A Tighter Convergence Proof of Reverse Experience Replay

Nan Jiang, Jinzhao Li, Yexiang Xue
{jiang631, li4255, yexiang}@purdue.edu
Department of Computer Science
Purdue University, USA

Abstract

In reinforcement learning, Reverse Experience Replay (RER) is a recently proposed algorithm that attains better sample complexity than the classic experience replay method. RER requires the learning algorithm to update the parameters through consecutive state-action-reward tuples in reverse order. However, the most recent theoretical analysis only holds for a minimal learning rate and short consecutive steps, which converge slower than those large learning rate algorithms without RER. In view of this theoretical and empirical gap, we provide a tighter analysis that mitigate the limitation on the learning rate and the length of consecutive steps. Furthermore, we show theoretically that RER converges with a larger learning rate and a longer sequence.

1 Introduction

Reinforcement Learning (RL) is highly successful for a variety of practical problems in the realm of long-term decision-making. Experience Replay (ER) of historical trajectories plays a vital role in Reinforcement Learning (RL) algorithms (Lin, 1992; Mnih et al., 2015). The trajectory is a sequence of transitions (states, actions, and reward tuples). The memory space used to store these experience trajectories is noted as the replay buffer. The methods to sample transitions from the experienced trajectories determine the rate and stability of the convergence of RL algorithms.

Recently, Reversed Experience Replay (RER) (Florensa et al., 2017; Rotinov, 2019; Lee et al., 2019; Agarwal et al., 2022) is an approach inspired by the hippocampal reverse replay mechanism in human neuron (Foster & Wilson, 2006; Ambrose et al., 2016; Igata et al., 2021). Theoretical analysis shows that RER improves the convergence rate towards optimal policies in comparison with ER-based algorithms. Unlike ER, which samples transitions uniformly (van Hasselt et al., 2016) (known as classic experience replay) or weightily (Schaul et al., 2016) (known as prioritized experience replay) from the replay buffer, RER samples consecutive sequences of transitions from the buffer and reversely fed into the learning algorithm.

However, the most recent theoretical analysis only holds for a minimal learning rate and short consecutive steps Agarwal et al. (2022), which converge slower than those large learning rate algorithms without RER. We attempt to bridge the gap between theory and practice for the newly proposed reverse experience replay algorithm.

In this paper, we provide a tighter analysis that relaxes the limitation on the learning rate and the length of the consecutive tuples. Our key idea is to transform the original problem involving a giant summation (shown in Equation 3) into a combinatorial counting problem (shown in Lemma 2), which greatly simplifies the whole problem. We hope the new idea of transforming the original problem into a combinatorial counting problem can enlighten other relevant domains. Furthermore, we show theoretically that RER converges faster with a larger learning rate η and a longer consecutive sequence L of state-action-reward tuples.

2 Preliminaries

Markov Decision Process We consider a Markov decision process (MDP) with discounted rewards, noted as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$. Here $\mathcal{S} \subset \mathbb{R}^d$ is the set of states, $\mathcal{A}$ is the set of actions, and $\gamma \in (0, 1)$ indicates the discounting factor. We use $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ as the transition probability kernel of MDP. For each pair $(s, a) \in \mathcal{S} \times \mathcal{A}$, $P(s'|s, a)$ is the probability of transiting to state s' from state s when action a is executed. The reward function is $r : \mathcal{S} \times \mathcal{A} \to [-1, 1]$, such that $r(s, a)$ is the immediate reward from state s when action a is executed (Puterman, 1994). The policy π is a mapping from states to a distribution over the set of actions: $\pi(s) : \mathcal{A} \to [0, 1]$, for $s \in \mathcal{S}$. A trajectory is noted as $\{(s_t, a_t, r_t)\}_{t=0}^{\infty}$, where s_t (*resp.* a_t) is the state (*resp.* the action taken) at time t, and $r_t = r(s_t, a_t)$ is the reward received at time t.

Value Function & Q-Function The value function of a policy π is noted as $V^\pi : \mathcal{S} \to \mathbb{R}$. For $s \in \mathcal{S}$, $V^\pi(s) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t | s_0 = s)\right]$, which is the expected discounted cumulative reward received when 1) the initial state is $s_0 = s$, 2) the actions are taken based on the policy π, *i.e.*, $a_t \sim \pi(s_t)$, for $t \geq 0$. 3) the trajectory is generated by the transition kernel, *i.e.*, $s_{t+1} \sim P(\cdot|s_t, a_t)$, for all $t \geq 0$. Similarly, let $Q^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ be the action-value function (also known as the Q-function) of a policy π. For $(s, a) \in \mathcal{S} \times \mathcal{A}$: $Q^\pi(s, a) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t | s_0 = s, a_0 = a)\right]$.

There exists an optimal policy, denoted as π^* that maximizes $Q^\pi(s, a)$ uniformly over all state-action pairs $(s, a) \in \mathcal{S} \times \mathcal{A}$ (Watkins, 1989). We denote Q^* as the Q-function corresponding to π^*, *i.e.*, $Q^* = Q^{\pi^*}$. The Bellman operator $\mathcal{T}$ on a Q-function is defined as: for $(s, a) \in \mathcal{S} \times \mathcal{A}$,

$$\mathcal{T}(Q)(s, a) := r(s, a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} \left[\max_{a' \in \mathcal{A}} Q(s', a')\right].$$

The optimal Q-function Q^* is the unique fixed point of the Bellman operator (Bertsekas & Yu, 2012).

Q-learning The Q-learning algorithm is a model-free algorithm to learn Q^* (Watkins & Dayan, 1992). The high-level idea is to find the fixed point of the Bellman operator. Given the trajectory $\{(s_t, a_t, r_t)\}_{t=0}^{\infty}$ generated by some underlying behavior policy π', the asynchronous Q-learning algorithm estimates a new Q-function $Q_{t+1} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ at each time. At time $t \geq 0$, given a transition (s_t, a_t, r_t, s_{t+1}), the algorithm update as follow:

$$\begin{aligned} Q_{t+1}(s_t, a_t) &= (1 - \eta) Q_t(s_t, a_t) + \eta \mathcal{T}_{t+1}(Q_t)(s_{t+1}, a_t), \\ Q_{t+1}(s, a) &= Q_t(s, a), \qquad \text{for all } (s, a) \neq (s_t, a_t). \end{aligned} \tag{1}$$

Here $\eta \in (0, 1)$ is the learning rate and $\mathcal{T}_{t+1}$ is the *empirical* Bellman operator: $\mathcal{T}_{t+1}(Q_t)(s_t, a_t) := r(s_t, a_t) + \gamma \max_{a' \in \mathcal{A}} Q_t(s_{t+1}, a')$. Under mild conditions, Q_t will converge to the fixed point of the Bellman operator and hence to Q^*. In practice, a tabular structure is used to store the values of $Q_t(s, a)$ for $(s, a) \in \mathcal{S} \times \mathcal{A}$.

Q-learning with Function Approximation When the state space $\mathcal{S}$ is large, the asynchronous Q-learning in Equation (1) cannot be applied since it needs to loop over a table of all states and actions. In this case, function approximation is brought into Q-learning. Let $Q^w : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ be an approximated Q-function, which is typically represented with a deep neural network (Mnih et al., 2015) and w denotes the parameters of the neural network. Q^w is often called the Q-network. Given a batch of transitions $\{(s_{t_i}, a_{t_i}, r_{t_i}, s_{t_i+1})\}_{i=1}^{m}$, define y_{t_i} as the image of $Q^{w'}(s_{t_i}, a_{t_i})$ under the empirical Bellman operator:

$$y_{t_i} := r_{t_i} + \gamma \max_{a' \in \mathcal{A}} Q^{w'}(s_{t_i+1}, a'), \qquad \text{for } 1 \leq i \leq m$$

where w' represents the parameters in *target* neural network. Parameters w' are synchronized to w every T_{target} steps of Stochastic Gradient Descent (SGD). Since Q^* is the fixed point of the Bellman operator, y_{t_i} should match $Q^w(s_{t_i}, a_{t_i})$ when Q^w converges to Q^*. Hence, Learning is done via minimizing the following objective using SGD: $\ell(w) = \frac{1}{m} \sum_{i=1}^{m} \|y_{t_i} - Q^w(s_{t_i}, a_{t_i})\|_2^2$.

Experience Replay For the Q-learning with function approximation, the new trajectories are generated by executing a behavioral policy, which are then saved into the *replay buffer*, noted as $\mathcal{B}$. When learning to minimize $\ell(w)$, SGD is performed on batches of *randomly sampled* transitions from the replay buffer. This process is often called Experience Replay (ER) (Lin, 1992; Li et al., 2022).

To improve the stability and convergence rate of Q-learning, follow-up works sample transitions from the replay buffer with non-uniform probability distributions. Prioritized experience replay favors those transitions with a large temporal difference (TD) errors (Schaul et al., 2016). Discor (Kumar et al., 2020) favors those transitions with small Bellman errors. LaBER proposes a generalized TD error to reduce the variance of gradient and improve learning stability (Lahire et al., 2022).

Reverse Experience Replay is a recently proposed variant of ER (Goyal et al., 2019; Bai et al., 2021; Agarwal et al., 2022). RER samples *consecutive* sequences of transitions (of length L) from the replay buffer. The Q-learning updates are performed in the *reverse* order of the sampled sequences. Compared with ER, RER converges faster towards the optimal policy empirically (Lee et al., 2019) and theoretically (Agarwal et al., 2022), under tabular and linear MDP settings. One intuitive explanation of why RER works is to consider a sequence of consecutive transitions $s_1 \xrightarrow{a_1,r_1} s_2 \xrightarrow{a_2,r_2} s_3$. Incorrect Q-function estimation of $Q(s_2, a_2)$ will affect the estimation of $Q(s_1, a_1)$. Hence, reverse order updates allow the Q-value updates of $Q(s_1, a_1)$ to use the most up-to-date value of $Q(s_2, a_2)$, hence accelerating the convergence.

2.1 Problem Setups for Reverse Experience Replay

Linear MDP Assumption In this paper, we follow the definition of linear MDP from Zanette et al. (2020), which states that the reward function can be written as the inner product of the parameter w and the feature function ϕ. Therefore, the Q function depends only on w and the feature vector $\phi(s, a) \in \mathbb{R}^d$ for State $s \in \mathcal{S}$ and action $a \in \mathcal{A}$.

Assumption 1 (Linear MDP setting (Zanette et al., 2020))**.** *There exists a vector $w \in \mathbb{R}^d$ such that $R(s, a; w) = \langle w, \phi(s, a)\rangle$, and the transition probability is proportional to its corresponding feature $\mathcal{P}(\cdot|s, a) \propto \phi(s, a)$. Therefore, the optimal Q-function is $Q^*(s, a; w^*) = \langle w^*, \phi(s, a)\rangle$ for every $s \in \mathcal{S}, a \in \mathcal{A}$.*

Definition 1 is the current popular Linear MDP assumption that allows us to quantify the convergence rate (or sample complexity) for the learning algorithm (Zanette et al., 2020; Agarwal et al., 2022). To get the final convergence rate result, we need the following additional assumptions. Assume the sequence of consecutive state-action tuples is of length L and the constant learning rate in the gradient descent is noted as η.

Definition 1. *Given the feature function $\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$. Denote the largest inner product between parameter w and the feature function ϕ as $\|w\|_\phi = \sup_{(s,a)} |\langle \phi(s, a), w\rangle|$. For clarity, we would use the simplified notation $\phi_l = \phi(s_l, a_l)$.*

Assumption 2 (from Zanette et al. (2020))**.** *The MDP has zero inherent Bellman error and $\phi(s, a)^\top \phi(s, a) \leq 1$ for all $(s, a) \in \mathcal{S} \times \mathcal{A}$. There exists constant $\kappa > 0$, such that $\mathbb{E}_{(s,a)\sim\mu} \phi(s, a)\phi(s, a)^\top \succeq \mathrm{I}/\kappa$. Here μ is the stationary distribution over all the state-action pairs of the Markov chain determined by the transition kernel and the policy.*

Remark 1. *Suppose we pick a set of state-action tuples $\mathcal{L} = \{(s, a)|(s, a) \in \mathcal{S} \times \mathcal{A}\}$, which may contains duplicated tuples. By linearity of expectation, we have: $\mathbb{E}_\mu \left(\sum_{(s,a)\in\mathcal{L}} \phi(s, a)\phi(s, a)^\top\right) = \sum_{\mathcal{L}} \mathbb{E}_{(s,a)\sim\mu} \left(\phi(s, a)\phi(s, a)^\top\right) \succeq \frac{|\mathcal{L}|}{\kappa}\mathrm{I}$. Here $|\mathcal{L}|$ indicates the number of state-action tuples in this set.*

Definition 2. *Let* I *be an identity matrix of dimension* $d \times d$ *and* $\eta \in \mathbb{R}$ *as the learning rate. Define matrix* Γ_l *recursively as follow:*

$$\Gamma_l \coloneqq \begin{cases} \mathrm{I} & \text{for } l = 0, \\ \left(\mathrm{I} - \eta \phi_{L+1-l} \phi_{L+1-l}^\top\right) \Gamma_{l-1} & \text{for } 1 \leq l \leq L, \end{cases}$$

where $\phi_{L+1-l} = \phi(s_{L+1-l}, a_{L+1-l})$*. The explicit form for* Γ_L *is:*

$$\Gamma_L = \left(\mathrm{I} - \eta \phi_1 \phi_1^\top\right) \left(\mathrm{I} - \eta \phi_2 \phi_2^\top\right) \ldots \left(\mathrm{I} - \eta \phi_L \phi_L^\top\right) = \prod_{l=1}^{L} \left(\mathrm{I} - \eta \phi_l \phi_l^\top\right)$$

The semantic interpretation of Γ_L *is the coefficient of the bias term (in Lemma 3) used in the error analysis of the parameter of the learning algorithm. The reason of having this joint product is because of RER algorithm updates the parameter over a sub-sequence of consecutive state-action tuples in reverse order.*

Its norm value is impacted by the sequence length L *and the learning rate* η*. When the norm of* Γ_L *is small, parameter of the learning model will quickly converge to its optimal.*

3 Methodology

3.1 Motivation

Let μ be the stationary distribution of the state-action pair in the MDP, η be the learning rate of the gradient descent algorithm and L be the length of the consecutive state-action tuples processed by the RER algorithm. Previous work (Agarwal et al., 2022, Lemmas 8 and 14) states that: when $\eta L \leq \frac{1}{3}$, the following result holds:

$$\mathbb{E}_{(s,a)\sim\mu}\left[\Gamma_L^\top \Gamma_L\right] \preceq \mathrm{I} - \eta \sum_{l=1}^{L} \mathbb{E}_{(s,a)\sim\mu}\left[\phi_l \phi_l^\top\right] \preceq \left(1 - \frac{\eta L}{\kappa}\right) \mathrm{I}, \tag{2}$$

where the matrix Γ_L is defined in Definition 2 and will be used as the "coefficient" in convergence analysis in Lemma 3; the positive semi-definite property "$\preceq$" is defined between two matrices on both sides (in Definition 4); I is an identity matrix of dimension $d \times d$; coefficient $\kappa > 0$ is introduced in Assumption 2. Note that the matrix Γ_L was mentioned in (Agarwal et al., 2022, Appendix E, Equation 5). We formalize its definition and clean up unnecessary variables in the original definition.

The requirement in Equation (2) was further brought into the requirement of convergence in (Agarwal et al., 2022, Theorem 1). It states that the RER algorithm cannot be applied to process too long sequences of consecutive state-action tuples (which correspond to a large value of L) or too large of learning rate in the gradient descent step (i.e., η). This is the major limitation between theoretical justification and real-world practice of the RER algorithm. In this research, we mitigate the above gap by offering a tighter theoretical justification to ease the requirement $\eta L \leq 1/3$.

We first explain the main difficulty of upper-bound the term $\mathbb{E}_{(s,a)\sim\mu}\left[\Gamma_L^\top \Gamma_L\right]$. According to Definition 2, we can expand the term $\Gamma_L^\top$ as $\Gamma_L^\top = \left(\mathrm{I} - \eta \phi_L \phi_L^\top\right) \cdots \left(\mathrm{I} - \eta \phi_1 \phi_1^\top\right)$. Based on the linearity of expectation, we expand the whole joint product $\Gamma_L^\top \Gamma_L$ subject to expectation as follows:

$$\begin{aligned} \mathbb{E}_{(s,a)\sim\mu}\left[\Gamma_L^\top \Gamma_L\right] &= \mathbb{E}_{(s,a)\sim\mu}\left[\left(\mathrm{I} - \eta \phi_L \phi_L^\top\right) \cdots \left(\mathrm{I} - \eta \phi_1 \phi_1^\top\right) \left(\mathrm{I} - \eta \phi_1 \phi_1^\top\right) \cdots \left(\mathrm{I} - \eta \phi_L \phi_L^\top\right)\right] \\ &= \mathrm{I} - 2\eta \mathbb{E}_{(s,a)\sim\mu}\left[\sum_{l=1}^{L} \phi_l \phi_l^\top\right] + \mathbb{E}_{(s,a)\sim\mu}\left[\sum_{k=2}^{2L} (-\eta)^k \sum_{l_1,\ldots,l_k} \phi_{l_1} \phi_{l_1}^\top \ldots \phi_{l_k} \phi_{l_k}^\top\right]. \end{aligned} \tag{3}$$

In the third term of the right-hand side (RHS) of the second line, the summation is over all valid combinations of the indices $(l_1, l_2, \ldots, l_k)$, for $l_1, l_2, \ldots, l_k \in \{1, 2, \ldots, L\}$. It is achieved by first

determining the index l_1 with a value in the sequence $[L, L-1, \ldots, 2, 1, 1, 2 \ldots, L-1, L]$, from the first row of the above equation. Thereafter determining the second index l_2, where l_2 should be on the right of l_1. The valid combination constraint requires the whole picked sequence $l_1, \ldots, l_k$ to satisfy: l_{i-1} should be on the left of l_i.

As there are combinatorially many high-order terms, the main difficulty is to upper bound the whole product $\Gamma_L^\top \Gamma_L$ subject to its expectation. The high-level idea of our approach is to show that the RHS of Equation (3) can be upper bounded in the form of $\mathbb{E}_{(s,a)\sim\mu}\left[\sum_{l=1}^{L} \phi_l \phi_l^\top\right]$ with appropriate coefficients. More specifically, we prove that the third term on the RHS, which contains combinatorially many terms in the form of $\phi_{l_1}\phi_{l_1}^\top \ldots \phi_{l_k}\phi_{l_k}^\top$, can be bounded by terms containing only $\phi_l \phi_l^\top$ (with $1 \le l \le L$) through combinatorial counting.

Theorem 1. *Let μ be the stationary distribution of the state-action pair in the MDP. The following matrix's positive semi-definite inequalities hold: when $\eta \in (0, 1)$,*

$$\mathbb{E}_{(s,a)\sim\mu}\left[\Gamma_L^\top \Gamma_L\right] \preceq \left(1 - \frac{\eta(4-2L)L + L - (1-\eta)^{L-1}L - \eta^2 L}{\kappa}\right) \mathrm{I},$$

where the matrix Γ_L is defined in Definition 2. Here "$\preceq$" is defined between two matrices on both sides (See Definition 4) for the positive semi-definite property[1]*.*

Proof of Sketch. By linearity of expectation, the second term of Equation (3) can be bouned as

$$-2\eta \mathbb{E}_{(s,a)\sim\mu}\left[\sum_{l=1}^{L} \phi_l \phi_l^\top\right] = -2\eta \sum_{l=1}^{L} \mathbb{E}_{(s,a)\sim\mu}\left[\phi_l \phi_l^\top\right] = -2\eta L \mathbb{E}_{(s,a)\sim\mu}\left[\phi\phi^\top\right] \preceq -\frac{2\eta L}{\kappa}\mathrm{I}.$$

Based on the result in the proposed Lemma (2), we have:

$$\begin{aligned}
\mathbb{E}_{(s,a)\sim\mu}\left[\sum_{k=2}^{2L}(-\eta)^k \sum_{l_1,\ldots,l_k} \phi_{l_1}\phi_{l_1}^\top \ldots \phi_{l_k}\phi_{l_k}^\top\right] &\preceq \mathbb{E}_{(s,a)\sim\mu}\left[\sum_{k=2}^{2L}(-\eta)^k \sum_{l_1,\ldots,l_k} \frac{1}{2}(\phi_{l_1}\phi_{l_1}^\top + \phi_{l_k}\phi_{l_k}^\top)\right] \\
&\preceq \left((1-\eta)^{L-1} + \eta^2 + \eta(2L-2) - 1\right) \mathbb{E}_{(s,a)\sim\mu}\left[\sum_{l=1}^{L} \phi_l \phi_l^\top\right] \\
&\preceq \frac{(1-\eta)^{L-1}L + \eta^2 L + \eta(2L-2)L - L}{\kappa}\mathrm{I}.
\end{aligned}$$

Combining the results in the above two inequalities, we finally have the upper bound in the theorem. Please see Appendix B for a detailed proof. □

Theorem 1 holds based on the proposed new analysis in Section 3.2. Theorem 1 will be applied as the key component in the final convergence proof of the RER algorithm, which is presented in Section 4.

3.2 Relaxing the Requirement $\eta L \le 1/3$ through Combinatorial Counting

Lemma 1. *Let $\mathbf{x} \in \mathbb{R}^d$ be any d-dimensional non-zero vector. For $l_1, \ldots, l_k \in \{1, 2 \ldots, L\}$ and $2 \le k \le 2L$, we consider one high-order term $\phi_{l_1}\phi_{l_1}^\top \ldots \phi_{l_k}\phi_{l_k}^\top$ in Equation (3). By Assumption 1, we can relax the high-order term as:*

$$|\mathbf{x}^\top \phi_{l_1}\phi_{l_1}^\top \ldots \phi_{l_k}\phi_{l_k}^\top \mathbf{x}| \le \frac{1}{2}\mathbf{x}^\top\left(\phi_{l_1}\phi_{l_1}^\top + \phi_{l_k}\phi_{l_k}^\top\right)\mathbf{x}$$

[1]The code implementation for the numerical evaluation of the equalities and inequalities in the proof is available at: https://github.com/jiangnanhugo/RER-proof.

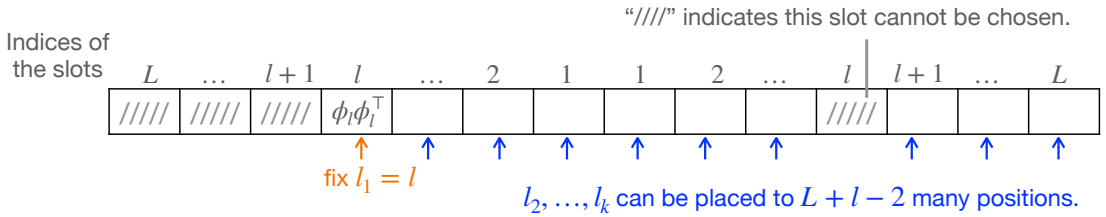

Figure 1: Case 1 in the propose combinatorial counting procedure. The task is to count how many terms $\phi_{l_1}\phi_{l_1}^\top \ldots \phi_{l_k}\phi_{l_k}^\top$ can be "reduced to" $\phi_l\phi_l^\top$ for a fixed l using Lemma 1, for $1 \le l \le L$. When we let l_1 pick the left l-th slot, l_k cannot choose the left terms with indices $L, \ldots, l+1$. Because of the sequential ordering constraint l_i should be on the right of l_{i-1}. To avoid double counting, we also disallow assigning the right l-th slot to l_k. There are $2L - (L - (l+1)) - 1 = L + l - 2$ many slots to assign the rest sequences $l_2, \ldots, l_k$ of length $k-1$. Therefore, we obtain $\binom{L+l-2}{k-1}$ many terms for the first case. See all the rest cases in Figure 2 in the appendix.

The proof of the above inequality is in Appendix A.1. The above result implies that: after relaxation, only the first term (i.e., $\phi_{l_1}\phi_{l_1}^\top$) indexed by l_1 and the last term (i.e., $\phi_{l_k}\phi_{l_k}^\top$) indexed by l_k determine the upper bound of the high order term $\phi_{l_1}\phi_{l_1}^\top \ldots \phi_{l_k}\phi_{l_k}^\top$. This relaxation allows us to transform the original combinatorial summation problem $\sum_{1 \le l_1, \ldots, l_k \le L}$ to count how many cases of picking valid l_1 and l_k at each possible position in the consecutive sequence of state-action tuples.

Lemma 2. *Based on the relaxation in Lemma 1, the weighted summation* $\sum_{k=2}^{2L}(-\eta)^k \sum_{l_1,\ldots,l_k}$ *in Equation (3) can be expanded combinatorially as follow:*

$$\sum_{k=2}^{2L}(-\eta)^k \sum_{l_1,\ldots,l_k} \frac{1}{2}(\phi_{l_1}\phi_{l_1}^\top + \phi_{l_k}\phi_{l_k}^\top) = \underbrace{\sum_{k=2}^{2L}(-\eta)^k \sum_{l=1}^{L} \left(\binom{L+l-2}{k-1} + \binom{L-l}{k-1} + \binom{2l-2}{k-2} \right)}_{\text{sum over combinatorially many terms}} \phi_l\phi_l^\top$$

Sketch of proof. As shown in Figure 1, we have two arrays of length L. The indices of the array are symmetry to each other, where the left one decreases from L to 1 and the right one increases from 1 to L. The two arrays are set up in this way to represent the indices of the matrix product in the first line of Equation (3). The left array simulates Γ_L and the right array simulates $\Gamma_L^\top$.

The key idea is: for a fixed l $(1 \le l \le L)$, we count the number of combination of l_1, l_k that can produce $\phi_l\phi_l^\top$. The first case shown in Figure 1 is when we let l_1 pick the left l-th slot, l_k cannot choose the slots in the left array with indices $L \ldots, l+1$. Because the sequential ordering constraint enforces that l_{i-1} should be on the left of l_i needs to be preserved. To avoid double counting, we also disallow assigning the right l-th slot to l_k. Therefore, there are $L + l - 2$ many slots to assign the sequences $l_2, \ldots, l_k$. This contributes to the first slot $\binom{L+l-2}{k-1}$ on the right-hand side. We leave all the rest cases in Figure 2 and their analysis in Appendix A.2, which contributes to the second and last term in Equation 2. □

Lemma 2 shows the process of transforming the complex summation $\sum_{l_1,\ldots,l_k}$ into a simpler summation form $\sum_{l=1}^{L}$, which becomes much easier to get the tighter upper bound. The upper bound in Lemma 2 is obtained by combinatorially counting the number of possible subcases and avoiding double counting.

4 Sample Complexity of Reverse Experience Replay-based Q-learning on Linear MDP

The following analysis is based on the assumption that every sub-trajectory of length L is almost (or asymptotically) independent of each other with high probability. This is commonly known as the mixing requirement for Markovian data: the statistical dependence between the two sub-trajectories

Algorithm 1 Episodic Q-learning with Reverse Experience Replay

Require: Sequence length L of consecutive state-action tuples; Replay buffer $\mathcal{B}$; Total learning episodes T; Target network update frequency N.
Ensure: The best-learned policy.

for $t = 1$ **to** T **do**
  Act by ϵ-greedy strategy *w.r.t.* policy π.
  Save the new trajectory into the replay buffer $\mathcal{B}$.
  Retrieve a sub-trajectory τ_L from buffer $\mathcal{B}$, where $\tau_l := (s_l, a_l, r_l)$, for all $1 \leq l \leq L$.
  for $l = 1$ **to** L **do**
    $\varepsilon \leftarrow r_{L-l} + \gamma \max_{a' \in \mathcal{A}} Q(s_{L+1-l}, a'; \theta_k) - Q_{L+1-l}$ ▷ reverse experience replay
    $w_{t,l+1} \leftarrow w_{t,l} + \eta \varepsilon \nabla Q_{t,L+1-l}$
  if $t \mod N = 0$ **then**
    $\theta_k \leftarrow w_{t,L+1}$ ▷ online target update
    $k \leftarrow k + 1$
  $\pi(s) \leftarrow \arg\max_{a \in \mathcal{A}}, Q(s, a; w_{t,L+1})$, for all $s \in \mathcal{S}$. ▷ policy extraction
Return The converged policy π.

τ_L, τ'_L goes to zero when they are far apart along the trajectory (Tagorti & Scherrer, 2015; Nagaraj et al., 2020).

Prior work Lee et al. (2019) only gives a convergence proof for the RER approach without the rate of convergence, due to the difficulty of quantifying the deep neural network. Linear MDP (in Definition 1), which assumes the reward function and transition kernel are linearly approximately by the feature and the optimal parameter w^* are then used to give an asymptotic behavior for the performance of the RER. Most recently, Agarwal et al. (2022) gives the first theoretical proof for RER. However, their analysis only works under strict conditions. The most notable one is the requirement on the minimal learning rate $\eta \leq \frac{1}{3L}$. Such limitations show that RER would not be able to compete against the plain ER method with a large learning rate.

To resolve the above challenge, we present a rigorous analysis of the RER method. In our Theorem 1, we directly remove the constraints on the learning rate for the algorithm to converge. Instead, we found the convergence rate can be improved with a large learning rate as well as a long sequence of State-action-reward tuples. Therefore, we bridge the gap between theoretical convergence analysis and the empirical learning result.

Lemma 3 (Bias and variance decomposition)**.** *Let the error terms for every parameter w as the difference between empirical estimation and true MDP: $\varepsilon_i(w) := Q(s_i, a_i) - Q^*(s_i, a_i)$. For the current iteration t, the difference between current estimated parameter w and the optimal parameter w^* accumulated along the L length state-action-reward tuples with reverse update is:*

$$w_L - w^* = \underbrace{\Gamma_L (w_1 - w^*)}_{\textit{Bias term}} + \underbrace{\eta \sum_{l=1}^{L} \varepsilon_l \Gamma_{l-1} \phi_l}_{\textit{variance term}} .$$

For clarity, Γ_L in Definition 2 is a joint product of L terms involving the feature vector of the consecutive state-action tuples. When the norm of Γ_L is small, parameter will quickly converge to its optimal.

The first part on RHS is noted as the bias, which decays geometrically with N and the second part on RHS is variance along the sub-trajectory of length L, which we will later show with zero mean.

The proof is presented in Appendix C.1. The result is obtained by unrolling the terms for consecutive L steps in reverse update order. This allows us to separately quantify the upper bound the bias term and the variance terms.

Lemma 4 (Bound on the bias term)**.** *Let $\mathbf{x} \in \mathbb{R}^d$ be a non-zero vector and N is the frequency for the target network to be updated. For $\eta \in (0,1), L \in \mathbb{N}$ and $L > 1$, the following matrix's positive semi-definite inequality holds with probability at least $1-\delta$:*

$$\mathbb{E}\left\|\prod_{j=N}^{1}\Gamma_L\mathbf{x}\right\|_\phi^2 \leq \exp\left(-\frac{N(\eta(4-2L)L+L-\eta^2L)}{\kappa}\right)\sqrt{\tfrac{\kappa}{\delta}}\left\|\mathbf{x}\right\|_\phi.$$

The ϕ-based norm is defined in Definition 1.

Sketch of proof. The result is obtained first expand the joint product over $\prod_{j=N}^{i}$ over Γ_L and integrate the result in Theorem 1. The detail proof is presented in Appendix C.2. □

In terms of the bound for the variance term in Lemma 3, even though the term Γ_l is involved in the expression, it turns out we do not need to modify the original proof and thus we follow the result in the original work. The exact statement is presented in the Appendix C.3.

Theorem 2. *For Linear MDP, assume the reward function, as well as the feature, is bounded $R(s,a) \in [0,1]$, $\|\phi(s,a)\|_2 \leq 1$, for all $(s,a) \in \mathcal{S}\times\mathcal{A}$. Let T be the maximum episodes, N be the frequency of the target network update, η be the learning rate and L be the length of sequence for RER described in Algorithm 1. When $\eta \in (0,1), L \geq 1$, with sample complexity*

$$\mathcal{O}\left(\frac{\gamma^{T/N}}{1-\gamma}+\sqrt{\frac{T\kappa}{N\delta(1-\gamma)^4}}\exp\left(-\frac{N(\eta(4-2L)L+L-\eta^2L)}{\kappa}\right)+\sqrt{\frac{\eta\log(\frac{T}{N\delta})}{(1-\gamma)^4}}\right),$$

$\|Q_T(s,a)-Q^*(s,a)\|_\infty \leq \varepsilon$ holds with probability at least $1-\delta$.

Proof of Sketch. we first show the independence sub-trajectories with length L. Then we decompose the error term of Q-value via bias-variance decomposition (in Lemma 3), where the RER method and target network can help to control the variance term using martingale sequences. We show the upper bound of the bias term in Lemma 4 and the upper bound of the variance term in Lemma C.3. Then we summarize the result and offer the final proof in Lemma 6, which leads to the probabilistic bound in this theorem. □

Compared to the original theorem in (Agarwal et al., 2022, Theorem 1), our work offers a tighter upper bound to relax the assumption for the final result to hold. This bridges the gap between the theoretical justification and the empirical MDP evaluation. Further, we hope the new idea of transforming the original problem into a combinatorial counting problem can enlighten other relevant domains.

We acknowledge that the main structure of convergence proof (i.e., Theorem 2) follows the original work. Here, we made contribution to present a cleaner proof pipeline of the proof and also integrate our tighter bound in Theorem 1.

5 Conclusion

In this work, we gave a tighter finite-sample analysis for heuristics which are heavily used in practical Q-learning and showed that seemingly simple modifications can have far-reaching consequences in linear MDP settings. We provide a rigorous analysis that relaxes the limitation on the learning rate and the length of the consecutive tuples. Our key idea is to transform the original problem involving a giant summation into a combinatorial counting problem, which greatly simplifies the whole problem. Finally, we show theoretically that RER converges faster with a larger learning rate η and a longer consecutive sequence L of state-action-reward tuples.

Acknowledgments

We thank all the reviewers for their constructive comments. We also thank Yi Gu for his feedback on the theoretical justification part of this paper. This research was supported by NSF grant CCF-1918327 and DOE – Fusion Energy Science grant: DE-SC0024583.

References

Naman Agarwal, Syomantak Chaudhuri, Prateek Jain, Dheeraj Nagaraj, and Praneeth Netrapalli. Online target q-learning with reverse experience replay: Efficiently finding the optimal policy for linear mdps. In *ICLR*. OpenReview.net, 2022.

R. Ellen Ambrose, Brad E. Pfeiffer, and David J. Foster. Reverse replay of hippocampal place cells is uniquely modulated by changing reward. *Neuron*, 91(5):1124–1136, 2016. ISSN 0896-6273.

Chenjia Bai, Lingxiao Wang, Lei Han, Jianye Hao, Animesh Garg, Peng Liu, and Zhaoran Wang. Principled exploration via optimistic bootstrapping and backward induction. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pp. 577–587. PMLR, 2021.

Dimitri P. Bertsekas and Huizhen Yu. Q-learning and enhanced policy iteration in discounted dynamic programming. *Math. Oper. Res.*, 37(1):66–94, 2012.

Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. Reverse curriculum generation for reinforcement learning. In *CoRL*, volume 78 of *Proceedings of Machine Learning Research*, pp. 482–495. PMLR, 2017.

David J Foster and Matthew A Wilson. Reverse replay of behavioural sequences in hippocampal place cells during the awake state. *Nature*, 440(7084):680–683, 2006.

Anirudh Goyal, Philemon Brakel, William Fedus, Soumye Singhal, Timothy P. Lillicrap, Sergey Levine, Hugo Larochelle, and Yoshua Bengio. Recall traces: Backtracking models for efficient reinforcement learning. In *ICLR*. OpenReview.net, 2019.

Roudy El Haddad. Repeated sums and binomial coefficients. *arXiv preprint arXiv:2102.12391*, 2021.

Hideyoshi Igata, Yuji Ikegaya, and Takuya Sasaki. Prioritized experience replays on a hippocampal predictive map for learning. *Proceedings of the National Academy of Sciences*, 118(1), 2021.

Prateek Jain, Suhas S. Kowshik, Dheeraj Nagaraj, and Praneeth Netrapalli. Streaming linear system identification with reverse experience replay. In *NIPS*, volume 34, pp. 30140–30152. Curran Associates, Inc., 2021.

Aviral Kumar, Abhishek Gupta, and Sergey Levine. Discor: Corrective feedback in reinforcement learning via distribution correction. In *NeurIPS*, 2020.

Thibault Lahire, Matthieu Geist, and Emmanuel Rachelson. Large batch experience replay. In *ICML*, volume 162 of *Proceedings of Machine Learning Research*, pp. 11790–11813. PMLR, 2022.

Su Young Lee, Sung-Ik Choi, and Sae-Young Chung. Sample-efficient deep reinforcement learning via episodic backward update. In *NeurIPS*, pp. 2110–2119, 2019.

Gen Li, Yuting Wei, Yuejie Chi, Yuantao Gu, and Yuxin Chen. Sample complexity of asynchronous q-learning: Sharper analysis and variance reduction. *IEEE Trans. Inf. Theory*, 68(1):448–473, 2022.

Long Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Mach. Learn.*, 8:293–321, 1992.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Dheeraj Nagaraj, Xian Wu, Guy Bresler, Prateek Jain, and Praneeth Netrapalli. Least squares regression with markovian data: Fundamental limits and algorithms. In *NeurIPS*, 2020.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley Series in Probability and Statistics. Wiley, 1994.

Egor Rotinov. Reverse experience replay. *CoRR*, abs/1910.08780, 2019.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *ICLR*, 2016.

Manel Tagorti and Bruno Scherrer. On the rate of convergence and error bounds for lstd (λ). In *ICML*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 1521–1529. JMLR.org, 2015.

Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *AAAI*, pp. 2094–2100. AAAI Press, 2016.

Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press, 2018.

Christopher J. C. H. Watkins and Peter Dayan. Technical note q-learning. *Mach. Learn.*, 8:279–292, 1992.

Christopher John Cornish Hellaby Watkins. Learning from delayed rewards. *PhD thesis, King's College, University of Cambridge*, 1989.

Andrea Zanette, Alessandro Lazaric, Mykel J. Kochenderfer, and Emma Brunskill. Learning near optimal policies with low inherent bellman error. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pp. 10978–10989. PMLR, 2020.