Single-Trajectory Distributionally Robust Reinforcement Learning

Zhipeng Liang [*1] **Xiaoteng Ma** [*2] **Jose Blanchet** [3] **Jun Yang** [2] **Jiheng Zhang** [1 4] **Zhengyuan Zhou** [5 6]

Abstract

To mitigate the limitation that the classical reinforcement learning (RL) framework heavily relies on identical training and test environments, Distributionally Robust RL (DRRL) has been proposed to enhance performance across a range of environments, possibly including unknown test environments. As a price for robustness gain, DRRL involves optimizing over a set of distributions, which is inherently more challenging than optimizing over a fixed distribution in the non-robust case. Existing DRRL algorithms are either model-based or fail to learn from a single sample trajectory. In this paper, we design a first fully model-free DRRL algorithm, called *distributionally robust Q-learning with single trajectory (DRQ)*. We delicately design a multi-timescale framework to fully utilize each incrementally arriving sample and directly learn the optimal distributionally robust policy without modeling the environment, thus the algorithm can be trained along a single trajectory in a model-free fashion. Despite the algorithm's complexity, we provide asymptotic convergence guarantees by generalizing classical stochastic approximation tools. Comprehensive experimental results demonstrate the superior robustness and sample complexity of our proposed algorithm, compared to non-robust methods and other robust RL algorithms.

[*]Equal contribution [1]Department of Industrial Engineering and Decision Analytics, Hong Kong University of Science and Technology [2]Department of Automation, Tsinghua University [3]Department of Management Science and Engineering, Stanford University [4]Department of Mathematics, Hong Kong University of Science and Technology [5]Stern School of Business, New York University [6]Arena Technologies. Correspondence to: Zhengyuan Zhou <zhengyuanzhou24@gmail.com>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

1. Introduction

Reinforcement Learning (RL) is a machine learning paradigm for studying sequential decision problems. Despite considerable progress in recent years (Silver et al., 2016; Mnih et al., 2015; Vinyals et al., 2019), RL algorithms often encounter a discrepancy between training and test environments. This discrepancy is widespread since test environments may be too complex to be perfectly represented in training, or the test environments may inherently shift from the training ones, especially in certain application scenarios, such as financial markets and robotic control. Overlooking the mismatch could impede the application of RL algorithms in real-world settings, given the known sensitivity of the optimal policy of the Markov Decision Process (MDP) to the model (Mannor et al., 2004; Iyengar, 2005).

To address this concern, Distributionally Robust RL (DRRL) (Zhou et al., 2021; Yang et al., 2022; Shi & Chi; Panaganti & Kalathil, 2022; Panaganti et al., 2022; Ma et al., 2022; Yang, 2018; Abdullah et al., 2019; Neufeld & Sester, 2022) formulates the decision problem under the assumption that the test environment varies but remains close to the training environment. The objective is to design algorithms optimizing the worst-case expected return over an ambiguity set encompassing all possible test distributions. Evaluating a DRRL policy necessitates deeper insight into the transition dynamics than evaluating a non-robust one, as it entails searching for the worst-case performance across all distributions within the ambiguity set. Therefore, most prior solutions are model-based, require the maintenance of an estimator for the entire transition model and the ambiguity set. Such requirements may render these algorithms less practical in scenarios with large state-action spaces or where adequate modeling of the real environment is unfeasible.

Prompted by this issue, we study a fully model-free DRRL algorithm in this paper, which learns the optimal DR policy without explicit environmental modeling. The algorithm's distinctive feature is its capacity to learn from a single sample trajectory, representing the least demanding requirement for data collection. This feature results from our innovative algorithmic framework, comprising incrementally updated estimators and a delicate approximation scheme. While most model-free non-robust RL algorithms support training

in this setting—contributing to their widespread use—no existing work can effectively address the DRRL problem in this way. The challenge arises from the fact that approximating a DR policy by learning from a single trajectory suffers from restricted control over state-action pairs and limited samples, i.e., only one sample at a time. As we will demonstrate, a simple plug-in estimator using one sample, which is unbiased in the non-robust Q-learning algorithm, fails to approximate any robust value accurately.

The complexity of this task is further affirmed by the sole attempt to develop a model-free DRRL algorithm in (Liu et al., 2022). It relies on a restricted simulator assumption, enabling the algorithm to access an arbitrary number of samples from any state-action pair, thereby amassing sufficient system dynamics information before addressing the DRRL problem. Relaxing the dependence on a simulator and developing a fully model-free algorithm capable of learning from a single trajectory necessitates a delicate one-sample estimator for the DR value, carefully integrated into an algorithmic framework to eradicate bias from insufficient samples and ensure convergence to the optimal policy. Moreover, current solutions heavily depend on the specific divergence chosen to construct the ambiguity set and fail to bridge different divergences, underscoring the practical importance of divergence selection.

Thus a nature question arises: *Is it possible to develop a model-free DRRL framework that can learn the optimal DR policy across different divergences using only a single sample trajectory for learning?*

1.1. Our Contributions

In this paper, we provide a positive solution to the aforementioned question by making the following contributions:

1. We introduce a pioneering approach to construct the ambiguity set using the Cressie-Read family of f-divergence. By leveraging the strong duality form of the corresponding distributionally robust reinforcement learning (DRRL) problem, we reformulate it, allowing for the learning of the optimal DR policies using misspecified MDP samples. This formulation effortlessly covers widely used divergences such as the Kullback-Leibler (KL) and χ^2 divergence.

2. To address the additional nonlinearity that arises from the DR Bellman equation, which is absent in its non-robust counterpart, we develop a novel multi-timescale stochastic approximation scheme. This scheme carefully exploits the structure of the DR Bellman operator. The update of the Q table occurs in the slowest loop, while the other two loops are delicately designed to mitigate the bias introduced by the plug-in estimator due to the nonlinearity.

3. We instantiate our framework into a DR variant of the Q-learning algorithm, called *distributionally robust Q-learning with single trajectory (DRQ)*. This algorithm solves discount Markov Decision Processes (MDPs) in a fully online and incremental manner. We prove the asymptotic convergence of our proposed algorithm by extending the classical two-timescale stochastic approximation framework, which may be of independent interest.

4. We conduct extensive experiments to showcase the robustness and sample efficiency of the policy learned by our proposed DR Q-learning algorithm. We also create a deep learning version of our algorithm and compare its performance to representative online and offline (robust) reinforcement learning benchmarks on classical control tasks.

1.2. Related Work

Robust MDPs and RL: The framework of robust MDPs has been studied in several works such as Nilim & El Ghaoui (2005); Iyengar (2005); Wiesemann et al. (2013); Lim et al. (2013); Ho et al. (2021); Goyal & Grand-Clement (2022). These works discuss the computational issues using dynamic programming with different choices of MDP formulation, as well as the choice of ambiguity set, when the transition model is known. Robust Reinforcement Learning (RL) (Roy et al., 2017; Badrinath & Kalathil, 2021; Wang & Zou, 2021) relaxes the requirement of accessing to the transition model by simultaneously approximating to the ambiguity set as well as the optimal robust policy, using only the samples from the misspecified MDP.

Online Robust RL: Existing online robust RL algorithms including Wang & Zou (2021); Badrinath & Kalathil (2021); Roy et al. (2017), highly relies on the choice of the R-contamination model and could suffer over-conservatism. This ambiguity set maintains linearity in their corresponding Bellman operator and thus inherits most of the desirable benefits from its non-robust counterpart. Instead, common distributionally robust ambiguity sets, such as KL or χ^2 divergence ball, suffer from extra nonlinearity when trying to learn along a single-trajectory data, which serves as the foundamental challenge in this paper.

Distributionally Robust RL: To tackle the over-conservatism aroused by probability-agnostic R-contamination ambiguity set in the aforementioned robust RL, DRRL is proposed by constructing the ambiguity set with probability-aware distance (Zhou et al., 2021; Yang et al., 2022; Shi & Chi; Panaganti & Kalathil, 2022; Panaganti et al., 2022; Ma et al., 2022), including KL and χ^2 divergence. As far as we know, most of the existing DRRL algorithms fall into the model-based fashion, which first estimate the whole transition model and then construct

the ambiguity set around the model. The DR value and the corresponding policy are then computed based upon them. Their main focus is to understand the sample complexity of the DRRL problem in the offline RL regime, leaving the more prevalent single-trajectory setting largely unexplored.

2. Preliminary

2.1. Discounted MDPs

Consider an infinite-horizon MDP $(\mathcal{S}, \mathcal{A}, \gamma, \mu, P, r)$ where $\mathcal{S}$ and $\mathcal{A}$ are finite state and action spaces with cardinality S and A. $P : \mathcal{S} \times \mathcal{A} \to \Delta_S$ is the state transition probability measure. Here Δ_S is the set of probability measures over $\mathcal{S}$. r is the reward function and γ is the discount factor. We assume that $r : \mathcal{S} \times \mathcal{A} \to [0, 1]$ is deterministic and bounded in $[0, 1]$. A stationary policy $\pi : \mathcal{S} \to \Delta_A$ maps, for each state s to a probability distribution over the action set $\mathcal{A}$ and induce a random trajectory $s_1, a_1, r_1, s_2, \cdots$, with $s_1 \sim \mu$, $a_n = \pi(s_n)$ and $s_{n+1} \sim P(\cdot|s_n, a_n) \coloneqq P_{s_n,a_n}$ for $n \in \mathbb{N}^+$. To derive the policy corresponding to the value function, we define the optimal state-action function $Q^\star : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ as the expected cumulative discounted rewards under the optimal policy, $Q^\star(s, a) \coloneqq \sup_{\pi\in\Pi} \mathbb{E}_{\pi,P}[\sum_{n=1}^{\infty} \gamma^{n-1} r(s_n, a_n)|s_1 = s, a_1 = a]$. The optimal state-action function Q^* is also the fixed point of the Bellman optimality equation,

$$Q^\star(s, a) = r(s, a) + \gamma \mathbb{E}_{s'\sim P}[\max_{a'\in\mathcal{A}} Q^\star(s', a')]. \quad (1)$$

2.2. Q-learning

Our model-free algorithmic design relies on the Q-learning template, originally designed to solve the non-robust Bellman optimality equation (Equation 1). Q-learning is a model-free reinforcement learning algorithm that uses a single sample trajectory to update the estimator for the Q function incrementally. Suppose at time n, we draw a sample (s_n, a_n, r_n, s'_n) from the environment. Then, the algorithm updates the estimated Q-function following:

$$Q_{n+1}(s_n, a_n) = (1 - \alpha_n)Q_n(s_n, a_n) + \alpha_n(r_n + \gamma \max_{a'\in\mathcal{A}} Q_n(s'_n, a')),$$

Here, $\alpha_n > 0$ is a learning rate. The algorithm updates the estimated Q function by constructing a unbiased estimator for the true Q value, i.e., $r_n + \gamma \max_{a'\in\mathcal{A}} Q_n(s'_n, a')$ using one sample.

2.3. Distributionally Robust MDPs

DRRL learns an optimal policy that is robust to unknown environmental changes, where the transition model P and reward function r may differ in the test environment. To focus on the perturbation of the transition model, we assume no pertubation to the reward function. Our approach adopts the notion of distributional robustness, where the true transition model P is unknown but lies within an ambiguity set $\mathcal{P}$ that contains all transition models that are close to the training environment under some probability distance D. To ensure computational feasibility, we construct the ambiguity set $\mathcal{P}$ in the (s, a)-rectangular manner, where for each $(s, a) \in \mathcal{S} \times \mathcal{A}$, we define the ambiguity set $\mathcal{P}_{s,a}$ as,

$$\mathcal{P}_{s,a} \coloneqq \{P'_{s,a} : \Delta_S | D(P'_{s,a} \| P_{s,a}) \le \rho\}. \quad (2)$$

We then build the ambiguity set for the whole transition model as the Cartesian product of every (s, a)-ambiguity set, i.e., $\mathcal{P} = \prod_{(s,a)\in\mathcal{S}\times\mathcal{A}} \mathcal{P}_{s,a}$. Given $\mathcal{P}$, we define the optimal DR state-action function $Q^\star$ as the value function of the best policy to maximize the worst-case return over the ambiguity set,

$$Q^{\mathrm{rob},\star}(s, a) \coloneqq \sup_{\pi\in\Pi} \inf_{P\in\mathcal{P}} \mathbb{E}_{\pi,P}[\sum_{n=1}^{\infty} \gamma^{n-1} r(s_n, a_n)|s_1 = s, a_1 = a].$$

Under the (s, a)-rectangular assumption, the Bellman optimality equation has been established by Iyengar (2005); Xu & Mannor (2010),

$$\begin{aligned} Q^{\mathrm{rob},\star}(s, a) =& \mathcal{T}_k(Q^{\mathrm{rob},\star})(s, a) \\ \coloneqq& r(s, a) + \gamma \inf_{P\in\mathcal{P}} \mathbb{E}_{s'\sim P}[\max_{a'\in\mathcal{A}} Q^{\mathrm{rob},\star}(s', a')]. \end{aligned} \quad (3)$$

For notation simplicity, we would ignore the superscript rob.

3. Distributonally Robust Q-learning with Single Trajectory

This section presents a general model-free framework for DRRL. We begin by instantiating the distance D as Cressie-Read family of f-divergence (Cressie & Read, 1984), which is designed to recover previous common choices such as the χ^2 and KL divergence. We then discuss the challenges and previous solutions in solving the corresponding DRRL problem, as described in Section 3.2. Finally, we present the design idea of our three-timescale framework and establish the corresponding convergence guarantee.

3.1. Divergence Families

Previous work on DRRL has mainly focused on one or several divergences, such as KL, χ^2, and total variation (TV) divergences. In contrast, we provide a unified framework that applies to a family of divergences known as the Cressie-Read family of f-divergences. This family is parameterized

by $k \in (-\infty, \infty)/\{0, 1\}$, and for any chosen k, the Cressie-Read family of f-divergences is defined as

$$D_{f_k}(Q\|P) = \int f_k(\frac{dP}{dQ})dQ,$$

with $f_k(t) := \frac{t^k - kt + k - 1}{k(k-1)}$. Based on this family, we instantiate our ambiguity set in Equation 2 as $\mathcal{P}_{s,a} = \{P'_{s,a} : \Delta_S | D_{f_k}(P'_{s,a}\|P_{s,a}) \leq \rho\}$ for some radius $\rho > 0$. The Cressie-Read family of f-divergence includes χ^2-divergence ($k = 2$) and KL divergence ($k \to 1$).

One key challenge in developing DRRL algorithms using the formulation in Equation 3 is that the expectation is taken over the ambiguity set $\mathcal{P}$, which is computationally intensive even with the access to the center model P. Since we only have access to samples generated from the possibly misspecific model P, estimating the expectation with respect to other models $P' \in \mathcal{P}$ is even more challenging. While importance sampling-based techniques can achieve this, the cost of high variance is still undesirable. To solve this issue, we rely on the dual reformulation of Equation 3:

Lemma 3.1 ((Duchi & Namkoong, 2021)). *For any random variable $X \sim P$, define $\sigma_k(X, \eta) = -c_k(\rho)\mathbb{E}_P[(\eta - X)_+^{k_*}]^{\frac{1}{k_*}} + \eta$ with $k_* = \frac{k}{k-1}$ and $c_k(\rho) = (1 + k(k-1)\rho)^{\frac{1}{k}}$. Then*

$$\inf_{Q \ll P}\{\mathbb{E}_Q[X] : D_{f_k}(Q\|P) \leq \rho\} = \sup_{\eta \in \mathbb{R}} \sigma_k(X, \eta), \quad (4)$$

Here $(x)_+ = \max\{x, 0\}$. Equation 4 shows that protecting against the distribution shift is equivalent to optimizing the tail-performance of a model, as only the value below the dual variable η are taken into account. Another key insight from the reformulation is that as the growth of $f_k(t)$ for large t becomes steeper for larger k, the f-divergence ball shrinks and the risk measure becomes less conservative. This bridges the gap between difference divergences, whereas previous literature, including Yang et al. (2022) and Zhou et al. (2021), treats different divergences as separate. By applying the dual reformulation, we can rewrite the Cressie-Read Bellman operator in Equation 3 as

$$\mathcal{T}_k(Q)(s, a) = r(s, a) + \gamma \sup_{\eta \in \mathbb{R}} \sigma_k(\max_{a' \in \mathcal{A}} Q(\cdot, a'), \eta). \quad (5)$$

3.2. Bias in Plug-in Estimator in Single Trajectory Setting

In this subsection, we aim to solve Equation 5 using single-trajectory data, which has not been addressed by previous DRRL literature. As we can only observe one newly arrival sample each time, to design a online model-free DRRL algorithm, we need to approximate the expectation in Equation 5 using that single sample properly. As mentioned in Section 2.2, the design of the Q-learning algorithm **relies on an one-sample unbiased estimator of the true Bellman operator**. However, this convenience vanishes in the DR Bellman operator. To illustrate this, consider plugging only one sample into the Cressie-Read Bellman operator Equation 5:

$$\begin{aligned} &r(s, a) + \gamma \sup_{\eta \in \mathbb{R}}\{\eta - c_k(\rho)(\eta - \max_{a'} Q(s', a'))_+\} \\ &\quad = r(s, a) + \gamma \max_{a'} Q(s', a'). \end{aligned}$$

This reduces to the non-robust Bellman operator and is obviously not an unbiased estimator for $\mathcal{T}_k(Q)$. This example reveals the inherently more challenging nature of the online DRRL problem. Whereas non-robust RL only needs to improve the expectation of the cumulative return, improving the worst-case return requires more information about the system dynamics, which seems hopeless to be obtained from only one sample and sharply contrasts with our target.

Even with the help of batch samples, deriving an appropriate estimator for the DR Bellman operator is still nontrivial. Consider a standard approach to construct estimators, sample average approximation (SAA): given a batch of sample size n starting from a fix state-action pair (s, a), i.e., $D_n = \{(s_i, a_i, s'_i, r_i), i \in [n], (s_i, a_i) = (s, a)\}$, the SAA empirical Bellman operator is defined as:

$$\widehat{\mathcal{T}}_k(Q)(s, a, D_n) = r(s, a) + \gamma \sup_{\eta \in \mathbb{R}} \widehat{\sigma}_k(\max_{a' \in \mathcal{A}} Q(\cdot, a'), \eta, D_n).$$

Here, $\widehat{\sigma}_k$ is the empirical Cressie-Read functional defined as

$$\widehat{\sigma}_k := -c_k(\rho)[\sum_{i \in [n]} (\eta - \max_{a' \in \mathcal{A}} Q(s'_i, a'))_+^{k_*}/n]^{\frac{1}{k_*}} + \eta.$$

As pointed out by Liu et al. (2022), the SAA estimator is biased, prompting the introduction of the multilevel Monte-Carlo method (Blanchet & Glynn, 2015). Specifically, it first obtains $N \in \mathbb{N}^+$ samples from the distribution $\mathbb{P}(N = n) = p_n = \epsilon(1-\epsilon)^n$, and then uses the simulator to draw 2^{N+1} samples $D_{2^{N+1}}$. The samples are further decomposed into two parts: $D_{:2^N}$ consists of the first 2^N samples, while $D_{2^N+1:}$ contains the remaining samples. Finally, the DR term in Equation 5 is approximated by solving three optimization problems:

$$\widehat{\mathcal{T}}_k(Q)(s, a, D_n) = r_1 + \max_{a' \in \mathcal{A}} Q(s'_1, a') + \frac{\Delta^q_{N,\delta}(Q)}{p_N},$$

$$\begin{aligned} \Delta^q_{N,\delta}(Q) := &\sup_{\eta \geq 0} \widehat{\sigma}_k(\max_{a' \in \mathcal{A}} Q(\cdot, a'), \eta, D_{2^{N+1}}) \\ &- \frac{1}{2}\sup_{\eta \geq 0} \widehat{\sigma}_k(\max_{a' \in \mathcal{A}} Q(\cdot, a'), \eta, D_{:2^N}) \\ &- \frac{1}{2}\sup_{\eta \geq 0} \widehat{\sigma}_k(\max_{a' \in \mathcal{A}} Q(\cdot, a'), \eta, D_{2^N+1:}). \end{aligned}$$

However, this multilevel Monte-Carlo solution requires a large batch of samples for the same state-action pair before the next update, resulting in unbounded memory costs/computational time that are not practical. Furthermore, it is prohibited in the single-trajectory setting, where each step only one sample can be observed. Our experimental results show that simply approximating the Bellman operator with simulation data, without exploiting its structure, suffers from low data efficiency.

3.3. Three-timescale Framework

The Q-learning is solving the nonrobust Bellman operator's fixed point in a stochastic approximation manner. A salient feature in the DR Bellman operator, compared with its nonrobust counterpart, is a bi-level optimization nature, i.e., jointly solving the dual parameter η and the fixed point Q of the Bellman optimality equation. We revisit the stochastic approximation view of the Q-learning and develop a three-timescale framework, by a faster running estimate of the optimal dual parameter, and a slower update of the Q table.

To solve Equation 5 using a stochastic approximation template, we iteratively update the variables η and Q table as follows: for the n-th iteration after observing a new transition sample (s_n, a_n, s'_n, r_n) and some learning rates $\zeta_1, \zeta_2 > 0$,

$$\eta_{n+1} = \eta_n - \zeta_1 * \text{Gradient of } \eta_n,$$
$$Q_{n+1} = r_n + \zeta_2 \gamma \sigma_k(\max_{a' \in \mathcal{A}} Q_n(\cdot, a'), \eta_n).$$

As the update of η and Q relies on each other, we keep the learning speeds of η and Q, i.e., ζ_1 and ζ_2, different to stabilize the training process. Additionally, due to the (s,a)-rectangular assumption, η is independent across different (s,a)-pairs, while the Q table depends on each other. The independent structure for η allows it to be estimated more easily; so we approximate it in a faster loop, while for Q we update it in a slower loop.

3.4. Algorithmic Design

In this subsection, we further instantiate the three-timescale framework to the Cressie-Read family of f-divergences. First, we compute the gradient of $\sigma_k(\max_{a' \in \mathcal{A}} Q(\cdot, a'), \eta)$ in Equation 5 with respect to η.

Lemma 3.2 (Sub-Gradient of the σ_k dual function)**.**

$$\partial \sigma_k(\max_{a' \in \mathcal{A}} Q(\cdot, a'), \eta) \in$$

$$\begin{cases} \{-c_k(\rho) Z_1^{\frac{1}{k^*}-1} \cdot Z_2 + 1\}, & \eta > \max_{a' \in A} Q(\cdot, a'), \\ [-c_k(\rho) Z_1^{\frac{1}{k^*}-1} \cdot Z_2 + 1, 0], & \eta = \max_{a' \in \mathcal{A}} Q(\cdot, a'), \\ \{1\}, & \eta < \max_{a' \in \mathcal{A}} Q(\cdot, a'), \end{cases} \quad (6)$$

Algorithm 1 Distributionally Robust Q-learning with Cressie-Read family of f-divergences

1: **Input:** Exploration rate ϵ, Learning rates $\{\zeta_i(n)\}_{i \in [3]}$, Cressie-Read family parameter k, Ambiguity set radius ρ.
2: **Init:** Initialize Q, Z and η with zero.
3: **for** $n = 1, 2, \cdots$ **do**
4: Observe the state s_n, execute the action $a_n = \arg\max_{a \in \mathcal{A}} Q(s_n, a)$ using ϵ-greedy policy
5: Observe the reward r_n and next state s'_n
6: Update

$$Z_1(s_n, a_n) \leftarrow (1 - \zeta_1(n)) Z_1(s_n, a_n) + \zeta_1(n)(\eta(s_n, a_n) - \max_a Q(s'_n, a))_+^{k_*},$$
$$Z_2(s_n, a_n) \leftarrow (1 - \zeta_1(n)) Z_2(s_n, a_n) + \zeta_1(n)(\eta(s_n, a_n) - \max_a Q(s'_n, a))_+^{k_*-1}.$$

7: Update
$\eta(s_n, a_n) \leftarrow \eta(s_n, a_n) + \zeta_2(n)(-c_k(\rho) Z_1^{\frac{1}{k_*}-1}(s_n, a_n) \cdot Z_2(s_n, a_n) + 1)$.
8: Update
$Q(s_n, a_n) \leftarrow (1 - \zeta_3(n)) Q(s_n, a_n) + \zeta_3(n)(r_n - \gamma(c_k(\rho) Z_1^{\frac{1}{k_*}}(s_n, a_n) - \eta(s_n, a_n)))$.
9: **end for**

where

$$Z_1 = \mathbb{E}_P[(\eta - \max_{a' \in \mathcal{A}} Q(\cdot, a'))_+^{k_*}], \quad (7)$$
$$Z_2 = \mathbb{E}_P[(\eta - \max_{a' \in \mathcal{A}} Q(\cdot, a'))_+^{k_*-1}]. \quad (8)$$

Due to the nonlinearity in Equation 6, the plug-in gradient estimator is in fact biased. The bias arises as for a random variable X, $\mathbb{E}[f(X)] \neq f(\mathbb{E}[X])$ for $f(x) = x^{\frac{1}{k_*}-1}$ in $Z_1^{\frac{1}{k_*-1}}$. To address this issue, we introduce another even faster timescale to estimate Z_1 and Z_2,

$$Z_1(s_n, a_n) \leftarrow (1 - \zeta_1(n)) Z_1(s_n, a_n) + \zeta_1(n)(\eta(s_n, a_n) - \max_{a'} Q(s'_n, a'))_+^{k_*}, \quad (9)$$

$$Z_2(s_n, a_n) \leftarrow (1 - \zeta_1(n)) Z_2(s_n, a_n) + \zeta_1(n)(\eta(s_n, a_n) - \max_{a'} Q(s'_n, a'))_+^{k_*-1}. \quad (10)$$

In the medium timescale, we approximate $\eta^\star(s,a) := \arg\max_{\eta \in \mathcal{R}} \sigma_k(\max_{a' \in \mathcal{A}} Q(s, a'), \eta)$ by incrementally update the dual variable η using the stochastic gradient descent method, where the true gradient computed in Equation 6 is

approximated by:

$$\eta(s_n,a_n) \leftarrow \eta(s_n,a_n) + \zeta_2(n)(-c_k(\rho)Z_1^{\frac{1}{k_*}-1}(s_n,a_n)\cdot Z_2(s_n,a_n)+1). \tag{11}$$

Finally, we update the DR Q function in the slowest timescale using Equation 12,

$$Q(s_n,a_n) \leftarrow (1-\zeta_3(n))Q(s_n,a_n) + \zeta_3(n)\widehat{\mathcal{T}}_{n,k}(Q)(s_n,a_n), \tag{12}$$

where $\widehat{\mathcal{T}}_{n,k}(Q)(s,a)$ is the empirical version of Equation 5 in the n-th iteration:

$$\widehat{\mathcal{T}}_{n,k}(Q)(s_n,a_n) = r_n - \gamma(c_k(\rho)Z_1^{\frac{1}{k_*}}(s_n,a_n) - \eta(s_n,a_n)).$$

Here $\zeta_1(n),\zeta_2(n)$ and $\zeta_3(n)$ are learning rates for three timescales at time n, which will be specified later. We summarize the ingredients into our DR Q-learning (DRQ) algorithm (Algorithm 1), and prove the almost surely (a.s.) convergence of the algorithm as Theorem 3.3. The proof is deferred in Appendix C.

Theorem 3.3. *The estimators at the n-th step in Algorithm 1, $(Z_{n,1},Z_{n,2},\eta_n,Q_n)$, converge to $(Z_1^\star,Z_2^\star,\eta^\star,Q^\star)$ a.s. as $n\to\infty$, where $\eta^\star$ and $Q^\star$ are the fixed-point of the equation $Q=\mathcal{T}_k(Q)$, and $Z_1^\star$ and $Z_2^\star$ are the corresponding quantity under $\eta^\star$ and $Q^\star$.*

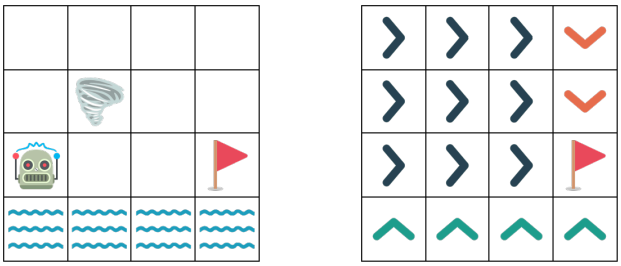
The proof establishes that, by appropriately selecting stepsizes to prioritize frequent updates of $Z_{n,1}$ and $Z_{n,2}$, followed by η_n, and with Q_n updated at the slowest rate, the solution path of $(Z_{n,1},Z_{n,2},\eta_n,Q_n)$ closely tracks a system of three-dimensional ordinary differential equations (ODEs) considering martingale noise. Our approach is to generalize the classic machinery of two-timescale stochastic approximation (Borkar, 2009) to a three-timescale framework, and use it to analyze our proposed algorithm. See Appendix B for the detailed proof.

4. Experiments

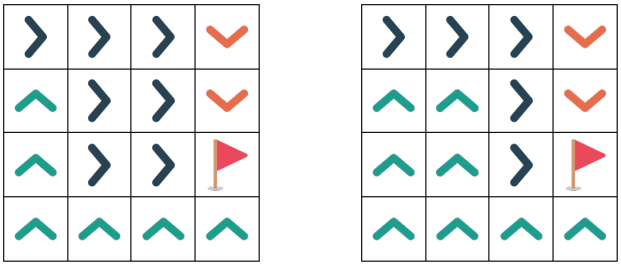
We demonstrate the robustness and sample complexity of our DRQ algorithm in the Cliffwalking environment (Delétang et al., 2021) and American put option environment (deferred in Appendix A). These environments provide a focused perspective on the policy and enable a clear understanding of the key parameters effects. We develop a deep learning version of DRQ and compare it with practical online and offline (robust) RL algorithms in classical control tasks, LunarLander and CartPole.

4.1. Convergence and Sample Complexity

Before we begin, let us outline the key findings and messages conveyed in this subsection: **(1) Our ambiguity set design provides substantial robustness**, as demonstrated through comparisons with non-robust Q-learning and R-contamination ambiguity sets (Wang & Zou, 2021). **(2) Our DRQ algorithm exhibits desirable sample complexity**, significantly outperforming the multi-level Monte Carlo based DRQ algorithm proposed by Liu et al. (2022) and comparable to the sample complexity of the model-based DRRL algorithm by Panaganti & Kalathil (2022).



(a) Environment (b) Nonrobust

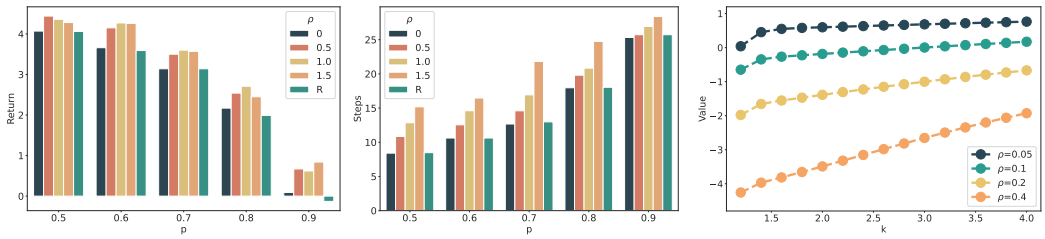


(c) $\rho = 1.0$ (d) $\rho = 1.5$

Figure 1. The Cliffwalking environment and the learned policies for different ρ's.

Experiment Setup: The Cliffwalking task is commonly used in risk-sensitive RL research (Delétang et al., 2021). Compared to the Frozen Lake environment used by Panaganti & Kalathil (2022), Cliffwalking offers a more intuitive visualization of robust policies (see Figure 1). The task involves a robot navigating from an initial state of $(2,0)$ to a goal state of $(2,3)$. At each step, the robot is affected by wind, which causes it to move in a random direction with probability p. Reaching the goal state earns a reward of $+5$, while encountering a wave in the water region $\{(3,j) \mid 0 \le j \le 3\}$ results in a penalty of -1. We train the agent in the nominal environment with $p = 0.5$ for 3 million steps per run, using an ϵ-greedy exploration strategy with $\epsilon = 0.1$. We evaluate its performance in perturbed environments, varying the choices of k and ρ to demonstrate different levels of robustness. We set the stepsize parameters according to Assumption B.1: $\zeta_1(t) = 1/(1+(1-\gamma)t^{0.6})$, $\zeta_2(t) = 1/(1+0.1(1-\gamma)t^{0.8})$, and $\zeta_3(t) = 1/(1+0.05(1-\gamma)*t)$, where the discount factor is $\gamma = 0.9$.

Robustness: To evaluate the robustness of the learned poli-



(a) Return (b) Episode length (c) Value of various k and ρ

Figure 2. Averaged return and steps with 100 random seeds in the perturbed environments. $\rho = 0$ corresponds to the non-robust Q-learning. R denotes the R-contamination ambiguity set.

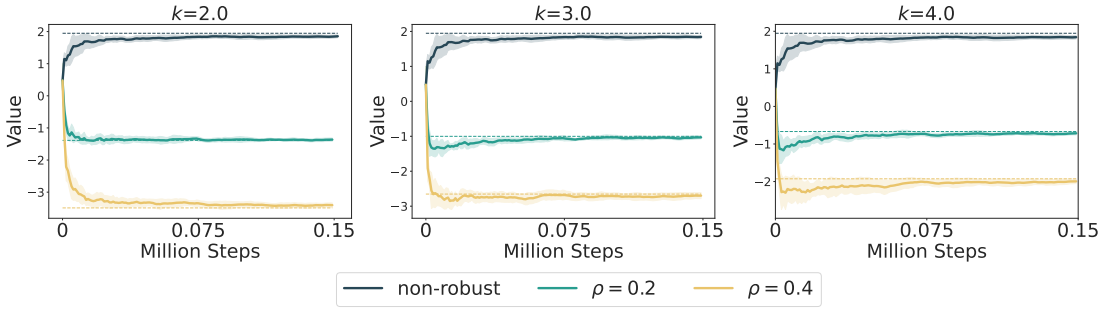


Figure 3. The training curves in the Cliffwalking environment. Each curve is averaged over 100 random seeds and shaded by their standard deviations. The dashed line is the optimal robust value with corresponding k and ρ.

cies, we compare their cumulative returns in perturbed environments with $p \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$ over 100 episodes per setting. We visulize the decision at each status in Figure 1 with different robustness level ρ. In particular, the more robust policy tends to avoid falling into the water, thus arrives to the goal state with a longer path by keeping going up before going right. Figure 2a shows the return distribution for each policy. Figure 2b displays the time taken for the policies to reach the goal, and the more robust policy tends to spend more time, which quantitatively supports our observations in Figure 1. Interestingly, we find that the robust policies outperform the nonrobust one even in the nominal environment. For the different ρ's, $\rho = 1.0$ is the best within a relatively wide range ($p \in \{0.6, 0.7, 0.8\}$), while $\rho = 1.5$ is preferred in the environment of extreme pertubation ($p = 0.9$). This suggests that DRRL provides a elegant trade-off for different robustness preferences.

We also compare our model-free DRRL algorithm with the robust RL algorithm presented in Wang & Zou (2021), which also supports training using a single trajectory. The algorithm in Wang & Zou (2021) uses an R-contamination ambiguity set. We select the best value of R from 0.1 to 0.9 and other detailed descriptions in Appendix A. In most cases, the R-contamination based algorithm performs very similarly to the non-robust benchmark, and even performs worse in some cases (i.e., $p = 0.8$ and 0.9), due to its excessive conservatism. As we mentioned in Section 3.1, larger k would render the the risk measure less conservative and thus less sensitive to the change in the ball radius ρ, which is empirically confirmed by Figure 2c.

Sample Complexity: The training curves in Figure 3 depict the estimated value $\max_a \widehat{Q}(s_0, a)$ (solid line) and the optimal robust value $V^*(s_0)$ (dashed line) for the initial state s_0. The results indicate that the estimated value converges quickly to the optimal value, regardless of the values of k and ρ. Importantly, our DRQ algorithm achieves a similar convergence rate to the non-robust baseline (represented by the black line). We further compare our algorithm with two robust baselines: the DRQ algorithm with a weak simulator proposed by Liu et al. (2022) (referred to as *Liu's*), and the model-based algorithm introduced by Panaganti & Kalathil (2022) (referred to as *Model*) in Figure 4. To ensure a fair comparison, we set the same learning rate, $\zeta_3(t)$, for our DRQ algorithm and the Q-table update loop of the Liu's algorithm, as per their recommended choices.

Our algorithm converges to the true DR value at a similar rate as the model-based algorithm, while the Liu's algorithm exhibits substantial deviation from the true value and converges relatively slowly. Our algorithm's superior sample efficiency is attributed to the utilization of first-order information to approximate optimal dual variables, whereas Liu's relies on a large amount of simulation data for an unbiased

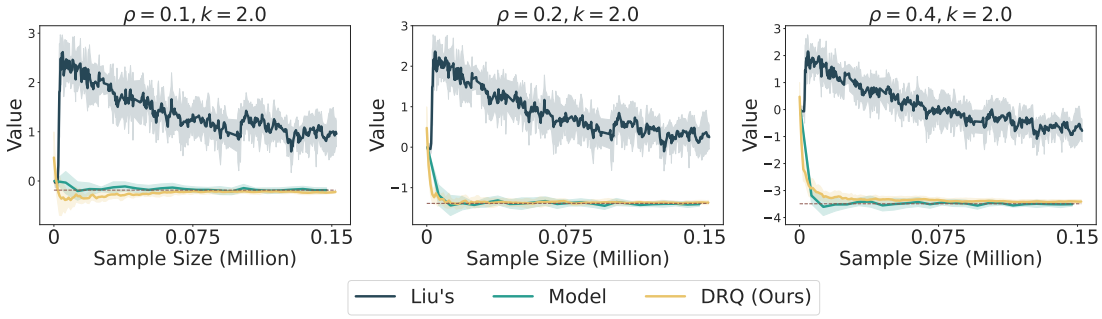


Figure 4. Sample complexity comparisons in Cliffwalking environment with Liu's and Model-based algorithms. Each curve is averaged over 100 random seeds and shaded by their standard deviations.

estimator.

4.2. Practical Implementation

We validate the practicality of our DRQ framework by implementing a practical version, called the Deep Distributionally Robust Q-learning (*DDRQ*) algorithm, based on the DQN algorithm (Mnih et al., 2015). We apply this algorithm to two classical control tasks from the OpenAI Gym (Brockman et al., 2016): CartPole and LunarLander.

Our practical algorithm, denoted as Algorithm 2, is a variant of Algorithm 1. Specifically, we adopt the Deep Q-Network (DQN) architecture (Mnih et al., 2015) and employ two sets of neural networks as functional approximators. One set, Q_{θ_1} and Q_{θ_2}, serves as approximators for the Q function, while the other set, η_{θ_3} and η_{θ_4}, approximates the distributionally robust dual variable η. To enhance training stability, we introduce a target network, Q_{θ_2}, for the fast Q network Q_{θ_1} and η_{θ_4} for the fast dual variable η network η_{θ_3}.

Due to the approximation error introduced by neural networks and to further improve sample efficiency, our practical DDRQ algorithm adopts a two-timescale update approach. In this approach, our Q network aims to minimize the Bellman error, while the dual variable η network strives to maximize the DR Q value defined in Equation 5. It's important to note that the two-timescale update approach could introduce bias in the convergence of the dual variable, and thus the dual variable η may not the optimal dual variable for the primal problem. Given the primal-dual structure of this DR problem, this could render an even lower target value for the Q network to learn. This approach can be understood as a robust update strategy for our original DRRL problem, share some spirits to the optimization techniques used in other algorithms like Variational Autoencoders (VAE)(Kingma & Welling, 2013), Proximal Policy Optimization (PPO)(Schulman et al., 2017), and Maximum a Posteriori Policy Optimization (MPO) (Abdolmaleki et al., 2018). Additional experimental details can be found in Appendix A.3.

To assess the effectiveness of our DDRQ algorithm, we compare it against the RFQI algorithm (Panaganti et al., 2022), the soft-robust RL algorithm (Derman et al., 2018), and the non-robust DQN and FQI algorithms. This comparison encompasses representative practical (robust) reinforcement learning algorithms for both online and offline datasets. To evaluate the robustness of the learned policies, we introduce action and physical environment perturbations. For action perturbation, we simulate the perturbations by varying the probability ϵ of randomly selecting an action for both CartPole and LunarLander tasks. We test with $\epsilon \in \{0, 0.1, 0.2, \cdots, 1.0\}$ for CartPole and $\epsilon \in \{0, 0.1, 0.2, \cdots, 0.6\}$ for LunarLander. Regarding physical environment perturbation in LunarLander, we decrease the power of all the main engine and side engines by the same proportions, ranging from 0 to 0.6. For CartPole, we reduce the "force mag" parameter from 0.2 to 0.8. We set the same ambiguity set radius for both our DDRQ and RFQI algorithm for fair comparisons. Figure 5 illustrates how our DDRQ algorithm successfully learns robust policies across all tested tasks, achieving comparable performance to other robust counterparts such as RFQI and SR-DQN. Conversely, the non-robust DQN and FQI algorithms fail to learn robust policies and deteriorate significantly even under slight perturbations. It is worth noting that RFQI does not perform well in the LunarLander environment, despite using the official code provided by the authors. This outcome could be attributed to the restriction to their TV distance in constructing the ambiguity set, while our Creass-Read ambiguity set can be flexibily chosen to well adopted to the environment nature. Additionally, the soft-robust RL algorithm requires generating data based on multiple models within the ambiguity set. This process can be excessively time-consuming, particularly in large-scale applications.

5. Conclusion

In this paper, we introduce our DRQ algorithm, a fully model-free DRRL algorithm trained on a single trajectory.

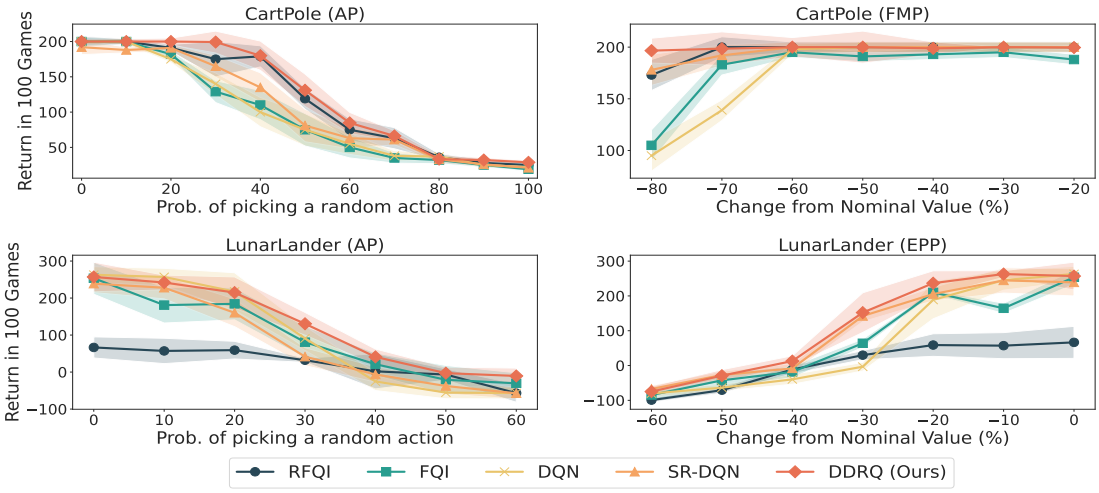


Figure 5. The return in the CartPole and LunarLander environment. Each curve is averaged over 100 random seeds and shaded by their standard deviations. AP: Action Perturbation; FMP: Force Mag Perturbation; EPP: Engines Power Perturbation.

By leveraging the stochastic approximation framework, we effectively tackle the joint optimization problem involving the state-action function and the DR dual variable. Through an extension of the classic two-timescale stochastic approximation framework, we establish the asymptotic convergence of our algorithm to the optimal DR policy. Our extensive experimentation showcases the convergence, sample efficiency, and robustness improvements achieved by our approach, surpassing non-robust methods and other robust RL algorithms. Our DDRQ algorithm further validates the practicality of our algorithmic framework.

Acknowledgements

This work is generously supported by the General Research Fund [Grants 16208120, and 16214121] from the Hong Kong Research Grants Council, the NSF grants CCF-2312205 and CCF-2312204.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning, especially to enhance the robustness of the widely-used reinforcement learning algorithms. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Abdolmaleki, A., Springenberg, J. T., Tassa, Y., Munos, R., Heess, N., and Riedmiller, M. Maximum a posteriori policy optimisation. *arXiv preprint arXiv:1806.06920*, 2018.

Abdullah, M. A., Ren, H., Ammar, H. B., Milenkovic, V., Luo, R., Zhang, M., and Wang, J. Wasserstein Robust Reinforcement Learning, 2019. URL http://arxiv.org/abs/1907.13196.

Badrinath, K. P. and Kalathil, D. Robust reinforcement learning using least squares policy iteration with provable performance guarantees. In *International Conference on Machine Learning*, pp. 511–520. PMLR, 2021.

Blanchet, J. H. and Glynn, P. W. Unbiased monte carlo for optimization and functions of expectations via multi-level randomization. In *2015 Winter Simulation Conference (WSC)*, pp. 3656–3667. IEEE, 2015.

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

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

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

Cox, J. C., Ross, S. A., and Rubinstein, M. Option pricing: A simplified approach. *Journal of financial Economics*, 7 (3):229–263, 1979.

Cressie, N. and Read, T. R. Multinomial goodness-of-fit tests. *Journal of the Royal Statistical Society: Series B (Methodological)*, 46(3):440–464, 1984.

Delétang, G., Grau-Moya, J., Kunesch, M., Genewein, T., Brekelmans, R., Legg, S., and Ortega, P. A. Model-free risk-sensitive reinforcement learning. *arXiv preprint arXiv:2111.02907*, 2021.

Derman, E., Mankowitz, D. J., Mann, T. A., and Mannor, S. Soft-robust actor-critic policy-gradient. *arXiv preprint arXiv:1803.04848*, 2018.

Duchi, J. C. and Namkoong, H. Learning models with uniform performance via distributionally robust optimization. *The Annals of Statistics*, 49(3):1378–1406, 2021.

Goyal, V. and Grand-Clement, J. Robust markov decision processes: Beyond rectangularity. *Mathematics of Operations Research*, 2022.

Ho, C. P., Petrik, M., and Wiesemann, W. Partial policy iteration for l1-robust markov decision processes. *J. Mach. Learn. Res.*, 22:275–1, 2021.

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

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

Lim, S. H., Xu, H., and Mannor, S. Reinforcement learning in robust markov decision processes. *Advances in Neural Information Processing Systems*, 26, 2013.

Liu, Z., Bai, Q., Blanchet, J., Dong, P., Xu, W., Zhou, Z., and Zhou, Z. Distributionally robust q-learning. In *International Conference on Machine Learning*, pp. 13623–13643. PMLR, 2022.

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

Mannor, S., Simester, D., Sun, P., and Tsitsiklis, J. N. Bias and variance in value function estimation. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 72, 2004.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533, 2015.

Neufeld, A. and Sester, J. Robust q-learning algorithm for markov decision processes under wasserstein uncertainty. *ArXiv*, abs/2210.00898, 2022.

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

Panaganti, K. and Kalathil, D. Sample complexity of robust reinforcement learning with a generative model. In *International Conference on Artificial Intelligence and Statistics*, pp. 9582–9602. PMLR, 2022.

Panaganti, K., Xu, Z., Kalathil, D., and Ghavamzadeh, M. Robust reinforcement learning using offline data. *Advances in neural information processing systems*, 35: 32211–32224, 2022.

Roy, A., Xu, H., and Pokutta, S. Reinforcement learning under model mismatch. *Advances in neural information processing systems*, 30, 2017.

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

Shi, L. and Chi, Y. Distributionally Robust Model-Based Offline Reinforcement Learning with Near-Optimal Sample Complexity. URL http://arxiv.org/abs/2208.05767.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Tamar, A., Mannor, S., and Xu, H. Scaling up robust mdps using function approximation. In *International conference on machine learning*, pp. 181–189. PMLR, 2014.

Tsitsiklis, J. N. Asynchronous stochastic approximation and q-learning. *Machine learning*, 16:185–202, 1994.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., Oh, J., Horgan, D., Kroiss, M., Danihelka, I., Huang, A., Sifre, L., Cai, T., Agapiou, J. P., Jaderberg, M., Vezhnevets, A. S., Leblond, R., Pohlen, T., Dalibard, V., Budden, D., Sulsky, Y., Molloy, J., Paine, T. L., Gulcehre, C., Wang, Z., Pfaff, T., Wu, Y., Ring, R., Yogatama, D., Wünsch, D., McKinney, K., Smith, O., Schaul, T., Lillicrap, T., Kavukcuoglu, K., Hassabis, D., Apps, C., and Silver, D. Grandmaster level in StarCraft II using multi-agent reinforcement learning. 575(7782):350–354, 2019. doi: 10.1038/s41586-019-1724-z.

Wang, Y. and Zou, S. Online robust reinforcement learning with model uncertainty. *Advances in Neural Information Processing Systems*, 34:7193–7206, 2021.

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

Xu, H. and Mannor, S. Distributionally robust markov decision processes. *Advances in Neural Information Processing Systems*, 23, 2010.

Yang, I. Wasserstein distributionally robust stochastic control: A data-driven approach. *IEEE Transactions on Automatic Control*, 66:3863–3870, 2018.

Yang, W., Zhang, L., and Zhang, Z. Toward theoretical understandings of robust markov decision processes: Sample complexity and asymptotics. *The Annals of Statistics*, 50(6):3223–3248, 2022.

Zhou, Z., Zhou, Z., Bai, Q., Qiu, L., Blanchet, J., and Glynn, P. Finite-sample regret bound for distributionally robust offline tabular reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 3331–3339. PMLR, 2021.

Appendix

In the subsequent sections, we delve into the experimental specifics and provide the technical proofs that were not included in the primary content.

In Section A, we commence by showcasing an additional experiment on the American call option. This aligns with the convergence and sample complexity discussions from the main content. We then elucidate the intricacies of Liu's algorithm to facilitate a transparent comparison with our methodology. Lastly, we discuss the algorithmic intricacies of our DDRQ algorithm and provide details on the experiments that were previously omitted.

In Section B, to prove Theorem 3.3, we begin by extending the two-timescale stochastic approximation framework to a three-timescale one. Following this, we adapt it to our algorithm, ensuring all requisite conditions are met.

A. Additional Experiments Details

A.1. Experiment on the American Put Option Problem

In this section, we present additional experimental results from a simulated American put option problem (Cox et al., 1979) that has been previously studied in robust RL literature (Zhou et al., 2021; Tamar et al., 2014). The problem involves holding a put option in multiple stages, whose payoff depends on the price of a financial asset that follows a Bernoulli distribution. Specifically, the next price s_{h+1} at stage $h+1$ follows,

$$s_{h+1} = \begin{cases} c_u s_h, & \text{w.p. } p_0, \\ c_d s_h, & \text{w.p. } 1-p_0, \end{cases} \tag{13}$$

where the c_u and c_d are the price up and down factors and p_0 is the probability that the price goes up. The initial price s_0 is uniformly sampled from $[\kappa-\epsilon, \kappa+\epsilon]$, where $\kappa = 100$ is the strike price and $\epsilon = 5$ in our simulation. The agent can take an action to exercise the option ($a_h = 1$) or not exercise ($a_h = 0$) at the time step h. If exercising the option, the agent receives a reward $\max(0, \kappa - s_h)$ and the state transits into an exit state. Otherwise, the price will fluctuate based on the above model and no reward will be assigned. Moreover we introduce a discount structure in this problem, i.e., the 1 reward in the stage $h+1$ worths γ in stage h as our algorithm is designed for discounted RL setting. In our experiments, we set $H = 5$, $c_u = 1.02$, $c_d = 0.98$ and $\gamma = 0.95$. We limit the price in $[80, 140]$ and discretize with the precision of 1 decimal place. Thus the state space size $|\mathcal{S}| = 602$.

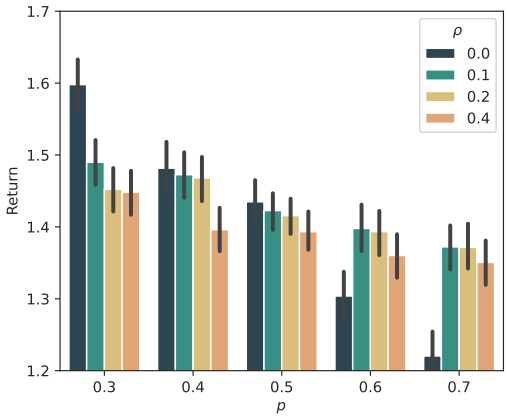


Figure 6. Averaged return in the American call option problem. $\rho = 0.0$ is the non-robust Q-learning.

We first demonstrate the robustness gain of our DR Q-learning algorithm by comparing with the non-robust Q-learning algorithm, and investigate the effect of different robustness levels by varying ρ. Each agent is trained for 10^7 steps with an ϵ-greedy exploration policy of $\epsilon = 0.2$ and evaluated in perturbed environments. We use the same learning rates for the three timescales in our DR Q-learning algorithm as in the Cliffwalking environment: $\zeta_1(t) = 1/(1+(1-\gamma)t^{0.6})$, $\zeta_2(t) = 1/(1+0.1*(1-\gamma)t^{0.8})$, and $\zeta_3(t) = 1/(1+0.01*(1-\gamma)t)$. For the non-robust Q-learning we set the same learning rate as in our Q-update, i.e., $\zeta_3(t)$. We perturb the transition probability to the price up and down status

$p = \{0.3, 0.4, 0.5, 0.6, 0.7\}$, and evaluate each agent for 5000 episodes. Figure 6 reports the average return and one standard deviation level. The non-robust Q-learning performs best when the price tends to decrease and the market gets more benefitial ($p = \{0.3, 0.4, 0.5\}$), which benefits the return of holding an American put option. However, when the prices tend to increase and the market is riskier ($p = \{0.6, 0.7\}$), our DR Q-learning algorithm significantly outperforms the non-robust counterpart, demonstrating the robustness gain of our algorithm against worst-case scenarios.

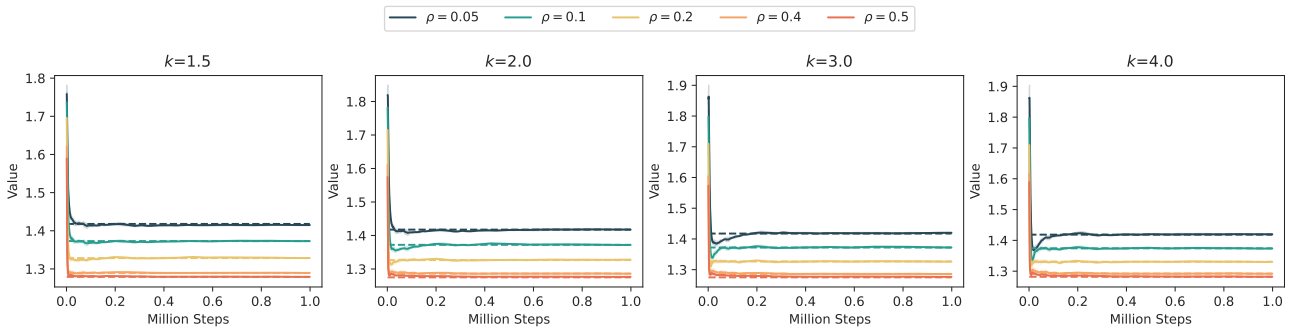


Figure 7. Convergence curve of DR Q-learning algorithm to the true DR value under different ρ's and k's. Each curve is averaged over 10 random seeds and shaded by their standard deviation. The dashed line is the optimal robust value with corresponding k and ρ.

We present the learning curve of our DR Q-learning algorithm with different ρ in Figure 7. Our algorithm can accurately learn the DR value under different ρ's and k's within 0.1 million steps. We compare the sample efficiency of our algorithm with the DR Q-learning algorithm in Liu et al. (2022) (referred to as *Liu's*) and the model-based algorithm in Panaganti & Kalathil (2022) (referred to as *Model*). We set a smaller learning rate for Liu's as $\zeta(t) = 1/(1 + (1 - \gamma)t)$. The reason is setting the same learning rate $\zeta_3(t)$ for their algorithm would render a much slower convergence performance, which is not fair for comparisons. We use the recommended choice $\varepsilon = 0.5$ for the sampling procedure in Liu algorithm. Both DR Q-learning and Liu are trained for $5 * 10^7$ steps per run, while the model-based algorithm is trained for 10^6 steps per run to ensure sufficient samples for convergence. As shown in Figure 8, the model-based approach is the most sample-efficient, converging accurately to the optimal robust value with less than 10^4 samples. Our DR Q-learning algorithm is slightly less efficient, using 10^5 samples to converge. Liu algorithm is significantly less efficient, using 10^7 samples to converge. Note that the model-based approach we compared here is to first obtain samples for each state-action pairs, and then conduct the learning procedure to learn the optimal robust value. In particular, we need to specify the number of samples for each state-action pair n. Then the total number of samples used is the sum of all these number, i.e., $S \times A \times n$, whose computation manner is different from that in the model-free algorithms we used where each update requires one or a batch of new samples.

To ensure self-containment, we provide the pseudocode for our implemented Liu algorithm (Algorithm 3) and the model-based algorithm (Algorithm 2) below. These algorithms were not originally designed to solve the ambiguity set constructed by the Cressie-Read family of f-divergences.

A.2. Liu's Algorithm Descriptions

In this subsection, we provide the pseudo-code for the Liu algorithm, represented in Algorithm 2. Our intention is to emphasize the differences in algorithmic design between their approach and ours.

Their algorithm, in particular, relies extensively on multi-level Monte Carlo, requiring the sampling of a batch of samples for each state-action pair. Once they estimate the Doubly Robust (DR) value for a specific state-action pair, the samples are promptly discarded and subsequently resampled from a simulator. To summarize, their algorithm exhibits significant distinctions from ours in terms of algorithmic design.

A.3. Practical Experiments

In this section, we provide a comprehensive description of our Deep Distributionally Robust Q-learning (DDRQ) algorithm, as illustrated in Algorithm 2, along with its experimental setup in the context of CaroPole and LunarLander.

Most of the hyperparameters are set the same for both LunarLander and CartPole. We choose Cressie-Read family parameter

Algorithm 2 Distributionally Robust Deep Q-learning with Cressie-Read family of f-divergences

Input: Discount Factor γ, Radius of robustness ρ, Cressie-Read family parameter k, Q-network target update rate τ_Q and η-network target update rate τ_η, mini-batch size N, maximum number of iterations T, start training timestep T_{tr}, training network update frequency F_{tr} and target network update frequency F_{up}.
Init: Two state-action neural networks Q_{θ_1} and Q_{θ_2}, two dual neural network η_{θ_1} and η_{θ_2}, $C = (1 + k * (k-1) * \rho)^{1/k}$.
for for $t = 1, \cdots, T$ **do**
  Observe a state s_t and execute an action a_t using ϵ-greedy policy.
  if $t \geq T_{tr}$ and $t\%F_{tr}$ **then**
    Sample a minibatch B with N samples from the replay buffer.
    Compute next-state target value for Q network

$$Q_i = r_t - \gamma C * (\eta_{\theta_1}(s_i, a_i) - \max_{a \in \mathcal{A}} Q_{\theta_1}(s_i, a_i))_+^{k*}, \quad \forall i \in B$$

    and for η network

$$Q_i' = r_t - \gamma C * (\eta_{\theta_2}(s_i, a_i) - \max_{a \in \mathcal{A}} Q_{\theta_2}(s_i, a_i))_+^{k*}, \quad \forall i \in B.$$

    Update $\theta_1 = \arg\min_\theta \sum_i (Q_i - Q_\theta(s_i, a_i))^2$.
    Update $\theta_3 = \arg\max_\theta \sum_i Q_i'(\theta)$.
  end if
  if $t \geq T_{tr}$ and $t\%F_{up}$ **then**
    Update target network $\theta_2 = (1 - \tau_Q)\theta_2 + \tau_Q\theta_2$, $\theta_4 = (1 - \tau_\eta)\theta_4 + \tau_\eta\theta_3$.
  end if
end for
$t = t + 1$

Environment	Maximum Training Step T	ϵ_{End}	τ_Q	τ_η
CartPole	$1e8$	0.05	1	0.05
LunarLander	$3e7$	0.2	0.5	0.1

Table 1. Different Hyperparamers between CartPole and LunarLander

$k = 2$, which is indeed the χ^2 ambiguity set and we set ambiguity set radius as $\rho = 0.3$. For RFQI we also use the same ρ for fair comparison. Our replay buffer size is set $1e6$ and the batch size for training is set 4096. Our fast Q and η network are update every 10 steps ($F_{tr} = 10$) and the target networks are updated every 500 steps ($F_{up} = 500$). The learning rate for Q network is 2.5×10^{-4} and for η network is 2.5×10^{-3}. The Q network and the η network both employ a dual-layer structure, with each layer consisting of 120 dimensions. For exploration scheme, we choose epsilon-greedy exploration with linearly decay epsilon with ending ϵ_{End}. The remain parameters tuned for each environments are referred in Table 1.

B. Multiple Timescale Convergence

We fix some notations that will be used in the following proof. For a positive integer n, $[n]$ denotes the set $\{1, 2, \cdots, n\}$. $|A|$ denotes the cardinality of the set A. We adopt the standard asymptotic notations: for two non-negative sequences a_n and b_n, $a_n = O(b_n)$ iff $\limsup_{n\to\infty} a_n/b_n < \infty$. Δ_d is the simplex on a d dimensional space, i.e., $\Delta_d = \{x : \sum_{i=1}^d x_i = 1, x_i \geq 0, \forall i \in [d]\}$. For any vector $x \in \mathbb{R}^d$ and any semi-positive matrix $A \in \mathbb{R}^{d\times d}$ with $A \succeq 0$, we denote $\|x\|_A := \sqrt{x^\top Ax}$. $\|\cdot\|$ is Euclidean norm.

B.1. Three Timescales Convergence Analysis

In this subsection, we outline the roadmap for establishing the a.s. convergence of the Algorithm 1. For ease of presentation, our analysis is given for the synchronous case, where every entry of the Q function is updated at each timestep. Extension to the asynchronous case, where only one state-action pair entry is updated at each timestep, follows Tsitsiklis (1994). Our

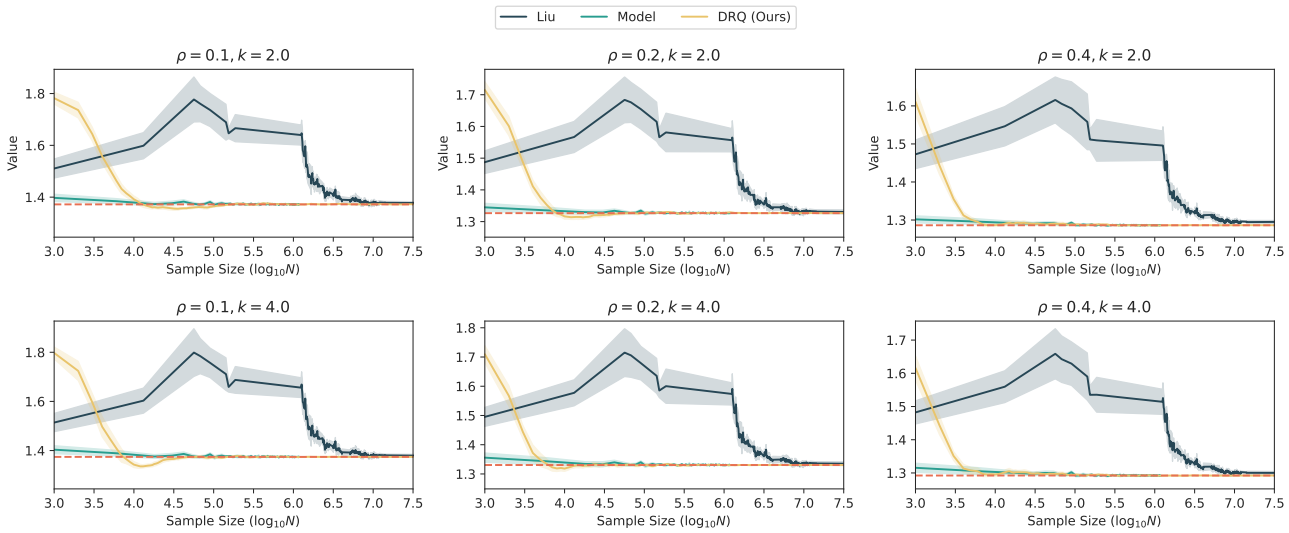


Figure 8. Sample complexity comparisons in American option environment with other DRRL algorithms. The dashed line is the optimal robust value with corresponding k and ρ. The x-axis is in log10 scale. Each curve is averaged over 10 random seeds and shaded by their one standard deviation. The dashed line is the optimal robust value with corresponding k and ρ.

approach is to generalize the classic machinery of two-timescale stochastic approximation (Borkar, 2009) to a three-timescale framework, and use it to analyze our proposed algorithm. We rewrite the Algorithm 1 as

$$Z_{n+1} = Z_n + \zeta_1(n)[f(Z_n, \eta_n, Q_n) + M_n^Z], \tag{14}$$
$$\eta_{n+1} = \eta_n + \zeta_2(n)[g(Z_n, \eta_n, Q_n) + \epsilon_n^\eta], \tag{15}$$
$$Q_{n+1} = Q_n + \zeta_3(n)[h(Z_n, \eta_n, Q_n) + \epsilon_n^Q]. \tag{16}$$

Here, we use $Z_n = (Z_{n,1}, Z_{n,2})$ to represent the $Z_{n,1}$ and $Z_{n,2}$ jointly. To echo with our algorithm, $f = (f_1, f_2)$ and $M_n^Z = (M_{n,1}^Z, M_{n,2}^Z)$ are defined as,

$$f_1(Z_n, \eta_n, Q_n)(s,a) = \mathbb{E}_{s'}[(\eta_n(s,a) - \max_{a'} Q_n(s',a'))_+^{k_*} - Z_{n,1}(s,a)],$$
$$f_2(Z_n, \eta_n, Q_n)(s,a) = \mathbb{E}_{s'_n}[(\eta_n(s,a) - \max_{a'} Q_n(s',a'))_+^{k_*-1} - Z_{n,2}(s,a)],$$
$$M_{n,1}^Z(s,a) = (\eta_n(s,a) - \max_{a'} Q_n(s',a'))_+^{k_*} - Z_{n,1}(s,a) - f_1(Z_n, \eta_n, Q_n)(s,a),$$
$$M_{n,2}^Z(s,a) = (\eta_n(s,a) - \max_{a'} Q_n(s',a'))_+^{k_*-1} - Z_{n,2}(s,a) - f_2(Z_n, \eta_n, Q_n)(s,a).$$

In the update of η_n (Equation 15), g and ϵ_n^η are defined as

$$g(Z_n, \eta_n, Q_n)(s,a) = -c_k(\rho)\mathbb{E}[(\eta_n(s,a) - \max_{a' \in \mathcal{A}} Q_n(s',a'))_+^{k_*}]^{\frac{1}{k_*}-1} \cdot \mathbb{E}[(\eta_n(s,a) - \max_{a' \in \mathcal{A}} Q_n(s',a'))_+^{k_*-1}] + 1,$$
$$\epsilon_n^\eta(s,a) = -c_k(\rho) Z_{n,1}^{\frac{1}{k_*}-1}(s,a) \cdot Z_{n,2}(s,a) + 1 - g(Z_n, \eta_n, Q_n)(s,a).$$

Finally in the update of Q_n (Equation 16), h and ϵ_n^Q are defined as

$$h(Z_n, \eta_n, Q_n)(s,a) = r(s,a) - \gamma(c_k(\rho)(\mathbb{E}_P[(\eta_n(s,a) - \max_{a' \in \mathcal{A}} Q_n(s',a'))_+^{k_*}])^{\frac{1}{k_*}} - \eta_n(s,a)),$$
$$\epsilon_n^Q(s,a) = r(s,a) - \gamma(c_k(\rho) Z_{n,1}^{\frac{1}{k_*}}(s,a) - \eta_n(s,a)) - h(Z_n, \eta_n, Q_n)(s,a).$$

The algorithm 1 approximates the dynamic described by the system of f, g and h through samples along a single trajectory, with the resulting approximation error manifesting as martingale noise M_n^Z conditioned on some filtration $\mathcal{F}_n$ and the error terms ϵ_n^η and ϵ_n^Q.

To analyze the dynamic of algorithm 1, we first obtain the continuous dynamic of f, g, and h using ordinary differential equations (ODEs) analysis. The second step is to analyze the stochastic nature of the noise term M_n^Z and the error terms ϵ_n^η and ϵ_n^Q, to ensure that they are negligible compared to the main trend of f, g, and h, which is achieved by the following stepsizes,

Condition B.1. The stepsizes $\zeta_i(n), i = 1, 2, 3$ satisfy

$$\sum_n \zeta_i(n) = \infty, \quad \sum_n \zeta_i^2(n) < \infty, \quad \zeta_1(n) = o(\zeta_2(n)), \quad \zeta_2(n) = o(\zeta_3(n)).$$

These stepsize schedules satisfy the standard conditions for stochastic approximation algorithms, ensuring that **(1).** the key quantities in gradient estimator Z_n update on the fastest timescale, **(2).** the dual variable for the DR problem, η_n, update on the intermediate timescale; and **(3).** the Q table updates on the slowest timescale. Examples of such stepsize are $\zeta_1(n) = \frac{1}{1+n^{0.6}}, \zeta_2(n) = \frac{1}{1+n^{0.8}}$ and $\zeta_3(n) = \frac{1}{1+n}$. Notably, the first two conditions in Condition B.1 ensure the martingale noise is negligible. The different stepsizes for the three loops specificed by the third and fourth conditions ensures that $Z_{n,1}$ and $Z_{n,2}$ are sufficiently estimated with respect to the η_n and Q_n, and these outer two loops are free from bias or noise in the stochastic approximation sense.

Under Condition B.1, when analyzing the behavior of the Z_n, the η_n and the Q_n can be viewed as quasi-static. To study the behavior of the fastest loop, we analyze the following ODEs:

$$\dot{Z}(t) = f(Z(t), \beta(t), Q(t)), \quad \dot{\eta}(t) = 0, \quad \dot{Q}(t) = 0, \tag{17}$$

and prove that ODEs (17) a.s. converge to $\lambda_1^{''}(\eta, Q)$ for proper η and Q and some mapping $\lambda_1^{''}$. Similarly, Q_n can be viewed as fixed when analyzing the behavior of η_n, and the corresponding ODEs to understand its behavior are

$$\dot{\eta}(t) = g(\lambda_1''(\eta(t), Q(t)), \eta(t), Q(t)), \quad \dot{Q}(t) = 0. \tag{18}$$

By exploiting the dual form of the distributionally robust optimization problem, we can prove these ODEs converge to the set $\{\lambda_1'(Q), \lambda_2'(Q), Q | Q \in V\}$ for some mapping λ_1' and λ_2' with V is the set containing all the mapping from $\mathcal{S}$ to $\mathbb{R}$. Lastly, we examine the slowest timescale ODE given by

$$\dot{Q}(t) = h(\lambda_1'(Q(t)), \lambda_2'(Q(t)), Q(t)), \tag{19}$$

and employ our analysis to establish the almost sure convergence of Algorithm 1 to the globally optimal pair $(Z_1^\star, Z_2^\star, \eta^\star, Q^\star)$.

Lemma B.2 (Discrete Gronwall inequality)**.** *Let $\{x_n, n \geq 0\}$ (resp. $\{a_n, n \geq 0\}$) be nonnegative (resp. positive) sequences and $C, L \geq 0$ scalars such that for all n,*

$$x_{n+1} \leq C + L\left(\sum_{m=0}^{n} a_m x_m\right).$$

Then for $T_n = \sum_{m=0}^{n} a_m$,

$$x_{n+1} \leq Ce^{LT_n}.$$

Lemma B.3 (Gronwall inequality)**.** *For continuous $u(\cdot), v(\cdot) \geq 0$ and scalars $C, K, T \geq 0$*

$$u(t) \leq C + K\int_0^t u(s)v(s)ds, \quad \forall t \in [0, T],$$

implies

$$u(t) \leq Ce^{K\int_0^T v(s)ds}, \quad \forall t \in [0, T].$$

B.2. Stability Criterion

Consider the stochastic approximation scheme $z_n \in \mathbb{R}^N$ given by

$$z_{n+1} = z_n + a_n\left[g\left(z_n\right) + M_{n+1}\right],$$

with the following Condition:

Condition B.4. $g : \mathbb{R}^N \to \mathbb{R}^N$ is Lipschitz.

Condition B.5. The sequence $\{a_n\} \subset \mathbb{R}$ satisfies $\sum_n a_n = \infty, \sum_n a_n^2 < \infty$.

Condition B.6. $\{M_n\}$ is a martingale difference sequence with respect to the filtration $\mathcal{F}_n = \sigma\left(z_m, M_m, m \leq n\right)$, there exists $K > 0$ such that $E\left[\|M_{n+1}\|^2 \mid \mathcal{F}_n\right] \leq K(1 + \|z_n\|^2)$ a.s..

Condition B.7. The functions $g_d(z) = g(dz)/d, d \geq 1$ satisfy $g_d(z) \to g_\infty(z)$ as $d \to \infty$ uniformly on compacts for some continuous function $g_\infty : \mathbb{R}^N \to \mathbb{R}^N$. In addition, the ODE

$$\dot{z}(t) = g_\infty(z(t))$$

has the origin as its globally asymptotically stable equilibrium.

We then have

Lemma B.8. *Under Condition B.4 to B.6, we have $\sup_n \|z_n\| < \infty$ a.s.*

See Section 2.2 and 3.2 in Borkar (2009) for the proof. As the stability proofs in Section 3.2 of Borkar (2009) are path-wise, we can apply this result to analyze multiple timescales dynamic.

B.3. Three Timescales Convergence Criterion

Consider the scheme

$$x_{n+1} = x_n + a_n \left[f\left(x_n, y_n, z_n\right) + M_{n+1}^{(1)} \right] \tag{20}$$

$$y_{n+1} = y_n + b_n \left[g\left(x_n, y_n, z_n\right) + M_{n+1}^{(2)} \right] \tag{21}$$

$$z_{n+1} = z_n + c_c \left[h\left(x_n, y_n, z_n\right) + M_{n+1}^{(3)} \right] \tag{22}$$

where $f : \mathbb{R}^{d+k+p} \to \mathbb{R}^d$, $g : \mathbb{R}^{d+k+p} \to \mathbb{R}^k$, $h : \mathbb{R}^{d+k+p} \to \mathbb{R}^p$, $\{M_n^{(i)}\}, i = 1, 2, 3$ are martingale difference sequences with respect to the σ-fields $\mathcal{F}_n = \sigma\left(x_m, y_m, M_m^{(1)}, M_m^{(2)}, M_m^{(3)}; m \leq n\right)$, and the a_n, b_n, c_n form decreasing stepsize sequences.

It is instructive to compare the stochastic update algorithms from Equations 20 to 22 with the following o.d.e.,

$$\dot{x}(t) = \frac{1}{a} f(x(t), y(t), z(t)),$$
$$\dot{y}(t) = \frac{1}{b} g(x(t), y(t), z(t)),$$
$$\dot{z}(t) = \frac{1}{c} h(x(t), y(t), z(t)),$$

in the limit that $a, b, c \to 0$ and $a = o(b)$, $c = o(b)$.

We impose the following conditions, which are necessary for the a.s. convergence for each timescale itself and are commonly used in the literature of stochastic approximation algorithms, e.g., (Borkar, 2009).

Condition B.9. f and g is L-Lipschitz map for some $0 < L < \infty$ and h is bounded.

Condition B.10.

$$\sum_n a_n = \sum_n b_n = \sum_n c_n = \infty, \sum_n (a_n^2 + b_n^2 + c_n^2) < \infty, \text{ and } b_n = o(a_n), c_n = o(b_n).$$

Condition B.11. For $i = 1, 2, 3$ and $n \in \mathbb{N}^+$, $\{M_n^{(i)}\}$ is a martingale differeence sequence with respect to the increasing family of σ-fields $\mathcal{F}_n$. Furthermore, there exists some $K > 0$, such that for $i = 1, 2, 3$ and $n \in \mathbb{N}^+$,

$$\mathbb{E}[\|M_{n+1}^{(i)}\|^2 | \mathcal{F}_n] \leq K(1 + \|x_n\|^2 + \|y_n\|^2 + \|z_n\|^2).$$

Condition B.12. $\sup_n(\|x_n\| + \|y_n\| + \|z_n\|) < \infty$, a.s..

Condition B.13. For each $y \in \mathbb{R}^k$ and $z \in \mathbb{R}^p$, $\dot{x}(t) = f(x(t), y, z)$ has a globally asymptotically stable equilibrium $\lambda_1(y, z)$, where $\lambda_1 : \mathcal{R}^{k+p} \to \mathcal{R}^d$ is a L-Lipschitz map for some $L > 0$.

Condition B.14. For each $z \in \mathbb{R}^p$, $\dot{y}(t) = g(\lambda_1(y(t), z), y(t), z)$ has a globally asymptotically stable equilibrium $\lambda_2(z)$, where $\lambda_2 : \mathcal{R}^p \to \mathcal{R}^k$ is a L-Lipschitz map for some $L > 0$.

Condition B.15. $\dot{z}(t) = h(\lambda_1(z(t)), \lambda_2(z(t)), z(t))$ has a globally asymptotically stable equilibrium $z^\star$.

Conditions B.9, B.10, B.11 and B.12 are necessary for the a.s. convergence for each timescale itself. Moreover, Condition B.12 itself requires Conditions like B.9, B.10, B.11, with an extra condition like Condition B.6. Instead, we need to prove the boundedness for each timescale, thus the three timescales version is as follow

Condition B.16. The ODE

$$\dot{z}(t) = f_\infty(x(t), y, z)$$
$$\dot{y}(t) = g_\infty(\lambda_1(y(t), z), y(t), z)$$
$$\dot{z}(t) = h_\infty(\lambda_1(z(t)), \lambda_2(z(t)), z(t))$$

all have the origin as their globally asymptotically stable equilibrium for each $y \in \mathcal{R}^k$ and $z \in \mathcal{R}^p$, where

$$f_\infty = \lim_{d \to \infty} \frac{f(dx)}{d}, \quad g_\infty = \lim_{d \to \infty} \frac{g(dx)}{d}, \text{ and } h_\infty = \lim_{d \to \infty} \frac{h(dx)}{d}.$$

We have the following results, which appears as a three timescales extension of Lemma 6.1 in Borkar (2009) and serves as a auxiliary lemma for the our a.s. convergence.

Lemma B.17. *Under the conditions B.9, B.10, B.11 and B.12.* $(x_n, y_n, z_n) \to \{\lambda_1'(z), \lambda_2'(z), z : z \in \mathcal{R}^p\}$ *a.s..*

Proof. Rewrite Equations 21 and 22 as

$$y_{n+1} = y_n + a_n \left[\epsilon_{1,n} + M_{n+1}^{(2)'}\right]$$
$$z_{n+1} = z_n + a_n \left[\epsilon_{2,n} + M_{n+1}^{(3)'}\right],$$

where $\epsilon_{1,n} = \frac{b_n}{a_n} g(x_n, y_n, z_n)$, $\epsilon_{2,n} = \frac{c_n}{a_n} h(x_n, y_n, z_n)$, $M_{n+1}^{(2)'} = \frac{b_n}{a_n} M_{n+1}^{(2)}$, $M_{n+1}^{(3)'} = \frac{c_n}{a_n} M_{n+1}^{(3)}$. Note that $\epsilon_{1,n}, \epsilon_{2,n} \to 0$ as $n \to \infty$. Consider them as the special case in the third extension in Section 2.2 in Borkar (2009) and then we can conclude that (x_n, y_n, z_n) converges to the internally chain transitive invariant sets of the o.d.e.,

$$\dot{x}(t) = h(x(t), y(t), z(t))$$
$$\dot{y}(t) = 0$$
$$\dot{z}(t) = 0,$$

which implies that $(x_n, y_n, z_n) \to \{\lambda_1'(y, z), y, z : y \in \mathcal{R}^k, z \in \mathcal{R}^p\}$.

Rewrite Equation 22 again as

$$z_{n+1} = z_n + b_n \left[\epsilon_{2,n}' + M_{n+1}^{(3)''}\right],$$

where $\epsilon_{2,n}' = \frac{c_n}{b_n} h(x_n, y_n, z_n)$ and $M_{n+1}^{(3)''} = \frac{c_n}{b_n} M_{n+1}^{(3)}$. We use the same extension again and can conclude that (x_n, y_n, z_n) converges to the internally chain transitive invariant sets of the o.d.e.,

$$\dot{y}(t) = g(\lambda_1'(y(t)), y(t), z(t))$$
$$\dot{z}(t) = 0.$$

Thus $(x_n, y_n, z_n) \to \{\lambda_1(y), \lambda_2(z), z : z \in \mathcal{R}^p\}$. □

Theorem B.18. *Under the Condition B.9 to B.16,* $(x_n, y_n, z_n) \to (\lambda_1(z^*), \lambda_2(z^*), z^*)$.

Proof. Let $t(0) = 0$ and $t(n) = \sum_{i=0}^{n-1} c_i$ for $n \geq 1$. Define the piecewise linear continuous function $\tilde{z}(t), t \geq 0$ where $\tilde{z}(t(n)) = z_n$ and $\tilde{z}(t) = \frac{t(n+1)-t}{t(n+1)-t(n)} z_{n+1} + \frac{t-t(n)}{t(n+1)-t(n)} z_n$ for $t \in [t(n), t(n+1)]$ with any $n \in N$. Let $\psi_n = \sum_{i=0}^{n-1} c_i M_{i+1}^{(3)}, n \in \mathbb{N}^+$. For any $t \geq 0$, denote $[t] = \max\{s(n) : s(n) \leq t\}$. Then for $n, m \geq 0$, we have

$$\begin{aligned}
\tilde{z}(t(n+m)) &= \tilde{z}(t(n)) + \sum_{k=1}^{m-1} c_{n+k} h(x_{n+k}, y_{n+k}, z_{n+k}) + (\psi_{m+n+1} - \psi_n) \\
&= \tilde{z}(t(n)) + \int_{t(n)}^{t(n+m)} h(\lambda_1(z(s)), \lambda_2(z(s)), z(s)) ds \\
&+ \int_{t(n)}^{t(n+m)} (h(\lambda_1(z([s])), \lambda_2(z([s])), z([s])) - h(\lambda_1(z(s)), \lambda_2(z(s)), z(s))) ds \\
&+ \sum_{k=0}^{m-1} c_{n+k} (h(x_{n+k}, y_{n+k}, z_{n+k}) - h(\lambda_1(z_{n+k}), \lambda_2(z_{n+k}), z_{n+k})) \\
&+ (\psi_{n+m+1} - \psi_n). \qquad (23)
\end{aligned}$$

We further define $z^{t(n)}(t)$ as the trajectory of $\dot{z}(t) = g(\lambda_1(z(t)), \lambda_2(z(t)), z(t))$ with $z^{t(n)}(t(n)) = \tilde{z}(t(n))$.

$$z^{t(n)}(t(n+m)) = \tilde{z}(t(n)) + \int_{t(n)}^{t(n+m)} h(\lambda_1(z^{t(n)}(s)), \lambda_2(z^{t(n)}(s)), z^{t(n)}(s)) ds. \qquad (24)$$

Taking the difference between Equation 23 and the Equation 24 we have

$$
\begin{aligned}
&|\tilde{z}(t(n+m)) - z^{t(n)}(t(n+m))| \\
&= \underbrace{\sum_{k=0}^{m-1} c_{n+k}(h(\lambda_1(\tilde{z}(t+k)), \lambda_2(\tilde{z}(t+k)), \tilde{z}(t+k)) - h(\lambda_1(z(t(n+k))), \lambda_2(z(t(n+k))), z(t(n+k))))}_{} \\
&\quad + \underbrace{|\int_{t(n)}^{t(n+m)} (h(\lambda_1(z([t])), \lambda_2(z([t])), z([t])) - h(\lambda_1(z(s)), \lambda_2(z(s)), z(s)))ds|}_{\text{I}} \\
&\quad + \underbrace{|\sum_{k=1}^{m-1} c_{n+k}(h(x_{n+k}, y_{n+k}, z_{n+k}) - h(\lambda_1(z_{n+k}), \lambda_2(z_{n+k}), z_{n+k}))|}_{\text{II}} \\
&\quad + \underbrace{|\psi_{n+m+1} - \psi_n|}_{\text{III}}.
\end{aligned}
$$

We analyze the I term. For notation simplicity we ignore the supsript $t(n)$.

$$
\begin{aligned}
&|h(\lambda_1(z([t])), \lambda_2(z([t])), z([t])) - h(\lambda_1(z(t)), \lambda_2(z(t)), z(t))| \\
&= |(h(\lambda_1(z([t])), \lambda_2(z([t])), z([t])) - h(\lambda_1(z([t])), \lambda_2(z([t])), z(t)))| \\
&\quad + |(h(\lambda_1(z([t])), \lambda_2(z([t])), z(t)) - h(\lambda_1(z([t])), \lambda_2(z([t])), z([t])))| \\
&= |(h(\lambda_1(z([t])), \lambda_2(z([t])), z([t])) - h(\lambda_1(z([t])), \lambda_2(z(t)), z(t)))| \\
&\quad + |h(\lambda_1(z([t])), \lambda_2(z([t])), z(t)) - h(\lambda_1(z([t])), \lambda_2(z([t])), z(t))| \\
&\quad + |(h(\lambda_1(z([t])), \lambda_2(z([t])), z(t)) - h(\lambda_1(z([t])), \lambda_2(z([t])), z([t])))|.
\end{aligned} \tag{25}
$$

By the Lipschitzness of the h we have

$$
\|h(x) - h(0)\| \leq L\|x\|,
$$

which implies

$$
\|h(x)\| \leq \|h(0)\| + L\|x\|.
$$

$$
\begin{aligned}
\|z^{t(n)}(t)\| &\leq \|\tilde{z}(s)\| + \int_s^t \|h(z^{t(n)}(s))\|ds \\
&\leq \|\tilde{z}(s)\| + \int_s^t (\|h(0)\| + L\|z^{t(n)}(s)\|)ds \\
&\leq (\|\tilde{z}(s)\| + \|h(0)\|T) + L\int_s^t \|z^{t(n)}(s)\|ds.
\end{aligned}
$$

By Gronwall's inequality (Lemma B.3), we have

$$
\|z^{t(n)}(t)\| \leq (C + \|h(0)\|T)e^{LT}, \quad \forall t \in [t(n), t(n+m)].
$$

Thus for all $t \in [t(n), t(n+m)]$, we have

$$
\|h(\lambda_1(z^{t(n)}(t)), \lambda_2(z^{t(n)}(t)), z^{t(n)}(t))\| \leq C_T \coloneqq \|h(0)\| + L(C + \|h(0)\|T)e^{LT} < \infty, a.s..
$$

For any $k \in [m-1]$ and $t \in [t(n+k), t(n+k+1)]$,

$$
\begin{aligned}
\|z^{t(n)}(t) - z^{t(n)}(t(n+k))\| &\leq \|\int_{t(n+k)}^t h(\lambda_1(z^{t(n)}(s)), \lambda_2(z^{t(n)}(s)), z^{t(n)}(s))ds\| \\
&\leq C_T(t - t(n+k)) \\
&\leq C_T a(n+k),
\end{aligned}
$$

where the last inequality is from the construction of $\{t(n) : n \in \mathbb{N}^+\}$. Finally we can conclude

$$\begin{aligned}
&\|\int_{t(n)}^{t(n+m)} (h(\lambda_1(z([s])), \lambda_2(z([s])), z(s)) - h(\lambda_1(z([s])), \lambda_2(z([s])), z([s])))ds\| \\
&\leq \int_{t(n)}^{t(n+m)} L\|z(s) - z([s])\|ds \\
&= L\sum_{k=0}^{m-1} \int_{t(n+k)}^{t(n+k-1)} \|z(s) - z(t(n+k))\|ds \\
&\leq C_T L \sum_{k=0}^{m-1} c_{n+k}^2 \\
&\leq C_T L \sum_{k=0}^{\infty} c_{n+k}^2 \to 0, a.s..
\end{aligned}$$

For the III term, it converges to zero from the martingale convergence property.

Subtracting equation 23 from 24 and take norms, we have

$$\begin{aligned}
&\|\tilde{z}(t(n+m)) - z^{t(n)}(t(n+m))\| \\
&\leq L\sum_{i=0}^{m-1} c_{n+i}\|\tilde{z}(t(n+i)) - z^{t(n)}(t(n+i))\| \\
&\quad + C_T L \sum_{k\geq 0} c_{n+k}^2 + \sup_{k\geq 0}\|\delta_{n,n+k}\|, a.s..
\end{aligned}$$

Define $K_{T,n} = C_T L \sum_{k\geq 0} c_{n+k}^2 + \sup_{k\geq 0}\|\delta_{n,n+k}\|$. Note that $K_{T,n} \to 0$ a.s. $n \to \infty$. Let $u_i = \|\tilde{x}(t(n+i)) - x^{t(n)}(t(n+i))\|$. Thus, above inequality becomes

$$u_m \leq K_{T,n} + L\sum_{i=0}^{m-1} c_{n+i}u_i.$$

Thus the above inequality becomes

$$z(t(n+m)) \leq K_{T,n} + L\sum_{k=0}^{m-1} c_k z(t(n+k)).$$

Note that $u_0 = 0$ and $\sum_{i=0}^{m-1} b_i \leq T$, then using the discrete Gronwall lemma (Lemma B.2) we have

$$\sup_{0\leq i\leq m} u_i \leq K_{T,n}e^{LT}.$$

Following the similar logic as in Lemma 1 in Borkar (2009), we can extend the above result to the case $\|\tilde{z}(t) - z^{t(n)}(t)\| \to 0$ where $t \in [0, T]$.

Then using the proof of Theorem 2 of Chapter 2 in Borkar (2009), we get $z_n \to z^*$ a.s. and thus by Lemma B.17 the proof can be concluded. □

C. Convergence of the DR Q-learning Algorithm

Before we start the proof of the DR Q-learning algorithm, we first introduce the following lemma.

Lemma C.1. *Denote* $\eta^* = \arg\max_\eta \sigma_k(X, \eta) = -c_k(\rho)\mathbb{E}_P[(\eta - X)_+^{k_*}]^{\frac{1}{k_*}} + \eta$. *Given that* $X(\omega) \in [0, M]$, *then we have* $\eta* \in [0, \frac{c_k(\rho)}{c_k(\rho)-1}M]$.

Proof. Note that for $\eta = \min_\omega X(\omega)$, $-c_k(\rho)\mathbb{E}_P[(\eta - X)_+^{k_*}]^{\frac{1}{k_*}} + \eta = \min_\omega X(\omega) \geq 0$. Also we know that when $\eta \geq \frac{c_k(\rho)}{c_k(\rho)-1}M$,

$$\begin{aligned}
&- c_k(\rho)\mathbb{E}_P[(\eta - X)_+^{k_*}]^{\frac{1}{k_*}} + \eta \\
\leq &- c_k(\rho)\mathbb{E}_P[(\eta - M)_+^{k_*}]^{\frac{1}{k_*}} + \eta \\
= &- c_k(\rho)(\eta - M) + \eta \\
\leq &0.
\end{aligned}$$

Then we can conclude that $\eta^* \leq \frac{c_k(\rho)}{c_k(\rho)-1}M$. Moreover, as $X(\omega) \geq 0$, we know $\sigma_k(X, 0) = 0$, which concludes that $\eta* \in [0, \frac{c_k(\rho)}{c_k(\rho)-1}M]$. □

Note that $Q_n \in [0, \frac{1}{1-\gamma}]$ when reward is bounded by $[0, 1]$. Thus $M = \frac{1}{1-\gamma}$ in our case and then we denote $\overline{\eta} = \frac{c_k(\rho)}{c_k(\rho)-1}M$. Now we are ready to prove the convergence of the DR Q-learning algorithm. For theoretical analysis, we consider the clipping version of our DR Q-learning algorithm.

Proof of Theorem 3.3. We define the filtration generated by the historical trajectory,

$$\mathcal{F}_n = \sigma(\{(s_t, a_t, s_t', r_t)\}_{t\in[n-1]}, s_n, a_n).$$

In the following analysis, we fix for a $(s, a) \in \mathcal{S} \times \mathcal{A}$ but ignore the (s, a) dependence for notation simplicity. Following the roadmap in Section 3.4, we rewrite the algorithm as

$$Z_{n+1,1} = Z_{n,1} + \zeta_1(n)[f_1(Z_{n,1}, Z_{n,2}, \eta_n, Q_n) + M_{n+1}^{(1)}], \tag{26}$$

$$Z_{n+1,2} = Z_{n,2} + \zeta_1(n)[f_2(Z_{n,1}, Z_{n,2}, \eta_n, Q_n) + M_{n+1}^{(2)}], \tag{27}$$

$$\eta_{n+1} = \Gamma_\eta\left[\eta_n + \zeta_2(n)f_3(Z_{n,1}, Z_{n,2}, \eta_n, Q_n)\right], \tag{28}$$

$$Q_{n+1} = \Gamma_Q[Q_n + \zeta_3(n)[f_4(Z_{n,1}, Z_{n,2}, \eta_n, Q_n)]]. \tag{29}$$

Here for theoretical analysis, we add a clipping operator $\Gamma_\eta(x) = \min(\max(x, 0), \overline{\eta})$ and $\Gamma_Q(x) = \min(\max(x, 0), M)$ compared with the algorithm presented in the main text.

We first proceed by first identifying the terms in Equation 26 and 27 and studying the corresponding ODEs

$$\begin{aligned}
\dot{Q}(t) &= 0, \\
\dot{\eta}(t) &= 0, \\
\dot{Z}_1(t) &= f_1(Z_1(t), Z_2(t), \eta(t), Q(t)). \\
\dot{Z}_2(t) &= f_2(Z_1(t), Z_2(t), \eta(t), Q(t)).
\end{aligned}$$

As f_1 and f_2 is in fact irrelavant to the Z_2 and Z_1, we analyze their equilibria seperately. For notation convenience, we denote $y_n(s) = \max_{a'\in\mathcal{A}} Q_n(s, a')$.

For ODE 26 and each $\eta_n \in \mathbb{R}, Q_n \in \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, it is easy to know there exists a unique global asymtotically stable equilibrium $Z_{n,1}^\star = \lambda_1(\eta_n, y_n) = \mathbb{E}[(\eta_n - y_n)_+^{k_*}]$. Similarly, For ODE 27 and each $\eta_n \in \mathbb{R}, Q_n \in \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, there exists a unique global asymtotically stable equilibrium $Z_{n,2}^\star = \lambda_2(\eta, y) = \mathbb{E}[(\eta_n - y_n)_+^{k_*-1}]$.

Second, $M_{n+1}^{(1)} = (\eta_n - y_n)_+^{y_*} - \mathbb{E}[(\eta_n - y_n)_+^{y_*}]$ and $M_{n+1}^{(2)} = (\eta_n - y_n)_+^{y_*-1} - \mathbb{E}[(\eta_n - y_n)_+^{y_*-1}]$. Note that for any $(s, a) \in \mathcal{S} \times \mathcal{A}$, $\eta_n(s, a) \leq \overline{\eta}$, $y_n(s') \leq M$ and $M \leq \overline{\eta}$. Thus $|(\eta_n(s, a) - y_n(s'))_+^{y_*}| \leq \overline{\eta}^{y_*}$, which leads to $|M_{n+1}^{(1)}(s, a)| = |(\eta_n(s, a) - y_n(s'))_+^{y_*} - \mathbb{E}[(\eta_n(s, a) - y_n(s'))_+^{y_*}]| \leq \overline{\eta}^{k_*}$.

Since $\|y_n\|_\infty \leq \|Q_n\|_\infty$ and $(x - y)_+^2 \leq x^2 + y^2$ for any x, y, we have,

$$\begin{aligned}
&\mathbb{E}[\|M_{n+1}^{(1)}\|^2|\mathcal{F}_n] \\
&= \mathbb{E}[\|(\eta_n - y_n)_+^{y_*} - \mathbb{E}[(\eta_n - y_n)_+^{y_*}]\|^2|\mathcal{F}_n] \\
&\leq K_1(1 + \|Z_{n,1}\|^2 + \|Z_{n,2}\|^2 + \|Q_n\|^2 + \|\eta_n\|^2),
\end{aligned}$$

where $K_1 = S\overline{\eta}^{2k_*}$. Similarly, we can conclude that $\mathbb{E}[\|M_{n+1}^{(2)}\|^2|\mathcal{F}_n] \leq K_2(1 + \|Z_{n,1}\|^2 + \|Z_{n,2}\|^2 + \|Q_n\|^2 + \|\eta_n\|^2)$ for some $K_2 = S\overline{\eta}^{2(k_*-1)}$.

Next we analyze the second loop.

$$\begin{aligned}\dot{Q}(t) &= 0,\\ \dot{\eta}(t) &= \Gamma_\eta[f_3(\lambda_1(\eta(t), Q(t)), \lambda_2(\eta(t), Q(t)), \eta(t), Q(t))],\end{aligned}$$

where

$$f_3(\lambda_1(\eta, Q), \lambda_2(\eta, Q), \eta, Q) = -c_k(\rho)\lambda_1(\eta, Q)^{\frac{1}{k_*}-1}\lambda_2(\eta, Q) + 1.$$

The global convergence point is $\eta^*(t) = \arg\max_{\eta\in[0,\overline{\eta}]}\{\sigma_k(Q, \eta)\} = \arg\max_{\eta\in\mathbb{R}}\{\sigma_k(Q, \eta)\}$.

Finally we arrive to the outer loop, i.e.,

$$\dot{Q}(t) = \Gamma_Q[f_4(\lambda_1(Q(t)), \lambda_2(Q(t)), \lambda_3(Q(t)), Q(t))].$$

By using the dual form of Cressie-Read Divergence (Lemma 3.1), we know that this is equivilant to

$$\dot{Q}(t) = r + \gamma \inf_{P\in\mathcal{P}} \mathbb{E}_P[\max_{a'} Q(s', a')] - Q(t),$$

for ambiguity set using Cressie-Read of f divergence.

Denote $H(t) = r + \gamma \inf_{P\in\mathcal{P}} \mathbb{E}_P[\max_{a'} Q(s', a')]$ and thus we can rewrite the above ODE as

$$\dot{Q}(t) = H(t) - Q(t).$$

Following , we consider its infity version, i.e., $H^\infty(t) = \lim_{c\to\infty} H(ct)/c$.

$$\dot{Q}(t) = \gamma \inf_{P\in\mathcal{P}} \mathbb{E}_P[\max_{a'} Q(s', a')] - Q(t).$$

This is a contraction by Theorem 3.2 in Iyengar (2005). By the proof in Section 3.2 in Borkar & Meyn (2000), we know the contraction can lead to the global unique equilibrium point in the ode. Thus we finish verifying all the conditions in Section B.3, which can lead to the desired result. □

Algorithm 3 Distributionally Robust Q-learning with Cressie-Read family of f-divergences **with Simulator**

Input: Exploration rate ϵ, Learning rates $\{\zeta_i(n)\}_{i\in[3]}$, Ambiguity set radius $\rho > 0$, parameter $\varepsilon \in (0, 0.5)$
Init: $\widehat{Q}(s,a) = 0, \forall (s,a) \in \mathcal{S} \times \mathcal{A}$
while Not Converge **do**
  for every $(s,a) \in \mathcal{S} \times \mathcal{A}$ **do**
    Sample $N \in \mathbb{N}$ from $P(N = n) = p_n = \varepsilon(1-\varepsilon)^n$.
    Draw 2^{N+1} samples $\{(r_i, s_i')\}_{i\in[2^{N+1}]}$ from the simulator
    Compute $\Delta^r_{N,\rho}$ via

$$\Delta^r_{N,\rho} = \sup_{\eta\in\mathbb{R}} \widehat{\sigma}^r_k([2^{N+1}], \eta) - \frac{1}{2}\sup_{\eta\in\mathbb{R}} \widehat{\sigma}^r_k([2^N], \eta) - \frac{1}{2}\sup_{\eta\in\mathbb{R}} \widehat{\sigma}^r_k([2^N:], \eta),$$

    where

$$\sup_{\eta\in\mathbb{R}} \widehat{\sigma}^r_k(I, \eta) = \sup_{\eta\in\mathbb{R}}\{-c_k(\rho)[\sum_{i\in I}(\eta - r_i)_+^{k_*}/n]^{\frac{1}{k_*}} + \eta\},$$

    and $[2^N] = \{1, 2, 3, \cdots, 2^N\}$ and $[2^N:] = \{2^N, 2^N+1, \cdots, 2^{N+1}\}$.
    Compute $\Delta^q_{N,\rho}(\widehat{Q}_t)$ via

$$\Delta^q_{N,\rho}(\widehat{Q}_t) = \sup_{\eta\in\mathbb{R}} \widehat{\sigma}^q_k(\widehat{Q}_t, [2^{N+1}], \eta) - \frac{1}{2}\sup_{\eta\in\mathbb{R}} \widehat{\sigma}^q_k(\widehat{Q}_t, [2^N], \eta) - \frac{1}{2}\sup_{\eta\in\mathbb{R}} \widehat{\sigma}^q_k(\widehat{Q}_t, [2^N:], \eta),$$

    where

$$\sup_{\eta\in\mathbb{R}} \widehat{\sigma}^q_k(\widehat{Q}_t, I, \eta) = \sup_{\eta\in\mathbb{R}}\{-c_k(\rho)[\sum_{i\in I}(\eta - \max_{a'\in\mathcal{A}} \widehat{Q}_t(s_i', a'))_+^{k_*}/n]^{\frac{1}{k_*}} + \eta\}.$$

    Set $R_\rho(s,a) = r_1 + \frac{\Delta^r_{N,\rho}}{p_N}$.
    Update Q via

$$\widehat{Q}_{t+1}(s,a) = (1-\zeta_t)\widehat{Q}_t(s,a) + \zeta_t \widehat{\mathcal{T}}_\rho(\widehat{Q}_t)(s,a),$$

    where

$$\widehat{\mathcal{T}}_\rho(\widehat{Q}_t)(s,a) = r_1 + \Delta^r_{N,\rho} + \gamma(\max_{a'\in\mathcal{A}} \widehat{Q}_t(s_1, a') + \frac{\Delta^q_{N,\rho}(\widehat{Q}_t)}{p_N}).$$

  end for
  $t = t + 1$
end while