Finite-Sample Analysis of Off-Policy TD-Learning via Generalized Bellman Operators

Anonymous Author(s)
Affiliation
Address
email

Abstract

In temporal difference (TD) learning, off-policy sampling is known to be more practical than on-policy sampling, and by decoupling learning from data collection, it enables data reuse. It is known that policy evaluation (including multi-step off-policy importance sampling) has the interpretation of solving a generalized Bellman equation. In this paper, we derive finite-sample bounds for any general off-policy TD-like stochastic approximation algorithm that solves for the fixed point of this generalized Bellman operator. Our key step is to show that the generalized Bellman operator is simultaneously a contraction mapping with respect to a weighted ℓ_p-norm for each p in $[1, \infty)$, with a common contraction factor.

Off-policy TD-learning is known to suffer from high variance due to the product of importance sampling ratios. A number of algorithms (e.g. $Q^\pi(\lambda)$, Tree-Backup, Retrace, and Q-trace) have been proposed in the literature to address this issue. Our results immediately imply finite-sample bounds of these algorithms. In particular, we provide first-known finite-sample guarantees for $Q^\pi(\lambda)$, Tree-Backup, and Retrace, and improve the best known bounds of Q-trace. Moreover, we show the bias-variance trade-offs in each of these algorithms.

1 Introduction

Reinforcement learning (RL) demonstrated its success in learning effective policies for a variety of decision making problems such as autonomous driving [23], recommender systems [1], and board games [24]. In RL, there is an important sub-problem – called the policy evaluation problem – of estimating the expected long term reward of a given policy.

The policy evaluation problem is usually solved with the temporal difference (TD) learning method [27]. A key ingredient in TD-learning is the policy used to collect samples (called the behavior policy). Ideally we want to generate samples from the target policy whose value function we want to estimate, and this is called on-policy sampling. However, in many cases such on-policy sampling is not possible due to practical reasons, and hence we need to work with historical data that is generated by a possibly different policy (aka off-policy sampling). For example, in high stake applications such as clinic trials [36] and health care [15], it is not practically possible to re-collect data every time we need to evaluate the performance of a given policy.

Although off-policy sampling is more practical than on-policy sampling, it is more challenging to analyze and is known to have high variance [14], which is due to the presence of the product of the importance sampling ratios, and is a fundamental difficulty in off-policy learning. To overcome this difficulty, many variants of off-policy TD-learning algorithms have been proposed in the literature, such as the $Q^\pi(\lambda)$ [16] algorithm, the TB(λ) algorithm [22], the Retrace(λ) algorithm [21], and the Q-trace algorithm [18], etc.

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

Main Contributions. In this work, we establish finite-sample bounds of a general n-step off-policy TD-learning algorithm that also subsumes several algorithms presented in the literature. The key step is to show that such algorithm can be modeled as a Markovian stochastic approximation (SA) algorithm for solving a generalized Bellman equation. We present sufficient conditions under which the generalized Bellman operator is contractive with respect to a weighted ℓ_p-norm for every $p \in [1, \infty)$, with a uniform contraction factor for all p. Our result shows that the sample complexity scales as $\tilde{\mathcal{O}}(\epsilon^{-2})$, where ϵ is the required accuracy. It also involves a factor that depends on the problem parameters, in particular, the generalized importance sampling ratios, and explicitly demonstrates the bias-variance trade-off.

Our result immediately gives finite-sample guarantees for variants of multi-step off-policy TD-learning algorithms including $Q^\pi(\lambda)$, TB(λ), Retrace(λ), and Q-trace. For $Q^\pi(\lambda)$, TB(λ), and Retrace(λ), we establish the first-known results in the literature, while for Q-trace, we improve the best known results in [18]. Based on the finite-sample bounds, we show that all four algorithms overcome the high variance issue in Vanilla off-policy TD-learning, but their convergence rates are all affected to varying degrees.

1.1 Generalized Bellman operator and stochastic approximation

In this section, we illustrate the interpretation of off-policy multi-step TD-learning as an SA algorithm for solving a generalized Bellman equation. Consider the policy evaluation problem where the goal is to estimate the state-action value function Q^π of a given policy π. See Section 1.3 for more details. In the simplest setting where TD(0) with on-policy sampling is employed, it is well known that the algorithm is an SA algorithm for solving the Bellman equation $Q = \mathcal{H}_\pi(Q)$, where $\mathcal{H}_\pi(\cdot)$ is the Bellman operator. The generalized Bellman operator $\mathcal{B}(\cdot)$ is defined by:

$$\mathcal{B}(Q) = \mathcal{T}(\mathcal{H}(Q) - Q) + Q, \tag{1}$$

where $\mathcal{T}(\cdot)$ and $\mathcal{H}(\cdot)$ are two auxiliary operators. In the special case where $\mathcal{T}(\cdot)$ is the identity operator and $\mathcal{H}(\cdot)$ is the Bellman operator $\mathcal{H}_\pi(\cdot)$, the generalized Bellman operator $\mathcal{B}(\cdot)$ reduces to the regular Bellman operator $\mathcal{H}_\pi(\cdot)$. Note that any fixed point of $\mathcal{H}(\cdot)$ is also a fixed point of $\mathcal{B}(\cdot)$, as long as $\mathcal{T}(\cdot)$ is such that $\mathcal{T}(\mathbf{0}) = \mathbf{0}$. Thus, the operator $\mathcal{H}(\cdot)$ controls the fixed-point of the generalized Bellman operator $\mathcal{B}(\cdot)$, and the operator $\mathcal{T}(\cdot)$ can be used to control its contraction properties. To further understand $\mathcal{B}(\cdot)$, we demonstrate in the following that both on-policy n-step TD and TD(λ) can be viewed as SA algorithms for solving the generalized Bellman equation $\mathcal{B}(Q) = Q$.

On-policy n-step TD is designed to solve the n-step Bellman equation $Q = \mathcal{H}_\pi^n(Q)$. Since the Bellman operator $\mathcal{H}_\pi(\cdot)$ is explicitly given by $\mathcal{H}_\pi(\cdot) = R + \gamma P_\pi(\cdot)$, where R is the reward vector, γ is the discount factor, and P_π is the transition probability matrix under π, the equation $Q = \mathcal{H}_\pi^n(Q)$ can be written as $Q = \sum_{i=0}^{n-1}(\gamma P_\pi)^i R + (\gamma P_\pi)^n Q$, which by reverse telescoping is equivalent to

$$Q = \textstyle\sum_{i=0}^{n-1}(\gamma P_\pi)^i (R + \gamma P_\pi Q - Q) + Q = \mathcal{T}(\mathcal{H}_\pi(Q) - Q) + Q,$$

where $\mathcal{T}(Q) = \sum_{i=0}^{n-1}(\gamma P_\pi)^i Q$. Similarly, one can formulate the TD(λ) Bellman equation in the form of $\mathcal{B}(Q) = Q$, where $\mathcal{T}(Q) = (1-\lambda)\sum_{i=0}^{\infty}\lambda^i \sum_{j=0}^{i-1}(\gamma P_\pi)^i Q$ and $\mathcal{H}(\cdot) = \mathcal{H}_\pi(\cdot)$.

In these examples, the operator $\mathcal{T}(\cdot)$ captures the impact of bootstrapping. In this work, we show that in addition to on-policy TD-learning, variants of off-policy TD-learning with multi-step bootstrapping can also be interpreted as SA algorithms for solving the generalized Bellman equation. Moreover, under some mild conditions, we show that the generalized Bellman operator $\mathcal{B}(\cdot)$ is a contraction mapping with respect to some weighted ℓ_p-norm for any $p \in [1, \infty)$, with a common contraction factor. This enables us to establish finite-sample bounds of generic multi-step off-policy TD-like algorithms.

1.2 Related literature

The TD-learning method was first proposed in [27] for solving the policy evaluation problem. Since then, there is an increasing interest in theoretically understanding TD-learning and its variants.

On-Policy TD. The most basic TD-learning method is the TD(0) algorithm [27]. Later it was extended to using multi-step bootstrapping (i.e., the n-step TD-learning algorithm [10, 34, 35]), and using eligibility trace (i.e., the TD(λ) algorithm [25, 27]). The asymptotic convergence of TD-learning was established in [12, 17, 32]. As for finite-sample analysis, a unified Lyapunov approach is

presented in [9]. To overcome the curse of dimensionality in RL, TD-learning is usually incorporated with function approximation in practice. In the basic setting where a linear parametric architecture is used, the asymptotic convergence of TD-learning was established in [33], and finite-sample bounds in [4, 11, 26, 31]. Very recently, the convergence and finite-sample guarantee of TD-learning with neural network approximation were studied in [6, 7].

Off-Policy TD. In the off-policy setting, since the samples are not necessarily generated by the target policy, usually importance sampling ratios (or "generalized" importance sampling ratios) are introduced in the TD-learning algorithm. The resulting algorithms are $Q^\pi(\lambda)$ [22], TB(λ) [16], Retrace(λ) [21], and Q-trace [18] (which is an extension of V-trace [13]), etc. The asymptotic convergence of these algorithms has been established in the papers in which they were proposed. To the best of our knowledge, finite-sample guarantees are established only for Q-trace and V-trace [8, 9, 18]. In the function approximation setting, TD-learning with off-policy sampling and function approximation is a typical example of the deadly triad [28]. In the presence of the deadly triad, the algorithm can be unstable [2, 28]. To achieve convergence, one needs to significantly modify the original TD-learning algorithm, resulting in two time-scale algorithms such as GTD [20], TDC [29], and emphatic TD [30], etc.

1.3 Preliminaries

In this section, we cover the background of RL and the TD-learning method for solving the policy evaluation problem. The RL problem is usually modeled as a Markov decision process (MDP). In this work, we consider an MDP with a finite set of states $\mathcal{S}$, a finite set of actions $\mathcal{A}$, a set of unknown action dependent transition probability matrices $\mathcal{P} = \{P_a \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|} \mid a \in \mathcal{A}\}$, an unknown reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \mapsto [0, 1]$, and a discount factor $\gamma \in (0, 1)$. In order for an MDP to progress, we must specify the policy of selecting actions based on the state of the environment. Specifically, a policy π is a mapping from the state-space to probability distributions supported on the action space, i.e., $\pi : \mathcal{S} \mapsto \Delta^{|\mathcal{A}|}$. The state-action value function Q^π associated with a policy π is defined by $Q^\pi(s, a) = \mathbb{E}_\pi[\sum_{k=0}^{\infty} \gamma^k \mathcal{R}(S_k, A_k) \mid S_0 = s, A_0 = a]$ for all (s, a). The goal in policy evaluation is to estimate the state-action value function Q^π for a given policy π.

Since the transition probabilities as well as the reward function are unknown, such state-action value function cannot be directly computed. The TD-learning algorithm is designed to estimate Q^π using the SA method. Specifically, in TD-learning, we first collect a sequence of samples $\{(S_k, A_k)\}$ from the model using some behavior policy π_b. Then the value function Q^π is iteratively estimated using the samples $\{(S_k, A_k)\}$. When $\pi_b = \pi$, the algorithm is called on-policy TD-learning, otherwise it is called off-policy TD-learning.

2 Finite-sample analysis of general off-policy TD-learning

In this section, we present our unified framework for finite-sample analysis of off-policy TD-learning algorithms using generalized importance sampling ratios and multi-step bootstrapping.

2.1 A generic model for multi-step off-policy TD-learning

Algorithm 1 presents our generic algorithm model. Due to off-policy sampling, the two functions $c, \rho : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}_+$ are introduced in Algorithm 1 to serve as generalized importance sampling ratios in order to account for the discrepancy between the target policy π and the behavior policy π_b. We denote $c_{\max} = \max_{s,a} c(s, a)$ and $\rho_{\max} = \max_{s,a} \rho(s, a)$. We next show how Algorithm 1 captures variants of off-policy TD-learning algorithms in the literature by using different generalized importance sampling ratios $c(\cdot, \cdot)$ and $\rho(\cdot, \cdot)$.

Vanilla IS. When $c(s, a) = \rho(s, a) = \pi(a|s)/\pi_b(a|s)$ for all (s, a), Algorithm 1 is the standard off-policy TD-learning with importance sampling [22]. We will refer to this algorithm as Vanilla IS. Although Vanilla IS was shown to converge to Q^π [22], since the product of importance sampling ratios $\prod_{j=k+1}^{i} \frac{\pi(A_j|S_j)}{\pi_b(A_j|S_j)}$ is not controlled in any way, it suffers the most from high variance.

The $Q^\pi(\lambda)$ Algorithm. When $c(s, a) = \lambda$ and $\rho(s, a) = \pi(a|s)/\pi_b(a|s)$, Algorithm 1 is the $Q^\pi(\lambda)$ algorithm [16]. The $Q^\pi(\lambda)$ algorithm overcomes the high variance issue in Vanilla IS by introducing the parameter λ. However, the algorithm converges only when λ is sufficiently small [21].

Algorithm 1 A Generic Algorithm for Multi-Step Off-Policy TD-Learning

Input: K, $\{\alpha_k\}$, Q_0, π, π_b, generalized importance sampling ratios $c, \rho : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}_+$, and sample trajectory $\{(S_k, A_k)\}_{0 \le k \le K+n}$ collected under the behavior policy π_b.
for $k = 0, 1, \cdots, K-1$ **do**
  $\alpha_k(s,a) = \alpha_k \mathbb{I}_{\{(s,a)=(S_k,A_k)\}}$ for all (s,a) ▷ Asynchronous update
  $\Delta(S_i, A_i, S_{i+1}, A_{i+1}, Q_k) = \mathcal{R}(S_i, A_i) + \gamma\rho(S_{i+1}, A_{i+1})Q_k(S_{i+1}, A_{i+1}) - Q_k(S_i, A_i)$ for all $i \in \{k, k+1, ..., k+n-1\}$. ▷ Δ is called the one-step temporal difference
  $Q_{k+1}(s,a) = Q_k(s,a) + \alpha_k(s,a) \sum_{i=k}^{k+n-1} \gamma^{i-k} \prod_{j=k+1}^{i} c(S_j, A_j) \Delta(S_i, A_i, S_{i+1}, A_{i+1}, Q_k)$ for all (s,a) ▷ The update equation
end for
Output: Q_K

The TB(λ) Algorithm. When $c(s,a) = \lambda\pi(a|s)$ and $\rho(s,a) = \pi(a|s)/\pi_b(a|s)$, we have the TB($\lambda$) algorithm [22]. The TB(λ) algorithm also overcomes the high variance issue in Vanilla IS and is guaranteed to converge to Q^π without needing any strong assumptions. However, as discussed in [21], the TB(λ) algorithm lacks sample efficiency as it does not effectively use the multi-step return.

The Retrace(λ) Algorithm. When $c(s,a) = \lambda \min(1, \pi(a|s)/\pi_b(a|s))$ and $\rho(s,a) = \pi(a|s)/\pi_b(a|s)$, we have the Retrace($\lambda$) algorithm, which overcomes the high variance and converges to Q^π. The convergence rate of Retrace(λ) is empirically observed to be better than TB(λ) in [21].

The Q-trace Algorithm. When we choose $c(s,a) = \min(\bar{c}, \pi(a|s)/\pi_b(a|s))$ and $\rho(s,a) = \min(\bar{\rho}, \pi(a|s)/\pi_b(a|s))$, where $\bar{\rho} \ge \bar{c}$, Algorithm 1 is the Q-trace algorithm [18]. The Q-trace algorithm is an analog of the V-trace algorithm [13] in that Q-trace estimates the Q-function instead of the V-function. The two truncation levels $\bar{c}$ and $\bar{\rho}$ in these algorithms separately control the variance and the asymptotic bias in the algorithm respectively. Note that due to the truncation level $\bar{\rho}$, the algorithm no longer converges to Q^π, but to a biased limit point, denoted by $Q^{\pi,\rho}$ [18].

From now on, we focus on studying the generic algorithm 1. To do that, we make the following assumption about the behavior policy π_b, which is fairly standard in off-policy TD-learning.

Assumption 2.1. The behavior policy π_b satisfies $\pi_b(a|s) > 0$ for all (s,a). In addition, the Markov chain $\{S_k\}$ induced by the behavior policy π_b is irreducible and aperiodic.

Irreducibility and aperiodicity together imply that the Markov chain $\{S_k\}$ has a unique stationary distribution, which we denote by $\kappa_S \in \Delta^{|\mathcal{S}|}$. Moreover, the Markov chain $\{S_k\}$ mixes geometrically fast in that there exist $C > 0$ and $\sigma \in (0,1)$ such that $\max_{s\in\mathcal{S}} \|P^k(s,\cdot) - \kappa_S(\cdot)\|_{\text{TV}} \le C\sigma^k$ for all $k \ge 0$, where $\|\cdot\|_{\text{TV}}$ is the total variation distance [19]. Let $\kappa_{SA} \in \Delta^{|\mathcal{S}||\mathcal{A}|}$ be such that $\kappa_{SA}(s,a) = \kappa_S(s)\pi_b(a|s)$ for all (s,a). Note that κ_{SA} is the stationary distribution of the Markov chain $\{(S_k, A_k)\}$. Let $\mathcal{K}_S = \text{diag}(\kappa_S) \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}$ and $\mathcal{K}_{SA} = \text{diag}(\kappa_{SA}) \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|\times|\mathcal{S}||\mathcal{A}|}$, and denote the minimal diagonal entries of $\mathcal{K}_S$ and $\mathcal{K}_{SA}$ by $\mathcal{K}_{S,\min}$ and $\mathcal{K}_{SA,\min}$ respectively.

2.2 Identifying the generalized Bellman operator

In this section, we identify the generalized Bellman equation which Algorithm 1 is trying to solve, and also the corresponding generalized Bellman operator and its asynchronous variant. Let $\mathcal{T}_c, \mathcal{H}_\rho : \mathbb{R}^{|\mathcal{S}||\mathcal{A}|} \mapsto \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$ be two operators defined by

$$[\mathcal{T}_c(Q)](s,a) = \sum_{i=0}^{n-1} \gamma^i \mathbb{E}_{\pi_b}\left[\prod_{j=1}^{i} c(S_j, A_j) Q(S_i, A_i) \;\middle|\; S_0 = s, A_0 = a\right], \text{ and}$$
$$[\mathcal{H}_\rho(Q)](s,a) = \mathcal{R}(s,a) + \gamma\mathbb{E}_{\pi_b}[\rho(S_{k+1}, A_{k+1})Q(S_{k+1}, A_{k+1}) \mid S_k = s, A_k = a]$$

for all (s,a). Note that the operator $\mathcal{T}_c(\cdot)$ depends on the generalized importance sampling ratio $c(\cdot,\cdot)$, while the operator $\mathcal{H}_\rho(\cdot)$ depends on the generalized importance sampling ratio $\rho(\cdot,\cdot)$.

With $\mathcal{T}_c(\cdot)$ and $\mathcal{H}_\rho(\cdot)$ defined above, Algorithm 1 can be viewed as an asynchronous SA algorithm for solving the generalized Bellman equation $\mathcal{B}_{c,\rho}(Q) = Q$, where the generalized Bellman operator $\mathcal{B}_{c,\rho}(\cdot)$ is defined by $\mathcal{B}_{c,\rho}(Q) = \mathcal{T}_c(\mathcal{H}_\rho(Q) - Q) + Q$. Since Algorithm 1 performs asynchronous update, using the terminology in [9], we further define the asynchronous variant $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$ of the generalized Bellman operator $\mathcal{B}_{c,\rho}(\cdot)$ by

$$\tilde{\mathcal{B}}_{c,\rho}(Q) := \mathcal{K}_{SA}\mathcal{B}_{c,\rho}(Q) + (I - \mathcal{K}_{SA})Q = \mathcal{K}_{SA}\mathcal{T}_c(\mathcal{H}_\rho(Q) - Q) + Q. \tag{2}$$

Each component of the asynchronous generalized Bellman operator $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$ can be thought of as a convex combination with identity, where the weights are the stationary probabilities of visiting state-action pairs $(s,a) \in \mathcal{S} \times \mathcal{A}$. This captures the fact that when performing asynchronous update, the corresponding component is updated only when the state-action pair (s,a) is visited. It is clear from its definition that $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$ has the same fixed-points as $\mathcal{B}_{c,\rho}(\cdot)$ (provided that they exist).

Under some mild conditions on the generalized importance sampling ratios $c(\cdot,\cdot)$ and $\rho(\cdot,\cdot)$, we will show in the next section that both the asynchronous generalized Bellman operator $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$ and the operator $\mathcal{H}_\rho(\cdot)$ are contraction mappings. Therefore, since $\mathcal{T}_c(\mathbf{0}) = \mathbf{0}$, the operators $\mathcal{H}_\rho(\cdot)$, $\mathcal{B}_{c,\rho}(\cdot)$, $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$ all share the same unique fixed-point. Since the fixed-point of the operator $\mathcal{H}_\rho(\cdot)$ depends only on the generalized importance sampling ratio $\rho(\cdot,\cdot)$, but not on $c(\cdot,\cdot)$, we can flexibly choose $c(\cdot,\cdot)$ to control the variance while maintaining the fixed-point of the operators $\mathcal{B}_{c,\rho}(\cdot)$ and $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$. As we will see later, this is the key property used in designing variants of variance reduced n-step off-policy TD-learning algorithms such as $Q^\pi(\lambda)$, TB(λ), and Retrace(λ).

2.3 Establishing the contraction property

In this section, we study the fixed-point and the contraction property of the asynchronous generalized Bellman operator $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$. We begin by introducing some notation. Let $D_c, D_\rho \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|\times|\mathcal{S}||\mathcal{A}|}$ be two diagonal matrices such that $D_c((s,a),(s,a)) = \sum_{a\in\mathcal{A}} \pi_b(a|s)c(s,a)$ and $D_\rho((s,a),(s,a)) = \sum_{a\in\mathcal{A}} \pi_b(a|s)\rho(s,a)$ for all (s,a). We denote $D_{c,\min}$ ($D_{c,\max}$) and $D_{\rho,\min}$ ($D_{\rho,\max}$) as the minimal (maximal) diagonal entries of the matrices D_c and D_ρ respectively.

In view of the definition of $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$ in Eq. (2), the fixed-point of $\mathcal{H}_\rho(\cdot)$ must also be a fixed-point of $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$. We first study the fixed point of $\mathcal{H}_\rho(\cdot)$ by establishing its contraction property.

Proposition 2.1. *Suppose that $D_{\rho,\max} < 1/\gamma$. Then the operator $\mathcal{H}_\rho(\cdot)$ is a contraction mapping with respect to the ℓ_∞-norm, with contraction factor $\gamma D_{\rho,\max}$. In this case, the unique fixed-point $Q^{\pi,\rho}$ of $\mathcal{H}_\rho(\cdot)$ satisfies the following inequality: $\|Q^\pi - Q^{\pi,\rho}\|_\infty \leq \frac{\gamma \max_{s,a} |\pi(a|s)-\pi_b(a|s)\rho(s,a)|}{(1-\gamma)(1-\gamma D_{\rho,\max})}$.*

Observe from Proposition 2.1 that when $\rho(s,a) = \pi(a|s)/\pi_b(a|s)$, which is the case for $Q^\pi(\lambda)$, TB(λ), and Retrace(λ), the unique fixed-point $Q^{\pi,\rho}$ is exactly the target value function Q^π. This agrees with the definition of the operator $\mathcal{H}_\rho(\cdot)$ in that it reduces to the regular Bellman operator $\mathcal{H}_\pi(\cdot)$ when $\rho(s,a) = \pi(a|s)/\pi_b(a|s)$ for all (s,a). If $\rho(s,a) \neq \pi(a|s)/\pi_b(a|s)$ for some (s,a), then in general the fixed-point of $\mathcal{H}_\rho(\cdot)$ is different from Q^π. See Appendix A.2 for more details. In that case, Proposition 2.1 provides an error bound on the difference between the potentially biased limit $Q^{\pi,\rho}$ and Q^π. Such error bound will be useful for us to study the Q-trace algorithm.

To further guarantee the uniqueness of the fixed-point of $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$, we establish the contraction property of $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$. We begin with the following definition.

Definition 2.1. Let $\{\mu_i\}_{1\leq i\leq d}$ be such that $\mu_i > 0$ for all i. Then for any $x \in \mathbb{R}^d$, the weighted ℓ_p-norm ($p \in [1,\infty)$) of x with weights $\{\mu_i\}$ is defined by $\|x\|_{\mu,p} = (\sum_i \mu_i |x_i|^p)^{1/p}$.

We next establish the contraction property of the operator $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$ in the following theorem. Let $\omega = \mathcal{K}_{SA,\min} f(\gamma D_{c,\min})(1-\gamma D_{\rho,\max})$, where the function $f : \mathbb{R} \mapsto \mathbb{R}$ is defined by $f(x) = n$ when $x = 1$, and $f(x) = \frac{1-x^n}{1-x}$ when $x \neq 1$.

Theorem 2.1. *Suppose that $c(s,a) \leq \rho(s,a)$ for all (s,a) and $D_{\rho,\max} < 1/\gamma$. Then for any $\theta \in (0,1)$, there exists a weight vector $\mu \in \Delta^{|\mathcal{S}||\mathcal{A}|}$ satisfying $\mu(s,a) \geq \frac{\omega(1-\theta)}{(1-\theta\omega)|\mathcal{S}||\mathcal{A}|}$ for all (s,a) such that the operator $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$ is a contraction mapping with respect to $\|\cdot\|_{\mu,p}$ for any $p \in [1,\infty)$, with a common contraction factor $1-\theta\omega$.*

Theorem 2.1 is the key result for our finite-sample analysis, and we present its proof in the next section. The weighted ℓ_p-norm (especially ℓ_2-norm) contraction property we established for the operator $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$ has a far-reaching impact even beyond the finite-sample analysis of tabular RL in this paper. First of all, it automatically implies that the matrix A is Hurwitz (see Appendix A.6). Second, note that the key property used for establishing the convergence of on-policy TD-learning with linear function approximation in the seminal work [33] is that the Bellman operator is a contraction mapping not only with respect to the ℓ_∞-norm, but also with respect to a weighted ℓ_2-norm. We establish the

same property in the off-policy setting, and hence lay down the foundation for extending our results to the function approximation setting. This is an immediate future research direction.

2.4 Proof of Theorem 2.1

We begin by explicitly computing the asynchronous generalized Bellman operator $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$. Let π_c and π_ρ be two policies defined by $\pi_c(a|s) = \frac{\pi_b(a|s)c(s,a)}{D_c((s,a),(s,a))}$ and $\pi_\rho(a|s) = \frac{\pi_b(a|s)\rho(s,a)}{D_\rho((s,a),(s,a))}$ for all (s,a).

Proposition 2.2. *The operator $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$ is explicitly given by $\tilde{\mathcal{B}}_{c,\rho}(Q) = AQ + b$ for any Q, where $A = I - \mathcal{K}_{SA}\sum_{i=0}^{n-1}(\gamma P_{\pi_c}D_c)^i(I - \gamma P_{\pi_\rho}D_\rho)$ and $b = \mathcal{K}_{SA}\sum_{i=0}^{n-1}(\gamma P_{\pi_c}D_c)^i R$.*

In light of Proposition 2.2, to prove Theorem 2.1, it is enough to work with the matrix A. To proceed, we require the following definition.

Definition 2.2. Given $\beta \in [0,1]$, a matrix $M \in \mathbb{R}^{d\times d}$ is called a substochastic matrix with modulus β if and only if $M_{ij} \geq 0$ for all i,j and $\sum_j M_{ij} \leq 1-\beta$ for all i.

We next show in the following two propositions that (1) the matrix A given in Proposition 2.2 is a sub-stochastic matrix with modulus ω, and (2) for any sub-stochastic matrix M with a positive modulus, there exist weights $\{\mu_i\}$ such that the induced matrix norm $\|M\|_{\mu,p}$ is strictly less than 1. These two results together imply Theorem 2.1.

Proposition 2.3. *Suppose that $c(s,a) \leq \rho(s,a)$ for all (s,a) and $D_{\rho,\max} < 1/\gamma$. Then the matrix A given in Proposition 2.2 is a sub-stochastic matrix with modulus ω, where $\omega = \mathcal{K}_{SA,\min}f(\gamma D_{c,\min})(1-\gamma D_{\rho,\max})$.*

The condition $c(s,a) \leq \rho(s,a)$ ensures that the matrix A is non-negative, and the condition $D_{\rho,\max} < 1/\gamma$ ensures that the each row of the matrix A sums up to at most $1-\omega$. Together they imply the substochasticity of A. The modulus ω is an important parameter for our finite-sample analysis. In view of Theorem 2.1, we see that large modulus gives smaller (or better) contraction factor of $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$.

Proposition 2.4. *For any sub-stochastic matrix $M \in \mathbb{R}^{d\times d}$ with a positive modulus $\beta \in (0,1)$, for any $\theta \in (0,1)$, there exists a weight vector $\mu \in \Delta^d$ satisfying $\mu_i \geq \frac{\beta(1-\theta)}{(1-\theta\beta)d}$ for all i such that $\|M\|_{\mu,p} \leq 1-\theta\beta$ for any $p \in [1,\infty)$. Furthermore, if M is irreducible [1], then we can choose $\theta = 1$.*

Note that Proposition 2.4 introduces the tunable parameter θ. It is clear that large θ gives better contraction factor of $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$ but worse lower bound on the entries of the weight vector μ. Since Proposition 2.4 is the major result for proving Theorem 2.1, we provide its proof sketch in Section 4.

2.5 Finite-sample convergence guarantees

In light of Theorem 2.1, Algorithm 1 is a Markovian SA algorithm for solving a fixed-point equation $\tilde{\mathcal{B}}_{c,\rho}(Q) = Q$, where the fixed-point operator $\tilde{\mathcal{B}}_{c,\rho}(\cdot)$ is a contraction mapping. Therefore, to establish the finite-sample bounds, we use a Lyapunov drift argument where we choose $W(Q) = \|Q - Q^{\pi,\rho}\|_{\mu,p}^2$ as the Lyapunov function. This leads to a finite-sample bound on $\mathbb{E}[\|Q_k - Q^{\pi,\rho}\|_{\mu,p}^2]$. However, since μ is unknown, to make the finite-sample bound independent of μ, we use the lower bound on $\mu(s,a)$ provided in Theorem 2.1 and also tune the parameters p and θ to obtain a finite-sample bound on $\mathbb{E}[\|Q_k - Q^{\pi,\rho}\|_\infty^2]$. The fact that the contraction factor $1-\theta\omega$ (cf. Theorem 2.1) is independent of p plays an important role in such tuning process.

To present the results, we need to introduce more notation. For any $\delta > 0$, define $t_\delta(\mathcal{MC}_S)$ as the mixing time of the Markov chain $\{S_k\}$ (induced by π_b) with precision δ, i.e., $t_\delta(\mathcal{MC}_S) = \min\{k \geq 0 : \max_{s\in\mathcal{S}}\|P^k(s,\cdot) - \kappa_S(\cdot)\|_{\text{TV}} \leq \delta\}$. Under Assumption 2.1, we can easily verify that $t_\delta(\mathcal{MC}_S) \leq L(\log(1/\delta)+1)$ for some constant $L > 0$. Let $\tau_{\delta,n} = t_\delta(\mathcal{MC}_S) + n + 1$. The parameters c_1, c_2 and c_3 used in presenting the following theorem are numerical constants.

Theorem 2.2. *Consider $\{Q_k\}$ of Algorithm 1. Suppose that (1) Assumptions 2.1 is satisfied, (2) $c(s,a) \leq \rho(s,a)$ for all (s,a) and $D_{\rho,\max} < 1/\gamma$, and (3) α is chosen such that $\alpha\tau_{\alpha,n} \leq \frac{c_1\omega}{\log(2|\mathcal{S}||\mathcal{A}|/\omega)f(\gamma c_{\max})^2(\gamma\rho_{\max}+1)^2}$. Then we have the following finite-sample convergence bound:*

$$\mathbb{E}[\|Q_k - Q^{\pi,\rho}\|_\infty^2] \leq \zeta_1\left(1-\frac{\omega\alpha}{2}\right)^{k-\tau_{\alpha,n}} + \zeta_2\frac{f(\gamma c_{\max})^2(\gamma\rho_{\max}+1)^2\log(2|\mathcal{S}||\mathcal{A}|/\omega)}{\omega}\alpha\tau_{\alpha,n}, \quad (3)$$

where $\zeta_1 = c_2(\|Q_0 - Q^{\pi,\rho}\|_\infty + \|Q_0\|_\infty + 1)^2$, and $\zeta_2 = c_3(3\|Q^{\pi,\rho}\|_\infty + 1)^2$.

[1] A non-negative matrix is irreducible if and only if its associated graph is strongly connected [3].

Theorem 2.2 enables one to design a wide class of off-policy TD variants with provable finite-sample guarantees by choosing appropriate generalized importance sampling ratios $c(\cdot,\cdot)$ and $\rho(\cdot,\cdot)$, which, as we will see soon, are closely related to the bias-variance trade-off in Algorithm 1. The first term on the RHS of Eq. (3) is usually called the convergence bias in SA literature [5], and it goes to zero at a geometric rate. The second term on the RHS of Eq. (3) stands for the variance in the iterates, and it is a constant proportional to $\alpha\tau_{\alpha,n}$. To see more explicitly the bias-variance trade-off, we derive the sample complexity of Algorithm 1 in the following.

Corollary 2.2.1. *For an accuracy $\epsilon > 0$, to obtain $\mathbb{E}[\|Q_k - Q^{\pi,\rho}\|_\infty] \leq \epsilon$, the sample complexity is*

$$\underbrace{\mathcal{O}\left(\frac{\log^2(1/\epsilon)}{\epsilon^2}\right)}_{T_1}\underbrace{\tilde{\mathcal{O}}\left(\frac{n}{(1-\gamma)^2}\right)}_{T_2}\underbrace{\tilde{\mathcal{O}}\left(\frac{f(\gamma c_{\max})^2(\gamma\rho_{\max}+1)^2}{\mathcal{K}_{SA}^2 f(\gamma D_{c,\min})^2(1-\gamma D_{\rho,\max})^2}\right)}_{T_3}. \tag{4}$$

From Corollary 2.2.1, we see that the dependency on the accuracy is $\tilde{\mathcal{O}}(\epsilon^{-2})$, and the dependency on the parameter n and the effective horizon $1/(1-\gamma)$ is $\tilde{\mathcal{O}}(n/(1-\gamma)^2)$, both of which are the same as TD-learning in the on-policy setting [9]. The impact of performing off-policy sampling is captured by the term T_3, which depends on the choice of the generalized importance sampling ratios $c(\cdot,\cdot)$ and $\rho(\cdot,\cdot)$.

In the numerator of T_3, we have $f(\gamma c_{\max})^2(\gamma\rho_{\max}+1)^2$, which is from the second term on the RHS of Eq. (3), and represents the impact of the variance on the sample complexity. It is clear that smaller $c_{\max}$ and $\rho_{\max}$ lead to smaller variance. As we will see later, this is the reason for the variance reduction of various off-policy TD-learning algorithms in the literature. In the denominator of T_3, we have $\mathcal{K}_{SA,\min}^2 f(\gamma D_{c,\min})^2(1-\gamma D_{\rho,\max})^2 = \omega^2$, which represents the effect of the contraction factor. To see this, recall from Theorem 2.1 that the contraction factor is $1-\theta\omega$. In light of the previous analysis, the bias-variance trade-off in general off-policy multi-step TD-learning algorithm 1 is intuitively of the form $\tilde{\mathcal{O}}\left(\frac{\text{Variance}}{(1-\text{ Contraction factor})^2}\right)$.

3 Application to various off-policy TD-learning algorithms

In this section, we apply Theorem 2.2 to various off-policy n-step TD-learning algorithms in the literature (e.g. $Q^\pi(\lambda)$, TB(λ), Retrace(λ), and Q-trace). We begin by introducing some notation. Let $\pi_{\max}$ ($\pi_{\min}$) and $\pi_{b,\max}$ ($\pi_{b,\min}$) be the maximal (minimal) entry of the target policy π and the behavior policy π_b respectively. Let $r_{\max} = \max_{s,a}(\pi(a|s)/\pi_b(a|s))$ ($r_{\min} = \min_{s,a}(\pi(a|s)/\pi_b(a|s))$) be the maximum (minimum) ratio between π and π_b. We will overload the notation of ζ_1 and ζ_2 from Theorem 2.2. Note that $Q^{\pi,\rho} = Q^\pi$ in $Q^\pi(\lambda)$, TB(λ), and Retrace(λ), but $Q^{\pi,\rho} \neq Q^\pi$ in Q-trace.

3.1 Finite-sample analysis of Vanilla IS

Consider the Vanilla IS algorithm, where $c(s,a) = \rho(s,a) = \pi(a|s)/\pi_b(a|s)$ for all (s,a). In this case, we have $c_{\max} = \rho_{\max} = r_{\max}$, $D_c = D_\rho = I$, and $\omega = \mathcal{K}_{SA,\min}(1-\gamma^n)$.

Theorem 3.1. *Consider Algorithm 1 with Vanilla IS update. Under Assumption 2.1, suppose α is chosen such that $\alpha\tau_{\alpha,n} \leq \frac{c_1\omega}{\log(2|\mathcal{S}||\mathcal{A}|/\omega)f(\gamma r_{\max})^2(\gamma r_{\max}+1)^2}$. Then we have for all $k \geq \tau_{\alpha,n}$: $\mathbb{E}[\|Q_k - Q^\pi\|_\infty^2] \leq \zeta_1\left(1-\frac{\omega\alpha}{2}\right)^{k-\tau_{\alpha,n}} + \zeta_2\frac{f(\gamma r_{\max})^2(\gamma r_{\max}+1)^2\log(2|\mathcal{S}||\mathcal{A}|/\omega)}{\omega}\alpha\tau_{\alpha,n}$, where $\omega = \mathcal{K}_{SA,\min}(1-\gamma^n)$. This leads to a sample complexity of $\tilde{\mathcal{O}}\left(\frac{\log^2(1/\epsilon)}{\epsilon^2}\frac{nf(\gamma r_{\max})^2(\gamma r_{\max}+1)^2}{\mathcal{K}_{SA}^2(1-\gamma^n)^2(1-\gamma)^2}\right)$.*

Note that the factor $f^2(\gamma r_{\max}) = (\sum_{i=0}^{n-1}(\gamma r_{\max})^i)^2$ appears on the variance term. When $\gamma r_{\max} > 1$ (which can usually happen), the variance term involves an exponential factor $(\gamma r_{\max})^n$. The reason is that the product of importance sampling ratios are not at all controlled by any means in Vanilla IS. Therefore, the variance can be very large. On the other hand, since the importance sampling ratios are not modified, Vanilla IS effectively uses the full n-step return. As a result, the parameter $\omega = \mathcal{K}_{SA,\min}(1-\gamma^n)$ within Vanilla IS is the largest (best) among all the algorithms we study.

3.1.1 Finite-sample analysis of $Q^\pi(\lambda)$

Consider the $Q^\pi(\lambda)$ algorithm, where $c(s,a) = \lambda$ and $\rho(s,a) = \pi(a|s)/\pi_b(a|s)$ for all (s,a). This implies that $c_{\max} = \lambda$, $\rho_{\max} = r_{\max}$, $D_c = \lambda I$, and $D_\rho = I$.

Theorem 3.2. *Consider Algorithm 1 with $Q^\pi(\lambda)$ update. Suppose that Assumption 2.1 is satisfied, $\lambda \le r_{\min}$, and α is chosen such that $\alpha\tau_{\alpha,n} \le \frac{c_1\omega}{\log(2|\mathcal{S}||\mathcal{A}|/\omega)f(\gamma\lambda)^2(\gamma r_{\max}+1)^2}$. Then we have for all $k \ge \tau_{\alpha,n}$: $\mathbb{E}[\|Q_k - Q^\pi\|_\infty^2] \le \zeta_1\left(1-\frac{\omega\alpha}{2}\right)^{k-\tau_{\alpha,n}} + \zeta_2\frac{f(\gamma\lambda)^2(\gamma r_{\max}+1)^2\log(2|\mathcal{S}||\mathcal{A}|/\omega)}{\omega}\alpha\tau_{\alpha,n}$, where $\omega = \mathcal{K}_{SA,\min}f(\gamma\lambda)(1-\gamma)$. This leads to a sample complexity of $\tilde{\mathcal{O}}\left(\frac{\log^2(1/\epsilon)}{\epsilon^2}\frac{n(\gamma r_{\max}+1)^2}{\mathcal{K}_{SA}^2(1-\gamma)^4}\right)$.*

To see how $Q^\pi(\lambda)$ overcomes the high variance issue in Vanilla IS. Note that since $\gamma\lambda \le \gamma r_{\min} \le \gamma < 1$, we have $f^2(\gamma\lambda) \le 1/(1-\gamma\lambda)^2$. Therefore, by replacing $c(s,a) = \pi(a|s)/\pi_b(a|s)$ in Vanilla IS with a properly chosen constant λ, $Q^\pi(\lambda)$ algorithm successfully avoids an exponential large factor in the variance. However, choosing a small λ to control the variance has a side effect on the contraction factor. Intuitively, when λ is small, $Q^\pi(\lambda)$ does not effectively use the full n-step return. Hence the parameter $\omega = \mathcal{K}_{SA,\min}f(\gamma\lambda)(1-\gamma)$ in $Q^\pi(\lambda)$ is less (worse) than the one in Vanilla IS.

3.1.2 Finite-sample analysis of TB(λ)

In the TB(λ) algorithm, we have $c(s,a) = \lambda\pi(a|s)$ and $\rho(s,a) = \pi(a|s)/\pi_b(a|s)$. This implies that $c_{\max} = \lambda\pi_{\max}$ and $\rho_{\max} = r_{\max}$. Moreover, we have $D_c(s,a) = \lambda\sum_a \pi_b(a|s)\pi(a|s)$ and $D_\rho(s,a) = 1$ for all (s,a).

Theorem 3.3. *Consider Algorithm 1 with TB(λ) update. Under Assumption 2.1, suppose $\lambda \le \frac{1}{\pi_{b,\max}}$ and α is chosen s.t. $\alpha\tau_{\alpha,n} \le \frac{c_1\omega}{\log(2|\mathcal{S}||\mathcal{A}|/\omega)f(\gamma\lambda\pi_{\max})^2(\gamma r_{\max}+1)^2}$. Then we have for all $k \ge \tau_{\alpha,n}$: $\mathbb{E}[\|Q_k - Q^\pi\|_\infty^2] \le \zeta_1\left(1-\frac{\omega\alpha}{2}\right)^{k-\tau_{\alpha,n}} + \zeta_2\frac{f(\gamma\lambda\pi_{\max})^2(\gamma r_{\max}+1)^2\log(2|\mathcal{S}||\mathcal{A}|/\omega)}{\omega}\alpha\tau_{\alpha,n}$, where $\omega = \mathcal{K}_{SA,\min}f(\gamma D_{c,\min})(1-\gamma)$. This implies a sample complexity of $\tilde{\mathcal{O}}\left(\frac{\log^2(1/\epsilon)nf(\gamma\lambda\pi_{\max})^2(\gamma r_{\max}+1)^2}{\epsilon^2\mathcal{K}_{SA}^2f(\gamma D_{c,\min})^2(1-\gamma)^4}\right)$.*

Suppose we further choose $\lambda < 1/(\gamma\pi_{\max})$, the TB($\lambda$) algorithm also overcomes the high variance issue in Vanilla IS because $f(\gamma\lambda\pi_{\max}) \le 1/(1-\gamma\lambda\pi_{\max})$, which does not involve any exponential large factor. When compared to $Q^\pi(\lambda)$, an advantage of TB(λ) is that the constraint on λ is much relaxed. However, the same side effect on the contraction factor is also present here. To see this, since $D_{c,\min} = \lambda\min_{s,a}\sum_a \pi_b(a|s)\pi(a|s) \le 1$, the TB($\lambda$) algorithm does not effectively use the full n-step return. As a result, the parameter $\omega = \mathcal{K}_{SA,\min}f(\gamma D_{c,\min})(1-\gamma)$ in TB(λ) is less (worse) than the one in Vanilla IS.

3.1.3 Finite-sample analysis of Retrace(λ)

We now analyze the Retrace(λ) algorithm, where $c(s,a) = \lambda\min(1, \pi(a|s)/\pi_b(a|s))$ and $\rho(s,a) = \pi(a|s)/\pi_b(a|s)$ for all (s,a). This implies that $c_{\max} = \lambda$ and $\rho_{\max} = r_{\max}$. In addition, we have $D_c(s,a) = \lambda\sum_a \min(\pi_b(a|s), \pi(a|s))$ and $D_\rho(s,a) = 1$ for all (s,a).

Theorem 3.4. *Consider Algorithm 1 with Retrace(λ) update. Under Assumption 2.1, suppose $\lambda \le 1$ and α is chosen s.t. $\alpha\tau_{\alpha,n} \le \frac{c_1\omega}{\log(2|\mathcal{S}||\mathcal{A}|/\omega)f(\gamma\lambda)^2(\gamma r_{\max}+1)^2}$. Then for all $k \ge \tau_{\alpha,n}$ we have $\mathbb{E}[\|Q_k - Q^\pi\|_\infty^2] \le \zeta_1\left(1-\frac{\omega\alpha}{2}\right)^{k-\tau_{\alpha,n}} + \zeta_2\frac{f(\gamma\lambda)^2(\gamma r_{\max}+1)^2\log(2|\mathcal{S}||\mathcal{A}|/\omega)}{\omega}\alpha\tau_{\alpha,n}$, where $\omega = \mathcal{K}_{SA,\min}f(\gamma D_{c,\min})(1-\gamma)$. This implies a sample complexity of $\tilde{\mathcal{O}}\left(\frac{\log^2(1/\epsilon)nf(\gamma\lambda)^2(\gamma r_{\max}+1)^2}{\epsilon^2\mathcal{K}_{SA}^2f(\gamma D_{c,\min})^2(1-\gamma)^4}\right)$.*

Similarly as in $Q^\pi(\lambda)$ and TB(λ), the Retrace(λ) algorithm overcomes the high variance issue in Vanilla IS by truncating the importance sampling ratio at 1, which prevents an exponential large factor in the variance term. In addition, it does not require choosing λ to be extremely small as required in $Q^\pi(\lambda)$. As for the compromise in the contraction factor, note that $\min(1, \pi(a|s)/\pi_b(a|s)) \ge \pi(a|s)$, which implies that $D_c(s,a)$ (and hence $D_{c,\min}$) is larger in the Retrace(λ) algorithm than the TB(λ) algorithm. Therefore, Retrace(λ) does not truncate the n-step return as heavy as TB(λ). As a result, the parameter ω is larger (better) in Retrace(λ) than in TB(λ).

3.1.4 Finite-sample analysis of Q-Trace

Lastly, we analyze the Q-trace algorithm, where $c(s,a) = \min(\bar{c}, \pi(a|s)/\pi_b(a|s))$ and $\rho(s,a) = \min(\bar{\rho}, \pi(a|s)/\pi_b(a|s))$ for all (s,a). This implies that $c_{\max} = \bar{c}$ and $\rho_{\max} = \bar{\rho}$. Moreover, we have $D_c(s,a) = \sum_a \min(\bar{c}\pi_b(a|s), \pi(a|s))$ and $D_\rho(s,a) = \sum_a \min(\bar{\rho}\pi_b(a|s), \pi(a|s))$ for all (s,a).

Theorem 3.5. *Consider Algorithm 1 with Q-trace update. Under Assumption 2.1, suppose $\bar{c} \le \bar{\rho}$ and α is chosen such that $\alpha\tau_{\alpha,n} \le \frac{c_1\omega}{\log(2|\mathcal{S}||\mathcal{A}|/\omega)f(\gamma\bar{c})^2(\gamma\bar{\rho}+1)^2}$. Then for all $k \ge \tau_{\alpha,n}$ we have $\mathbb{E}[\|Q_k - Q^{\pi,\rho}\|_\infty^2] \le \zeta_1\left(1-\frac{\omega\alpha}{2}\right)^{k-\tau_{\alpha,n}} + \zeta_2\frac{f(\gamma\bar{c})^2(\gamma\bar{\rho}+1)^2\log(2|\mathcal{S}||\mathcal{A}|/\omega)}{\omega}\alpha\tau_{\alpha,n}$,*

where $\omega = \mathcal{K}_{SA,\min} f(\gamma D_{c,\min})(1-\gamma D_{\rho,\max})$. *This implies a sample complexity of* $\tilde{\mathcal{O}}\left(\frac{\log^2(1/\epsilon) n f(\gamma\bar{c})^2(\gamma\bar{\rho}+1)^2}{\epsilon^2 \mathcal{K}_{SA}^2 f(\gamma D_{c,\min})^2(1-\gamma D_{\rho,\max})^2(1-\gamma)^2}\right)$.

To avoid an exponential large variance, in view of the term $f(\gamma\bar{c})$ in our bound, we need to choose $\bar{c} \leq 1/\gamma$. The major difference between Q-trace and Retrace(λ) is that the importance sampling ratio $\rho(\cdot,\cdot)$ inside the temporal difference (line 4 of Algorithm 1) also involves a truncation. As a result, by choosing $\bar{c}=1$ and $\bar{\rho} \geq 1$, the Q-trace algorithm enjoys a larger (or better) parameter ω compared to Retrace(λ). However, as shown in Section 2.3, due to introducing the truncation level $\bar{\rho}$, the algorithm converges to a biased limit $Q^{\pi,\rho}$ instead of Q^{π}. Such truncation bias can be controlled using Proposition 2.1. These observations agree with the results [18], where the finite-sample bounds of Q-trace were first established.

Compared to [18], we have an improved sample complexity. Specifically, we have a sample complexity of $\tilde{\mathcal{O}}(\frac{\log^2(1/\epsilon) n f(\gamma\bar{c})^2(\gamma\bar{\rho}+1)^2}{\epsilon^2 \mathcal{K}_{SA}^2 f(\gamma D_{c,\min})^2(1-\gamma D_{\rho,\max})^2})$, while the result in [18] implies a sample complexity of $\tilde{\mathcal{O}}(\frac{\log^2(1/\epsilon) n f(\gamma\bar{c})^2(\gamma\bar{\rho}+1)^2}{\epsilon^2 \mathcal{K}_{SA}^3 f(\gamma D_{c,\min})^3(1-\gamma D_{\rho,\max})^3})$, which has an additional factor of $(\mathcal{K}_{SA} f(\gamma D_{c,\min})(1-\gamma D_{\rho,\max}))^{-1}$. Since $\mathcal{K}_{SA,\min}^{-1} \geq |\mathcal{S}||\mathcal{A}|$, our result improves the dependency on the size of the state-action space by a factor of at least $|\mathcal{S}||\mathcal{A}|$ compared to [18]. Similarly, since V-trace is an analog of Q-trace, we can improve the sample complexity for V-trace in [9].

4 Proof sketch of Proposition 2.4

The idea is to construct a stochastic matrix M'' such that the following two conditions are satisfied: (1) M'' dominates M in the sense that $M''_{ij} \geq M_{ij}$ for all i,j, and (2) the Markov chain associated with M'' is irreducible, hence admits a unique stationary distribution μ satisfying $\mu_i > 0$ for all i. Using μ as weights, we have the desired result. The detailed analysis is presented in Appendix A.7. We here only present how to construct such a stochastic matrix M''.

First of all, consider the special case where M itself is irreducible. Then we first scale up M by a factor of $1/(1-\omega)$ to obtain $M' = \frac{M}{1-\omega}$, which is clearly a substochastic matrix, with modulus being 0. Hence there exists a stochastic matrix M'' that dominates M' (and hence also dominates M). Moreover, M'' is also irreducible. This is equivalent to choosing $\theta = 1$ in Proposition 2.4.

Now consider the general case where M is not necessarily irreducible. We construct the intermediate matrix M' by performing a convex combination of the matrix $\frac{M}{1-\omega}$ and the uniform stochastic matrix $\frac{E}{d}$, where E is the all one matrix, with weight $\frac{1-\omega}{1-\theta\omega}$. Specifically, for any $\theta \in (0,1)$, we define

$$M' = \left(\frac{1-\omega}{1-\theta\omega}\right)\frac{M}{1-\omega} + \left(1 - \frac{1-\omega}{1-\theta\omega}\right)\frac{E}{d}.$$

Note that M' is a non-negative matrix. In addition, since $M'\mathbf{1} \leq \frac{1-\omega}{1-\theta\omega}\mathbf{1} + \left(1-\frac{1-\omega}{1-\theta\omega}\right)\mathbf{1} = \mathbf{1}$, where $\mathbf{1}$ is the all one vector, the matrix M' is a substochatic matrix with modulus 0, and is also irreducible because all its entries are strictly positive. Therefore, there exists a stochastic matrix M'' such that $M'' \geq M'$. In addition, since M'' also has strictly positive entries, the Markov chain associated with M'' is irreducible, hence admits a unique stationary distribution $\mu \in \Delta^d$. By our construction, we can show a lower bound on the components of the stationary distribution μ.

5 Conclusion

In this work, we establish finite-sample guarantees of general n-step off-policy TD-learning algorithms. The key in our approach is to identify a generalized Bellman operator and establishes its contraction property with respect to a weighted ℓ_p-norm for each $p \in [1,\infty)$, with a uniform contraction factor. Our results are used to derive finite-sample guarantees of variants of n-step off-policy TD-learning algorithms in the literature. Specifically, for $Q^{\pi}(\lambda)$, TB(λ), and Retrace(λ), we provide the first-known results, and for Q-trace, we improve the result in [18]. The finite-sample bounds we establish also provide insights about the trade-off between the convergence rate and the variance.

One limitation in our analysis is that it currently applies only to the tabular setting. Extending the result to the function approximation setting is our immediate future direction. The weighted ℓ_p-norm (especially ℓ_2-norm) contraction property we established in this paper will play an important role.

References

[1] Aggarwal, C. C. et al. (2016). *Recommender systems*, volume 1. Springer.

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

[3] Berman, A. and Plemmons, R. J. (1994). *Nonnegative matrices in the mathematical sciences*. SIAM.

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

[5] Bottou, L., Curtis, F. E., and Nocedal, J. (2018). Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311.

[6] Cai, Q., Yang, Z., Lee, J. D., and Wang, Z. (2019). Neural temporal-difference learning converges to global optima. *Advances in Neural Information Processing Systems*, 32.

[7] Cayci, S., Satpathi, S., He, N., and Srikant, R. (2021). Sample Complexity and Overparameterization Bounds for Projection-Free Neural TD Learning. *Preprint arXiv:2103.01391*.

[8] Chen, Z., Maguluri, S. T., Shakkottai, S., and Shanmugam, K. (2020). Finite-Sample Analysis of Contractive Stochastic Approximation Using Smooth Convex Envelopes. *Advances in Neural Information Processing Systems*, 33.

[9] Chen, Z., Maguluri, S. T., Shakkottai, S., and Shanmugam, K. (2021). A Lyapunov theory for finite-sample guarantees of asynchronous Q-learning and TD-learning variants. *Preprint arXiv:2102.01567*.

[10] Cichosz, P. and Mulawka, J. J. (1995). Fast and efficient reinforcement learning with truncated temporal differences. In *Machine Learning Proceedings 1995*, pages 99–107. Elsevier.

[11] Dalal, G., Szörényi, B., Thoppe, G., and Mannor, S. (2018). Finite sample analyses for TD (0) with function approximation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

[12] Dayan, P. (1992). The convergence of TD(λ) for general λ. *Machine learning*, 8(3-4):341–362.

[13] Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., et al. (2018). IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures. In *International Conference on Machine Learning*, pages 1407–1416.

[14] Glynn, P. W. and Iglehart, D. L. (1989). Importance sampling for stochastic simulations. *Management science*, 35(11):1367–1392.

[15] Gottesman, O., Futoma, J., Liu, Y., Parbhoo, S., Celi, L., Brunskill, E., and Doshi-Velez, F. (2020). Interpretable off-policy evaluation in reinforcement learning by highlighting influential transitions. In *International Conference on Machine Learning*, pages 3658–3667. PMLR.

[16] Harutyunyan, A., Bellemare, M. G., Stepleton, T., and Munos, R. (2016). Q(λ) with Off-Policy Corrections. In *International Conference on Algorithmic Learning Theory*, pages 305–320. Springer.

[17] Jaakkola, T., Jordan, M. I., and Singh, S. P. (1994). Convergence of stochastic iterative dynamic programming algorithms. In *Advances in neural information processing systems*, pages 703–710.

[18] Khodadadian, S., Chen, Z., and Maguluri, S. T. (2021). Finite-Sample Analysis of Off-Policy Natural Actor-Critic Algorithm. *The 38th International Conference on Machine Learning*.

[19] Levin, D. A. and Peres, Y. (2017). *Markov chains and mixing times*, volume 107. American Mathematical Soc.

[20] Maei, H. R. (2011). *Gradient temporal-difference learning algorithms*. University of Alberta.

[21] Munos, R., Stepleton, T., Harutyunyan, A., and Bellemare, M. G. (2016). Safe and efficient off-policy reinforcement learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 1054–1062.

[22] Precup, D., Sutton, R. S., and Singh, S. P. (2000). Eligibility Traces for Off-Policy Policy Evaluation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 759–766.

[23] Shalev-Shwartz, S., Shammah, S., and Shashua, A. (2016). Safe, multi-agent, reinforcement learning for autonomous driving. *Preprint arXiv:1610.03295*.

[24] Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. (2017). Mastering the game of go without human knowledge. *Nature*, 550(7676):354.

[25] Singh, S. P., Jaakkola, T., and Jordan, M. I. (1994). Learning without state-estimation in partially observable Markovian decision processes. In *Machine Learning Proceedings 1994*, pages 284–292. Elsevier.

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

[27] Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44.

[28] Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

[29] Sutton, R. S., Maei, H. R., Precup, D., Bhatnagar, S., Silver, D., Szepesvári, C., and Wiewiora, E. (2009). Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 993–1000.

[30] Sutton, R. S., Mahmood, A. R., and White, M. (2016). An emphatic approach to the problem of off-policy temporal-difference learning. *The Journal of Machine Learning Research*, 17(1):2603–2631.

[31] Thoppe, G. and Borkar, V. (2019). A concentration bound for stochastic approximation via Alekseev's formula. *Stochastic Systems*, 9(1):1–26.

[32] Tsitsiklis, J. N. (1994). Asynchronous stochastic approximation and Q-learning. *Machine learning*, 16(3):185–202.

[33] Tsitsiklis, J. N. and Van Roy, B. (1997). Analysis of temporal-difference learning with function approximation. In *Advances in neural information processing systems*, pages 1075–1081.

[34] Van Seijen, H., Mahmood, A. R., Pilarski, P. M., Machado, M. C., and Sutton, R. S. (2016). True online temporal-difference learning. *The Journal of Machine Learning Research*, 17(1):5057–5096.

[35] Watkins, C. J. C. H. (1989). *Learning from delayed rewards*. King's College, Cambridge United Kingdom.

[36] Zhao, Y., Zeng, D., Socinski, M. A., and Kosorok, M. R. (2011). Reinforcement learning strategies for clinical trials in nonsmall cell lung cancer. *Biometrics*, 67(4):1422–1433.

Checklist

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