim P^o_{s_i,a_i}}[(f'(s_i,a_i) - (T_g f)(s_i,a_i))\,(f'(s_i,a_i) + (T_g f)(s_i,a_i) - 2y_i)] \\
&= (f'(s_i,a_i) - (T_g f)(s_i,a_i))^2, \\
\mathbb{E}_{s_i' \sim P^o_{s_i,a_i}}[(z_i^{f'})^2] &= \mathbb{E}_{s_i' \sim P^o_{s_i,a_i}}[(f'(s_i,a_i) - (T_g f)(s_i,a_i))^2\,(f'(s_i,a_i) + (T_g f)(s_i,a_i) - 2y_i)^2] \\
&= (f'(s_i,a_i) - (T_g f)(s_i,a_i))^2\,\mathbb{E}_{s_i' \sim P^o_{s_i,a_i}}[(f'(s_i,a_i) + (T_g f)(s_i,a_i) - 2y_i)^2] \\
&\le C_1 (f'(s_i,a_i) - (T_g f)(s_i,a_i))^2,
\end{aligned}$$

where $C_1 = 16^2/(2-\gamma)^2(1-\gamma)^2)$. This immediately implies that

$$\mathbb{E}_{s_i,a_i \sim \nu,\, s_i' \sim P^o_{s_i,a_i}}[z_i^{f'}] = \|T_g f - f'\|_{2,\nu}^2,$$

$$\mathbb{E}_{s_i,a_i \sim \nu,\, s_i' \sim P^o_{s_i,a_i}}[(z_i^{f'})^2] \le C_1 \|T_g f - f'\|_{2,\nu}^2.$$

From these calculations, it is also straightforward to see that $|z_i^{f'}| \le 2C_1$ almost surely.

Now, using the Bernstein's inequality (Lemma 2), together with a union bound over $f' \in \mathcal{F}$, with probability at least $1-\delta$, we have

$$\|T_g f - f'\|_{2,\nu}^2 - \frac{1}{N}\sum_{i=1}^N z_i^{f'} \le \sqrt{\frac{2C_1\|T_g f - f'\|_{2,\nu}^2 \log(2|\mathcal{F}|/\delta)}{N}} + \frac{2C_1\log(2|\mathcal{F}|/\delta)}{3N}, \quad (15)$$

for all $f' \in \mathcal{F}$. Setting $f' = \hat{f}_g$, with probability at least $1 - \delta/2$, we have

$$\|f - \hat{f}_g\|_{2,\rho}^2 \leq \frac{1}{N}\sum_{i=1}^{N} z_i^{\hat{f}_g} + \sqrt{\frac{2C_1 k T_g \|f - \hat{f}_g\|_{2,\rho}^2 \log(4|\mathcal{F}|/\delta)}{N}} + \frac{2C_1 \log(4|\mathcal{F}|/\delta)}{3N}. \quad (16)$$

Now we upper-bound $(1/N)\sum_{i=1}^{N} z_i^{\hat{f}_g}$ in the following. Consider a function $f^e \in \arg\min_{h \in \mathcal{F}} \|h - T_g f\|_{2,\rho}^2$. Note that f^e is independent of the dataset. We note that our earlier first and second moment calculations hold true for f^e, replacing f', as well. Now, from (15) setting $f' = f^e$, with probability at least $1 - \delta/2$ we have

$$\frac{1}{N}\sum_{i=1}^{N} z_i^{f^e} \leq k\,\|T_g f - f^e\|_{2,\rho}^2 + \sqrt{\frac{2C_1 k T_g \|f - f^e\|_{2,\rho}^2 \log(4|\mathcal{F}|/\delta)}{N}} + \frac{2C_1 \log(4|\mathcal{F}|/\delta)}{3N}. \quad (17)$$

Suppose $(1/N)\sum_{i=1}^{N} z_i^{f^e} \leq 2C_1 \log(4|\mathcal{F}|/\delta)/N$ holds, then from (17) we get

$$\frac{1}{N}\sum_{i=1}^{N} z_i^{f^e} \leq k\,\|T_g f - f^e\|_{2,\rho}^2 + \sqrt{k T_g \|f - f^e\|_{2,\rho}^2 \cdot \frac{1}{N}\sum_{i=1}^{N} z_i^{f^e}} + \frac{2C_1 \log(4|\mathcal{F}|/\delta)}{N}. \quad (18)$$

We note the following algebra fact: Suppose $x^2 \leq ax + b$ with $b > 0$ and $a^2 \leq 4b$, then we have $x \leq a$. Taking $x = (1/N)\sum_{i=1}^{N} z_i^{f^e}$ in this fact, from (18) we get

$$\frac{1}{N}\sum_{i=1}^{N} z_i^{f^e} \leq 3k\,\|T_g f - f^e\|_{2,\rho}^2 + \frac{4C_1 \log(4|\mathcal{F}|/\delta)}{3N} \leq 3k\,\|T_g f - f^e\|_{2,\rho}^2 + \frac{2C_1 \log(4|\mathcal{F}|/\delta)}{N}. \quad (19)$$

Now suppose $(1/N)\sum_{i=1}^{N} z_i^{f^e} \leq 2C_1 \log(4|\mathcal{F}|/\delta)/N$, then (19) holds immediately. Thus, (19) always holds with probability at least $1 - \delta/2$. Furthermore, recall $f^e \in \arg\min_{h \in \mathcal{F}} \|h - T_g f\|_{2,\rho}^2$, we have

$$\begin{aligned}\frac{1}{N}\sum_{i=1}^{N} z_i^{f^e} &\leq 3k\,\|T_g f - f^e\|_{2,\rho}^2 + \frac{2C_1 \log(4|\mathcal{F}|/\delta)}{N} \\ &= 3\min_{h \in \mathcal{F}} \|h - T_g f\|_{2,\rho}^2 + \frac{2C_1 \log(4|\mathcal{F}|/\delta)}{N} \leq 3\varepsilon_c + \frac{2C_1 \log(4|\mathcal{F}|/\delta)}{N}; \quad (20)\end{aligned}$$

where the last inequality follows from the approximate robust Bellman completion assumption (Assumption 2).

We note that since $\hat{f}_g$ is the least-squares regression solution, we know $(1/N)\sum_{i=1}^{N} z_i^{\hat{f}_g} \leq (1/N)\sum_{i=1}^{N} z_i^{f^e}$. With this note in (20), from (16), with probability at least $1 - \delta$, we have

$$\begin{aligned}\|f - \hat{f}_g\|_{2,\rho}^2 \leq\ & 3\varepsilon_c + \frac{2C_1 \log(4|\mathcal{F}|/\delta)}{N} \\ & + \sqrt{\frac{2C_1 k T_g \|f - \hat{f}_g\|_{2,\rho}^2 \log(4|\mathcal{F}|/\delta)}{N}} + \frac{2C_1 \log(4|\mathcal{F}|/\delta)}{3N} \\ & \sqrt{3\varepsilon_c + \frac{3C_1 \log(4|\mathcal{F}|/\delta)}{N} + \sqrt{\frac{3C_1 k T_g \|f - \hat{f}_g\|_{2,\rho}^2 \log(4|\mathcal{F}|/\delta)}{N}}}.\end{aligned}$$

From the earlier algebra fact, taking $x = \|f - \hat{f}_g\|_{2,\rho}$, with probability at least $1 - \delta$, we have

$$\|f - \hat{f}_g\|_{2,\rho}^2 \leq 6\varepsilon_c + \frac{9C_1 \log(4|\mathcal{F}|/\delta)}{N}.$$

From the fact $\sqrt{x + y} \leq \sqrt{x} + \sqrt{y}$, with probability at least $1 - \delta$, we get

$$\|f - \hat{f}_g\|_{2,\rho} \leq \sqrt{6\varepsilon_c} + \sqrt{\frac{9C_1 \log(4|\mathcal{F}|/\delta)}{N}}.$$

Using union bound for $f \in F$ and $g \in G$, with probability at least $1 - \delta$, we finally obtain

$$\sup_{f \in F} \sup_{g \in G} \| T_g f - f_g \|_{2,\nu} \leq \sqrt{6\varepsilon_c} + \sqrt{\frac{18 C_1 \log(2|F||G|/\delta)}{N}},$$

which completes the least-squares generalization bound analysis. □

We are now ready to prove the main theorem.

Proof of Theorem 1 We let $V_k(s) = Q_k(s, \pi_k(s))$ for every $s \in S$. 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 2 that Q^π is a fixed-point of the robust Bellman operator T^π defined in (3). We also note that the same holds true for any stationary deterministic policy from Iyengar (2005) that Q^π satisfies $Q^\pi(s,a) = r(s,a) + \gamma \min_{P_{s,a} \in \mathcal{P}_{s,a}} E_{s' \sim P_{s,a}}[V^\pi(s')]$. We can now further use the dual form (5) under Assumption 3. We first characterize the performance decomposition between V^* and V^{π_K}. For a given $s_0 \in S$, we observe that

$$\begin{aligned}
V^*(s_0) - V^{\pi_K}(s_0) &= (V^*(s_0) - V_K(s_0)) - (V^{\pi_K}(s_0) - V_K(s_0)) \\
&= (Q^*(s_0, \pi^*(s_0)) - Q_K(s_0, \pi_K(s_0))) - (Q^{\pi_K}(s_0, \pi_K(s_0)) - Q_K(s_0, \pi_K(s_0))) \\
&\overset{(a)}{\leq} Q^*(s_0, \pi^*(s_0)) - Q_K(s_0, \pi^*(s_0)) + Q_K(s_0, \pi_K(s_0)) - Q^{\pi_K}(s_0, \pi_K(s_0)) \\
&= Q^*(s_0, \pi^*(s_0)) - Q_K(s_0, \pi^*(s_0)) + Q_K(s_0, \pi_K(s_0)) - Q^*(s_0, \pi_K(s_0)) \\
&\quad + Q^*(s_0, \pi_K(s_0)) - Q^{\pi_K}(s_0, \pi_K(s_0)) \\
&\overset{(b)}{\leq} Q^*(s_0, \pi^*(s_0)) - Q_K(s_0, \pi^*(s_0)) + Q_K(s_0, \pi_K(s_0)) - Q^*(s_0, \pi_K(s_0)) \\
&\quad + \gamma \sup_{P \in \mathcal{P}^o_{s_0, \pi_K(s_0)}} (E_{s_1 \sim P}[(V^*(s_1) - V^{\pi_K}(s_1))_+] - E_{s_1 \sim P}[(V^* - V^{\pi_K})(s_1)_+]) \\
&\overset{(c)}{\leq} |Q^*(s_0, \pi^*(s_0)) - Q_K(s_0, \pi^*(s_0))| + |Q^*(s_0, \pi_K(s_0)) - Q_K(s_0, \pi_K(s_0))| \\
&\quad + \gamma \sup_{P \in \mathcal{P}^o_{s_0, \pi_K(s_0)}} E_{s_1 \sim P}[|V^*(s_1) - V^{\pi_K}(s_1)|].
\end{aligned}$$

(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

$$\begin{aligned}
E_{s_0 \sim d_0}[V^*] - E_{s_0 \sim d_0}[V^{\pi_K}] &\leq E_{s_0 \sim d_0}|Q^*(s_0, \pi^*(s_0)) - Q_K(s_0, \pi^*(s_0))| + |Q^*(s_0, \pi_K(s_0)) - Q_K(s_0, \pi_K(s_0))| \\
&\quad + \gamma E_{s_1 \sim P^o_{s_0, \pi_K(s_0)}}[|V^*(s_1) - V^{\pi_K}(s_1)|].
\end{aligned}$$

Since $V^*(s) \geq V^{\pi_K}(s)$ for any $s \in S$, by telescoping we get

$$E_{s_0 \sim d_0}[V^*] - E_{s_0 \sim d_0}[V^{\pi_K}] \leq \sum_{h=0}^{\infty} \gamma^h E_{s \sim d_{h,\pi_K}}[|Q^*(s, \pi^*(s)) - Q_K(s, \pi^*(s))| + |Q^*(s, \pi_K(s)) - Q_K(s, \pi_K(s))|], \quad (21)$$

where $d_{h,\pi_K} \in \Delta(S)$ 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 distributions 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 Agarwal et al. (2019).

Recall $\|f\|_{p,\nu}^2 = (\mathbb{E}_{s,a\sim\nu}|f(s,a)|^p)^{1/p}$, where $\nu \in (\mathcal{S}\times\mathcal{A})$. With this we have

$$\mathbb{E}_{s_0\sim d_0}[V^{\pi^*}] - \mathbb{E}_{s_0\sim d_0}[V^{\pi_K}] \le \sum_{h=0}^{K-1} \|Q^{\pi^*} - Q_K\|_{1,d_{h,\pi_K}} + \|Q^{\pi^*} - Q_K\|_{1,d_{h,\pi_K,\pi^*}}, \tag{22}$$

where the state-action distributions $d_{h,\pi}(s,a) = d_{h,\pi}(s)\mathbb{1}\{a = \pi(s)\}$ and $d_{h,\pi,\pi^*}(s,a) = d_{h,\pi}(s)\mathbb{1}\{a = \pi^*(s)\}$ directly follows by comparing with (21).

We now bound one of the RHS terms above by bounding for any state-action distribution ν satisfying Assumption 1 (in particular the following bound is true for d_{h,π_K} or d_{h,π_K,π^*} in (21)):

$$\begin{aligned}
&\|Q^{\pi^*} - Q_K\|_{1,\nu} \le \|Q^{\pi^*} - \mathcal{T}Q_{K-1}\|_{1,\nu} + \|\mathcal{T}Q_{K-1} - Q_K\|_{1,\nu} \\
&\overset{(a)}{\le} \|Q^{\pi^*} - \mathcal{T}Q_{K-1}\|_{1,\nu} + \sqrt{C}\|\mathcal{T}Q_{K-1} - Q_K\|_{2,\nu} \\
&= (\mathbb{E}_{s,a\sim\nu}|Q^{\pi^*}(s,a) - \mathcal{T}Q_{K-1}(s,a)|) + \sqrt{C}\|\mathcal{T}Q_{K-1} - Q_K\|_{2,\nu} \\
&\overset{(b)}{\le} (\mathbb{E}_{s,a\sim\nu}\sup|\mathbb{E}_{s'\sim P^o_{s,a}}((\max_{a'} Q_{K-1}(s',a'))_+ - (\max_{a'} Q^{\pi^*}(s',a'))_+)|) \\
&\qquad + \sqrt{C}\|\mathcal{T}Q_{K-1} - Q_K\|_{2,\nu} \\
&\overset{(c)}{\le} (\mathbb{E}_{s,a\sim\nu}|\mathbb{E}_{s'\sim P^o_{s,a}}(\max_{a'} Q^{\pi^*}(s',a') - \max_{a'} Q_{K-1}(s',a'))_+|) + \sqrt{C}\|\mathcal{T}Q_{K-1} - Q_K\|_{2,\nu} \\
&\overset{(d)}{\le} (\mathbb{E}_{s,a\sim\nu}\mathbb{E}_{s'\sim P^o_{s,a}}\max_{a'}|Q^{\pi^*}(s',a') - Q_{K-1}(s',a')|) + \sqrt{C}\|\mathcal{T}Q_{K-1} - Q_K\|_{2,\nu} \\
&\overset{(e)}{=} \|Q^{\pi^*} - Q_{K-1}\|_{1,\nu'} + \sqrt{C}\|\mathcal{T}Q_{K-1} - Q_K\|_{2,\nu} \\
&\overset{(f)}{\le} \|Q^{\pi^*} - Q_{K-1}\|_{1,\nu'} + \sqrt{C}\|\mathcal{T}_{g_{K-1}}Q_{K-1} - Q_K\|_{2,\nu} + \sqrt{C}\|\mathcal{T}Q_{K-1} - \mathcal{T}_{g_{K-1}}Q_{K-1}\|_{2,\nu},
\end{aligned} \tag{23}$$

where (a) follows by the concentratability assumption (Assumption 1), (b) from Bellman equation, operator $\mathcal{T}$, and the fact $|\sup_x p(x) - \sup_x q(x)| \le \sup_x |p(x) - q(x)|$, (c) from the fact $|(x)_+ - (y)_+| \le |x - y|$ for any $x, y \in \mathbb{R}$, (d) follows by Jensen's inequality and by the fact $|\sup_x p(x) - \sup_x q(x)| \le \sup_x |p(x) - q(x)|$ and $(x)_+ \le |x|$ for any $x, y \in \mathbb{R}$, and (e) by defining the distribution ν' as $\nu'(s',a') = \sum_{s,a}\nu(s,a)P^o_{s,a}(s')\mathbb{1}\{a' = \arg\max_b |Q^{\pi^*}(s',b) - Q_{K-1}(s',b)|\}$, and (f) using the fact that $\|\cdot\|_{1,\nu} \le \|\cdot\|_{2,\nu}$.

Now, by recursion until iteration 0, we get

$$\begin{aligned}
&\|Q^{\pi^*} - Q_K\|_{1,\nu} \le \gamma^K \sup\|Q^{\pi^*} - Q_0\|_{1,\nu'} + \sqrt{C}\sum_{t=0}^{K-1}\gamma^t\|\mathcal{T}Q_{K-1-t} - \mathcal{T}_{g_{K-1-t}}Q_{K-1-t}\|_{2,\nu} \\
&\qquad + \sqrt{C}\sum_{t=0}^{K-1}\gamma^t\|\mathcal{T}_{g_{K-1-t}}Q_{K-1-t} - Q_{K-t}\|_{2,\nu} \\
&\overset{(a)}{\le} \frac{\gamma^K}{1-\gamma} + \sqrt{C}\sum_{t=0}^{K-1}\gamma^t\|\mathcal{T}Q_{K-1-t} - \mathcal{T}_{g_{K-1-t}}Q_{K-1-t}\|_{2,\nu} \\
&\qquad + \sqrt{C}\sum_{t=0}^{K-1}\gamma^t\|\mathcal{T}_{g_{K-1-t}}Q_{K-1-t} - Q_{K-t}\|_{2,\nu} \\
&\overset{(b)}{\le} \frac{\gamma^K}{1-\gamma} + \frac{\sqrt{C}}{1-\gamma}\sup_{f\in\mathcal{F}}\|\mathcal{T}f - \mathcal{T}_{g_f}f\|_{2,\nu} + \frac{\sqrt{C}}{1-\gamma}\sup_{f\in\mathcal{F}}\|\mathcal{T}_{g_f}f - f_{g_f}\|_{2,\nu} \\
&\le \frac{\gamma^K}{1-\gamma} + \frac{\sqrt{C}}{1-\gamma}\sup_{f\in\mathcal{F}}\|\mathcal{T}f - \mathcal{T}_{g_f}f\|_{1,\nu} + \frac{\sqrt{C}}{1-\gamma}\sup_{f\in\mathcal{F}}\sup_{g\in\mathcal{G}}\|\mathcal{T}_g f - f_g\|_{2,\nu},
\end{aligned} \tag{24}$$

where (a) follows since $|Q^{\pi^*}(s,a)| \le 1/(1-\gamma)$, $Q_0(s,a) = 0$, and (b) follows since g_f is the dual variable function from the algorithm for the state-action value function f and 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 (22) and (24) with Lemma 6 and Lemma 7. □

D Related Works

Here we provide a more detailed description of the related work to complement what we listed in the introduction (Section 1).

Of ine RL: The problem of learning the optimal policy only using an of ine dataset is rst addressed under the generative model assumption (Singh and Yee, 1994; Azar et al., 2013; Haskell et al., 2016; Sidford et al., 2018; Agarwal et al., 2020; Li et al., 2020; Kalathil et al., 2021). 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 (Antos et al., 2008; Bertsekas, 2011; Lange et al., 2012; Chen and Jiang, 2019; Xie and Jiang, 2020; Levine et al., 2020; Xie et al., 2021) that utilize function approximation under similar assumption, concentratability assumption (Chen and Jiang, 2019) in which the data distribution suf ciently covers the discounted state-action occupancy. There is rich literature (Munos and Szepesvári, 2008; Farahmand et al., 2010; Lazaric et al., 2012; Chen and Jiang, 2019; Liu et al., 2020; Xie et al., 2021) in the conquest of identifying and improving these necessary and suf cient assumptions for of ine RL that use variations of Fitted Q-Iteration (FQI) algorithm (Gordon, 1995; Ernst et al., 2005). There is also rich literature (Fujimoto et al., 2019; Kumar et al., 2019, 2020; Yu et al., 2020; Zhang and Jiang, 2021) that develop of ine deep RL algorithms focusing on the algorithmic and empirical aspects and propose multitude heuristic approaches to advance the eld. All these results assume that the of ine data is generated according to a single model and the goal is to nd the optimal policy for the MDP with the same model. In particular, none of these works consider the of ine robust RL problem where the of ine 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, Iyengar (2005) and Nilim and El Ghaoui (2005) introduced the RMDP framework. Iyengar (2005) showed that the optimal robust value function and policy can be computed using the robust counterparts of the standard value iteration and policy iteration algorithms. To tackle the parameter uncertainty problem, other works considered distributionally robust setting (Xu and Mannor, 2010), modi ed policy iteration (Kaufman and Schaefer, 2013), and more general uncertainty set (Wiesemann et al., 2013). These initial works mainly focused on the planning problem (known transition probability dynamics) in the tabular setting. Tamar et al. (2014) 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. Roy et al. (2017) proposed the robust versions of the classical model-free reinforcement learning algorithms, such as Q-learning, SARSA, and TD-learning in the tabular setting. They also proposed function approximation based algorithms for the policy evaluation. However, this work does not have a policy iteration algorithm with provable guarantees for learning the optimal robust policy. Derman et al. (2018) introduced soft-robust actor-critic algorithms using neural networks, but does not provide any global convergence guarantees for the learned policy. Tessler et al. (2019) proposed a min-max game framework to address the robust learning problem focusing on the tabular setting. Lim and Autef (2019) proposed a kernel-based RL algorithm for nding the robust value function in a batch learning setting. Mankowitz et al. (2020) employed an entropy-regularized policy optimization algorithm for continuous control using neural network, but does not provide any provable guarantees for the learned policy. Panaganti and Kalathil (2021) proposed least-squares policy iteration method to handle large state-action space in robust RL, but only provide asymptotic policy evaluation convergence guarantees whereas Panaganti and Kalathil (2021) provide nite 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 (Zhou et al., 1996; Dullerud and Paganini, 2013). Recently, there are some interesting works that address the robust RL problem using this framework, especially focusing on the linear quadratic regulator setting (Zhang et al., 2020b). Risk sensitive RL algorithms (Borkar, 2002; Prashanth and Ghavamzadeh, 2016; Fei et al., 2021) and adversarial RL algorithms (Pinto et al., 2017; Zhang et al., 2021; Huang et al., 2022) also address the robustness problem implicitly under different frameworks which are independent from RMDPs. (Zhang et al., 2022) addresses the problem of adversarially robust of ine RL problem, where an adversary is allowed to change a fraction of the samples of an of ine RL dataset and the goal is to nd the optimal policy for the nominal linear MDP model (according to

which the of ine data is generated). Our framework and approach of robust MDP is signi cantly different from these line of works.

This work: The works that are closest to ours are by Zhou et al. (2021); Yang et al. (2021); Panaganti and Kalathil (2022) that address the robust RL problem in a tabular setting under the generative model assumption. Due to the generative model assumption, the of ine 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 signi cantly 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 of ine 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 rst work that addresses the of ine robust RL problem with arbitrary large state space using function approximation, with provable guarantees on the performance of the learned policy.

E Experiment Details

We provide more detailed and practical version of our RFQI algorithm (Algorithm 1) in this section. We also provide more experimental results evaluated on Cartpole, Hopper, and Half-Cheetah OpenAI Gym Mujoco (Brockman et al., 2016) environments.

We provide our code in github webpage 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 (Fujimoto et al., 2019)[2] and Pessimistic Q-learning (PQL) algorithm (Liu et al., 2020)[3]. We note that PQL algorithm (with $b = 0$ ltration thresholding (Liu et al., 2020)) and BCQ algorithm are the practical versions of FQI algorithm with neural network architecture.

E.1 RFQI Practical Algorithm

We provide the practical version of our RFQI algorithm in Algorithm 2 and highlight the difference with BCQ and PQL algorithms in blue (steps 8 and 9).

RFQI algorithm implementation details: The Variational Auto-Encoder (VAE) G^a_{ω} (Kingma and Welling, 2013) is de ned 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 $D_{\omega_2}(s,z) \to a$. Then the evidence lower bound ELBO of VAE is given by $ELBO(\mathcal{B}; G^a_{\omega}) = \sum_{\mathcal{B}} (a - \hat{a})^2 + D_{KL}(N(\mu, \sigma); N(0,1))$, where N is the normal distribution with mean and standard deviation parameters. We refer to (Fujimoto et al., 2019) for more details on VAE. We also use the default VAE architecture from BCQ algorithm (Fujimoto et al., 2019) and PQL algorithm (Liu et al., 2020) in our RFQI algorithm.

We now focus on the additions described in blue (steps 8 and 9) in Algorithm 2. For all the other networks we use default architecture from BCQ algorithm (Fujimoto et al., 2019) and PQL algorithm (Liu et al., 2020) in our RFQI algorithm.
(1) In each iteration k, we solve the dual variable function optimization problem (step 4 in Algorithm 1, step 8 in Algorithm 2) implemented by ADAM (Kingma and Ba, 2014) on the minibatch $\mathcal{B}$ with the learning rate ℓ_1 mentioned in Table 1.
(2) Our state-action value target function corresponds to the robust state-action value target function described in (10). This is re ected in step 9 of Algorithm 2. The state-action value function Q optimization problem (step 5 in Algorithm 1, step 9 in Algorithm 2) is implemented by ADAM (Kingma and Ba, 2014) on the minibatch $\mathcal{B}$ with the learning rate ℓ_2 mentioned in Table 1.

[2] Available at https://github.com/sfujim/BCQ
[3] Available at https://github.com/yaoliucs/PQL

Algorithm 2 RFQI Practical Algorithm

Input: Of ine dataset $\mathcal{D}$, radius of robustness , maximum perturbation , target update rate , mini-batch size N, maximum number of iterations K, number of actions n.
Initialize: Two state-action neural networks Q_{θ_1} and Q_{θ_2}, one dual neural network g_{θ_3} policy (perturbation) model: 2 [;]) ; and action VAE G^a_ω, with random parameters $_1$, $_2$, ' , ! , and target networks $Q_{\theta'_1}; Q_{\theta'_2}$, ' with $\theta'_1 \quad \theta_1$; $\theta'_2 \quad \theta_2$, ' ' .
for $k = 1$; ; K do
  Sample a minibatch B with N samples from $\mathcal{D}$.
  Train ! arg min $_!$ ELBO $(B; G^a_\omega)$: Sample u actions a^0_i from $G^a_\omega(s^0)$ for each s^0.
  Perturb u actions $a^0_i = a^0_i +$ $(s^0; a^0_i)$.
  Compute next-state value target for each s^0 in B:

$$V_t = \max_{a^0_i}(0{:}75 \ \min\{Q_{\theta'_1}; Q_{\theta'_2}\} + 0{:}25 \ \max\{Q_{\theta'_1}; Q_{\theta'_2}\}):$$

  $_3$ arg min $\sum$ [max$\{g_\theta(s; a) \quad V_t(s^0); 0\}$ (1 $)g_\theta(s; a)$]:
  Compute next-state Q target for each $(s; a; r; s^0)$ pair in B:

$$Q_t(s; a) = r \quad \max\{g_{\theta_3}(s; a) \quad V_t(s^0); 0\} + \quad (1 \quad)g_{\theta_3}(s; a):$$

  arg min $\sum$ $(Q_t(s; a) \quad Q_\theta(s; a))^2$:
  Sample u actions a_i from $G^a_\omega(s)$ for each s.
  ' arg max$_{a_i}$ max$_{a_i}$ $Q_{\theta_1}(s; a_i +$ $(s; a_i))$.
  Update target network: $^0 = (1 \quad)\ ^0 + \ ; \ ' \ ^0 = (1 \quad)' \ ^0 + \ ' \ .$
end for
Output policy: Given s, sample u actions a_i from $G^a_\omega(s)$. Select action $a =$ arg max$_{a_i}$ $Q_{\theta_1}(s; a_i +$ $(s; a_i))$.

Environment	Discount	Learning rates $[l_1; l_2]$	Q Neural nets $_1 = \ _2 = [h_1; h_2]$	Dual Neural nets $_3 = [h_1; h_2]$
CartPole	0:99	[10 ^{3}; 10 ^{3}]	[400; 300]	[64; 64]
Hopper	0:99	[10 ^{3}; 8 10 ^{4}] [3 10 ^{4}; 6 10 ^{4}]	[400; 300]	[64; 64]
Half-Cheetah	0:99	[10 ^{3}; 8 10 ^{4}] [3 10 ^{4}; 6 10 ^{4}]	[400; 300]	[64; 64]

Table 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 1. We use same hyper-parameters across different algorithms. Across all learning algorithms we use $= 0{:}005$ for the target network update, $K = 5 \quad 10^5$ for the maximum iterations, $|\mathcal{D}| = 10^6$ for the of ine dataset, $|B| = 1000$ for the minibatch size. We used grid-search for in $\{0{:}2; 0{:}3; \ ; 0{:}6\}$. We also picked best of the two sets of learning rates mentioned in Table 1. For all the other hyper-parameters we use default values from BCQ algorithm (Fujimoto et al., 2019) and PQL algorithm (Liu et al., 2020) in our RFQI algorithm that can be found in our code.

Of ine datasets: Now we discuss the of ine 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 of ine datasets.

Cartpole dataset $\mathcal{D}_c$: We rst train proximal policy optimization (PPO) (Schulman et al., 2017) algorithm, under default RL baseline zoo (Raf n, 2020) parameters. We then generate the Cartpole dataset $\mathcal{D}_c$ with 10^5 samples using an -greedy (= 0:3) version of this PPO trained policy. We note that this of ine dataset contains non-expert behavior meeting the richness of the data-generating distribution assumption in practice.

Mixed dataset $\mathcal{D}_m$: For the MuJoCo environments, Hopper and Half-Cheetah, we increase the richness of the dataset since these are high dimensional problems. We rst train soft actor-critic (SAC) (Haarnoja et al., 2018) algorithm, under default RL baseline zoo (Raf n, 2020) 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 $\mathcal{D}_m$ 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 $\mathcal{D}_c$ generation for fair comparison between FQI and RFQI algorithms.

D4RL dataset $\mathcal{D}_d$: We consider the hopper-medium and halfcheetah-medium offline datasets in (Fu et al., 2020) which are benchmark datasets in offline RL literature (Fu et al., 2020; Levine et al., 2020; Liu et al., 2020). These `medium' datasets are generated by first training a policy online using Soft Actor-Critic (Haarnoja et al., 2018), early-stopping the training, and collecting 10^6 samples from this partially-trained policy. We refer to (Fu et al., 2020) 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.

E.2 More Experimental Results

Figure 4: Cartpole simulation results on offline dataset $\mathcal{D}_c$. Average cumulative reward of 20 episodes versus different model parameter perturbations mentioned in the respective titles.

Figure 5: Hopper simulation results on offline dataset $\mathcal{D}_m$. Average cumulative reward of 20 episodes versus different model parameter perturbations mentioned in the respective titles.

Here we provide more experimental results and details in addition to Fig. 1-3 in Section 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 Derman et al. (2018). We trained FQI and RFQI algorithms on the dataset $\mathcal{D}_c$ (a detailed description of data set is provided in Appendix E.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 1.0 and 0.5 respectively. Fig. 4 shows superior robust performance of RFQI compared to the non-robust FQI and DQN. For example, consider the action perturbation performance plot in Fig. 4 where RFQI algorithm improves by 75% compared to FQI algorithm in average cumulative reward for 40% chance of action perturbation. We note that we found $\kappa = 0.5$ 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 (Brockman et al., 2016) environments Hopper and Half-Cheetah, we first mention their model parameters and its corresponding nominal values in Table 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 (Brockman et al., 2016) 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 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 (Fujimoto et al., 2018). We trained FQI and RFQI algorithms on the mixed dataset $\mathcal{D}_m$ (a detailed description of dataset provided in Appendix E.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. 3 and Fig. 5 shows RFQI algorithm is consistently robust compared to the non-robust algorithms. We note that we found $= 0.5$ 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. 5, 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 (Fujimoto et al., 2019) is a powerful off-policy policy gradient algorithm that relies on large 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 of line dataset, we repeat the above experiments, on additional OpenAI Gym MuJoCo (Brockman et al., 2016) environment Half-Cheetah, using D4RL dataset $\mathcal{D}_d$ (a detailed description of dataset provided in Appendix E.1) which are benchmark in of line RL literature (Fu et al., 2020; Levine et al., 2020; Liu et al., 2020) than our mixed dataset $\mathcal{D}_m$. Since D4RL dataset is a benchmark dataset for of line RL algorithms, here we focus only on the comparison between the two of line 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 parameter gravity, joint_damping and joint_frictionloss. Fig. 6 shows RFQI algorithm is consistently robust compared to the non-robust FQI algorithm. We note that we found $= 0.5$ 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. 6, for the

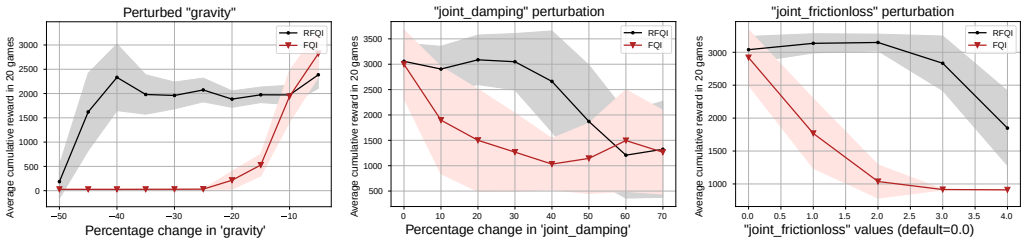

Figure 6: *Hopper evaluation simulation results on offline dataset D_d.* Average cumulative reward in 20 episodes versus different model parameter perturbations mentioned in the respective titles.

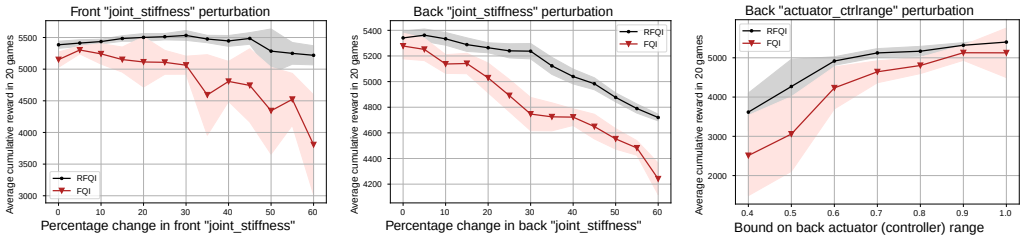

Figure 7: *Half-Cheetah evaluation simulation results on offline dataset D_d.* Average cumulative reward in 20 episodes versus different model parameter perturbations mentioned in the respective titles.

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* of front and back joints, and *actuator_ctrlrange* of back joint. Fig. 7 shows RFQI algorithm is consistently robust compared to the non-robust FQI algorithm. We note that we found = 0:3 is the best from grid-search for RFQI algorithm. For example, in plot 1 in Fig. 7, 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.

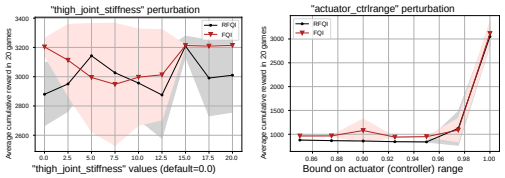

Figure 8: Similar performance of RFQI and FQI in Hopper on dataset D_{d} w.r.t. parameters *actuator_ctrlrange* and *thigh_joint_stiffness*.

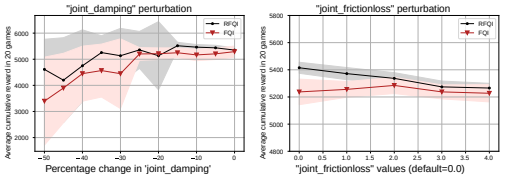

Figure 9: Similar performance of RFQI and FQI in Half-Cheetah on dataset D_{d} w.r.t. parameters *joint_damping* and *joint_frictionloss*.

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 D_{d} perform similarly under the perturbations of the *Hopper* model parameters *actuator_ctrlrange* and *thigh_joint_stiffness* as shown in Fig. 8. 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. 9. We observed that the robustness performance can depend on the offline data available, which was also observed for non-robust offline RL algorithms (Liu et al., 2020; Fu et al., 2020; Levine et al., 2020). Also, perturbing some parameters may make the problem 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.