A Unified Approach for Multi-step Temporal-Difference Learning with Eligibility Traces in Reinforcement Learning

Long Yang, Minhao Shi, Qian Zheng, Wenjia Meng, and **Gang Pan**[*]
College of Computer Science and Technology, Zhejiang University, China
{yanglong,minhaowill,qianzheng,mengwenjia,gpan}@zju.edu.cn

Abstract

Recently, a new multi-step temporal learning algorithm $Q(\sigma)$ unifies n-step Tree-Backup (when $\sigma = 0$) and n-step Sarsa (when $\sigma = 1$) by introducing a sampling parameter σ. However, similar to other multi-step temporal-difference learning algorithms, $Q(\sigma)$ needs much memory consumption and computation time. Eligibility trace is an important mechanism to transform the off-line updates into efficient on-line ones which consume less memory and computation time. In this paper, we combine the original $Q(\sigma)$ with eligibility traces and propose a new algorithm, called $Q^\pi(\sigma,\lambda)$, where λ is trace-decay parameter. This new algorithm unifies Sarsa(λ) (when $\sigma = 1$) and $Q^\pi(\lambda)$ (when $\sigma = 0$). Furthermore, we give an upper error bound of $Q^\pi(\sigma,\lambda)$ *policy evaluation* algorithm. We prove that $Q^\pi(\sigma,\lambda)$ *control* algorithm converges to the optimal value function exponentially. We also empirically compare it with conventional temporal-difference learning methods. Results show that, with an intermediate value of σ, $Q^\pi(\sigma,\lambda)$ creates a mixture of the existing algorithms which learn the optimal value significantly faster than the extreme end ($\sigma = 0$, or 1).

1 Introduction

In reinforcement learning (RL), experiences are sequences of states, actions and rewards which are generated by the agent interacts with environment. The agent's goal is learning from experiences and seeking an optimal policy from the delayed reward decision system. There are two fundamental mechanisms of learning have been studied in RL, one is temporal-difference (TD) learning method which is a combination of Monte Carlo method and dynamic programming [Sutton, 1988]. The other one is eligibility trace [Sutton, 1984; Watkins, 1989] which is a short-term memory process as a function of states. TD(λ) combines TD learning with eligibility trace and unifies one-step learning ($\lambda = 0$) and Monte Carlo methods ($\lambda = 1$) through the trace-decay parameter λ [Sutton, 1988]. Results show that the unified algorithm TD(λ) performs better at an intermediate value of λ rather than either extreme case.

Recently, Multi-step $Q(\sigma)$ [Sutton and Barto, 2017] unifies n-step Sarsa ($\sigma = 1$, *full-sampling*) and n-step Tree-backup ($\sigma = 0$, *pure-expectation*). For some intermediate value $\sigma(0 < \sigma < 1)$, $Q(\sigma)$ creates a mixture of full-sampling and pure-expectation approach, can perform better than the extreme case $\sigma = 0$ or 1 [De Asis *et al.*, 2018].

The results in [De Asis *et al.*, 2018] implies a fundamental trade-off problem in reinforcement learning: should one estimates the value function by adopting pure-expectation ($\sigma = 0$) algorithm or full-sampling ($\sigma = 1$) algorithm? Although pure-expectation approach has lower variance, it needs more complexity and larger calculation [Van Seijen *et al.*, 2009]. On the other hand, full-sampling algorithm needs smaller calculation time, however, it may have a worse asymptotic performance [De Asis *et al.*, 2018]. Multi-step $Q(\sigma)$ [Sutton and Barto, 2017] firstly attempts to combine pure-expectation with full-sample algorithms, however, multi-step temporal-difference learning is too expensive in complexity of time and space during its training.

Eligibility trace is an effective way to address the above complexity problem. In this paper, we try to combine the $Q(\sigma)$ algorithm with eligibility trace, and create a new algorithm, called $Q^\pi(\sigma,\lambda)$. The proposed $Q^\pi(\sigma,\lambda)$ unifies the Sarsa(λ) algorithm [Rummery and Niranjan, 1994] and $Q^\pi(\lambda)$ algorithm [Harutyunyan, 2016]. When σ varies from 0 to 1, $Q^\pi(\sigma,\lambda)$ changes continuously from Sarsa(λ) ($\sigma = 1$ in $Q^\pi(\sigma,\lambda)$) to $Q^\pi(\lambda)$ ($\sigma = 0$ in $Q^\pi(\sigma,\lambda)$). We also focus on the trade-off between pure-expectation and full-sample in *control task*, our experiments show that an intermediate value σ achieves a better performance than either extreme case.

Our contributions are summarised as follows: We define a *mixed-sampling operator* through which we derive the proposed $Q^\pi(\sigma,\lambda)$ policy evaluation algorithm and control algorithm. The new algorithm $Q^\pi(\sigma,\lambda)$ unifies Sarsa(λ) and $Q^\pi(\lambda)$. For $Q^\pi(\sigma,\lambda)$ policy evaluation algorithm, we give its upper error bound. For the control problem, we prove that both of the off-line and on-line $Q^\pi(\sigma,\lambda)$ algorithm converge to the optimal value function.

2 Framework and Notation

The standard episodic reinforcement learning framework [Sutton and Barto, 2017] is often formalized as *Markov deci-*

[*]Corresponding author.

sion processes (MDPs). Such framework considers 5-tuples form $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$, where $\mathcal{S}$ indicates the set of all states, $\mathcal{A}$ indicates the set of all actions, $P^a_{ss'}$ is the element of $\mathcal{P}$ and it indicates a state-transition probability from state s to state s' under taking action a, $a \in \mathcal{A}, s', s \in \mathcal{S}$; $R^a_{ss'}$ is the element of $\mathcal{R}$ and it indicates the expected reward for a transition, $\gamma \in [0, 1]$ is the discount factor. A *policy* π is a probability distribution on $\mathcal{S} \times \mathcal{A}$ and *stationary policy* is a policy that does not change over time. In this paper, we denote $\{(S_t, A_t, R_t)\}_{t \geq 0}$ as a *trajectory* of the state-reward sequence in one episode.

state-action value q maps on $\mathcal{S} \times \mathcal{A}$ to $\mathbb{R}$, for a given policy π, has a corresponding state-action value:

$$q^\pi(s, a) = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t R_t | S_0 = s, A_0 = a].$$

Optimal state-action value is defined as:

$$q^*(s, a) = \max_\pi q^\pi(s, a).$$

Bellman operator $\mathcal{T}^\pi$:

$$\mathcal{T}^\pi q = \mathcal{R}^\pi + \gamma \mathcal{P}^\pi q, \tag{1}$$

Bellman optimal operator $\mathcal{T}^*$:

$$\mathcal{T}^* q = \max_\pi (\mathcal{R}^\pi + \gamma \mathcal{P}^\pi q), \tag{2}$$

where $\mathcal{P}^\pi \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{S}|}$, $\mathcal{R}^\pi \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{A}|}$, and the corresponding entry is:

$$\mathcal{P}^\pi_{ss'} = \sum_{a \in \mathcal{A}} \pi(s, a) P^a_{ss'}, \mathcal{R}^\pi(s, a) = \pi(s, a) \sum_{s' \in \mathcal{S}} P^a_{ss'} R^a_{ss'}.$$

Value function q^π and q^* satisfy the following *Bellman equation* and *optimal Bellman equation* correspondingly:

$$\mathcal{T}^\pi q^\pi = q^\pi, \;\; \mathcal{T}^* q^* = q^*.$$

Both $\mathcal{T}^\pi$ and $\mathcal{T}^*$ are γ-contraction operator with sup-norm, that is to say, $\|\mathcal{T}Q_1 - \mathcal{T}Q_2\|_\infty \leq \gamma \|Q_1 - Q_2\|_\infty$ for any $Q_1, Q_2 \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{A}|}$, where $\mathcal{T} = \mathcal{T}^\pi$ or $\mathcal{T}^*$. Based on the fact that the fixed point of a contraction operator is unique, then *value iteration* converges: $(\mathcal{T}^\pi)^n Q \to q^\pi$, $(\mathcal{T}^*)^n Q \to q^*$, as $n \to \infty$, for any initial Q.

2.1 One-step TD Learning Algorithms

Unfortunately, both the system (1) and (2) can not be solved directly because of fact that the $\mathcal{P}$ and $\mathcal{R}$ in the environment are usually unknown or the dimension of $\mathcal{S}$ is huge in practice. A unavailable model in reinforcement learning is called *model free*.

TD learning algorithm [Sutton, 1984; Sutton, 1988] is one of the most significant algorithms in model free reinforcement learning, the idea of *bootstrapping* is critical to TD learning: the evluation of the value function are used as targets during the learning process.

In reinforcement learning, *target policy* μ is the policy being learned about , and the policy μ used to generate trajectory $\{(S_t, A_t, R_t)\}_{t \geq 0}$ is called the *behavior policy*. If $\pi = \mu$, the learning is called *on-policy* learning, otherwise it is *off-policy* learning.

Sarsa: For a given sample transition (S, A, R, S', A'), *Sarsa* [Rummery and Niranjan, 1994] is a on-policy learning algorithm and it updates Q value as follows:

$$Q_{k+1}(S, A) = Q_k(S, A) + \alpha_k \delta^S_k, \tag{3}$$

$$\delta^S_k = R + \gamma Q_k(S', A') - Q_k(S, A), \tag{4}$$

where δ^S_k is the *k*-th TD error, α_k is *stepsize*.

Expected-Sarsa: *Expected-Sarsa* [Van Seijen *et al.*, 2009] uses expectation of all the next state-action value pairs according to the target policy π to estimate Q value as follows:

$$\begin{aligned} Q_{k+1}(S, A) &= Q_k(S, A) + \alpha_k \delta^{ES}_k, \\ \delta^{ES}_k &= R + \gamma \mathbb{E}_\pi [Q_k(S', \cdot)] - Q_k(S, A), \qquad (5) \\ &= R + \sum_{a \in \mathcal{A}} \pi(S', a) Q_k(S', a) - Q_k(S, A), \end{aligned}$$

where δ^{ES}_k is the *k*-th *expected TD error*. Expected-Sarsa is a off-policy learning algorithm if $\mu \neq \pi$, for example, when π is greedy with respect to Q then Expected-Sarsa is restricted to *Q-Learning* [Watkins, 1989]. If the trajectory is generated by π, Expected-Sarsa is a on-policy algorithm [Van Seijen *et al.*, 2009].

The above two algorithms are guaranteed convergence under some conditions [Singh *et al.*, 2000; Van Seijen *et al.*, 2009].

Q(σ): One-step $Q(\sigma)$ [Sutton and Barto, 2017; De Asis *et al.*, 2018] is a weighted average between the Sarsa update and Expected Sarsa update through a sampling parameter σ:

$$\begin{aligned} Q_{k+1}(S, A) &= Q_k(S, A) + \alpha_k \delta^\sigma_k, \\ \delta^\sigma_k &= \sigma \delta^S_k + (1 - \sigma) \delta^{ES}_k, \end{aligned} \tag{6}$$

where $\sigma \in [0, 1]$ is *degree of sampling*. When $\sigma = 1$, $Q(\sigma)$ denotes a *full-sampling*, and $\sigma = 0$, $Q(\sigma)$ denotes a *pure-expectation* with no sampling. δ^S_t and δ^{ES}_t are defined in (4) and (5).

2.2 λ-Return Algorithm

One-step TD learning algorithm can be generalized to multi-step bootstrapping learning method. The λ-*return* algorithm [Watkins, 1989] is a particular way to mix many multi-step TD learning algorithms through weighting *n-step returns* proportionally to λ^{n-1}.

λ-*operator*[1] $\mathcal{T}^\pi_\lambda$ is a flexible way to express λ-return algorithm, for a given trajectory $\{(S_t, A_t, R_t)\}_{t \geq 0}$ that is generated by policy π,

$$\begin{aligned} (\mathcal{T}^\pi_\lambda q)(s, a) &= \Big\{(1 - \lambda) \sum_{n=0}^{\infty} \lambda^n (\mathcal{T}^\pi)^{n+1} q\Big\}(s, a) \\ &= \sum_{n=0}^{\infty} (1 - \lambda) \lambda^n \mathbb{E}_\pi [G_n | S_0 = s, A_0 = a] \\ &= q(s, a) + \sum_{n=0}^{\infty} (\lambda\gamma)^n \mathbb{E}_\pi [\delta_n], \end{aligned}$$

[1]The notation is coincident with [Bertsekas *et al.*, 2012].

where $G_n = \sum_{t=0}^{n} \gamma^t R_t + \gamma^n Q(S_{n+1}, A_{n+1})$ is *n-step returns* from initial state-action pair (S_0, A_0), the term $\sum_{n=0}^{\infty}(1-\lambda)\lambda^n G_n$ is called λ-*returns*, and $\delta_n = R_n + \gamma Q(S_{n+1}, A_{n+1}) - Q(S_n, A_n)$.

Based on the fact that q^π is fixed point of $\mathcal{T}^\pi$, thus q^π remains the fixed point of $\mathcal{T}^\pi_\lambda$. When $\lambda = 0$, $\mathcal{T}^\pi_\lambda$ is equal to the usual Bellman operator $\mathcal{T}^\pi$. When $\lambda = 1$, the evaluation iteration $Q_{k+1} = \mathcal{T}^\pi_\lambda|_{\lambda=1} Q_k$ becomes *Monte Carlo* method. It is well-known that λ trades off the bias of the bootstrapping with an approximate q^π, with the variance of sampling multi-step returns estimation [Kearns and Singh, 2000]. In practice, a high and intermediate value of λ could be typically better [Singh and Dayan, 1998; Sutton, 1996].

3 Mixed-sampling Operator

In this section, we present the *mixed-sampling operator* $\mathcal{T}^{\pi,\mu}_\sigma$, which is one of our key contribution and is flexible to analysis our new algorithm later. By introducing a sampling parameter $\sigma \in [0, 1]$, the mixed-sampling operator varies continuously from pure-expectation method to full-sampling method. In this section, we analysis the contraction of $\mathcal{T}^{\pi,\mu}_\sigma$ firstly. Then we introduce the λ-return vision of mixed-sampling operator $\mathcal{T}^{\pi,\mu}_{\sigma,\lambda}$. Finally, we give a upper error bound of the corresponding policy evaluation algorithm.

3.1 Contraction of Mixed-sampling Operator

Definition 1. *Mixed-sampling operator $\mathcal{T}^{\pi,\mu}_\sigma$ is a map on $\mathbb{R}^{|\mathcal{S}|\times|\mathcal{A}|}$ to $\mathbb{R}^{|\mathcal{S}|\times|\mathcal{A}|}$, $\forall s \in \mathcal{S}, a \in \mathcal{A}, \sigma \in [0, 1]$:*

$$\mathcal{T}^{\pi,\mu}_\sigma : \mathbb{R}^{|\mathcal{S}|\times|\mathcal{A}|} \to \mathbb{R}^{|\mathcal{S}|\times|\mathcal{A}|}$$

$$q(s,a) \mapsto q(s,a) + \mathbb{E}_\mu \sum_{t=0}^{\infty} \Big[\gamma^t \delta_t^{\pi,\sigma}\Big] \tag{7}$$

where

$$\begin{aligned}\delta_t^{\pi,\sigma} \;&=\; \sigma\Big(R_t + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)\Big)\\ &\quad +(1-\sigma)\Big(R_t + \gamma \mathbb{E}_\pi Q(S_{t+1}, \cdot) - Q(S_t, A_t)\Big)\end{aligned}$$

$$\mathbb{E}_\pi Q(S_{t+1}, \cdot) = \sum_{a\in\mathcal{A}} \pi(S_{t+1}, a) Q(S_{t+1}, a).$$

The parameter σ is also *degree* of sampling that is introduced by the $Q(\sigma)$ algorithm [De Asis *et al.*, 2018]. In one of extreme end ($\sigma = 0$, pure-expectation), $\mathcal{T}^{\pi,\mu}_{\sigma=0}$ deduces the n-step returns G^π_n in $Q^\pi(\lambda)$ [Harutyunyan, 2016], where $G^\pi_n = \sum_{k=t}^{t+n} \gamma^{k-t}\delta_k^{ES} + \gamma^{n+1}\mathbb{E}_\pi Q(S_{t+n+1}, \cdot)$, δ_k^{ES} is the k-th expected TD error. *Multi-step Sarsa* [Sutton and Barto, 2017] is in another extreme end ($\sigma = 1$, full-sampling). Every intermediate value σ creates a mixed method varies continuously from pure-expectation to full-sampling which is why we call $\mathcal{T}^{\pi,\mu}_\sigma$ mixed-sample operator.

λ-**Return Version** We now define the λ-version of $\mathcal{T}^{\pi,\mu}_\sigma$, and denote it as $\mathcal{T}^{\pi,\mu}_{\sigma,\lambda}$:

$$\mathcal{T}^{\pi,\mu}_{\sigma,\lambda} q(s,a) = q(s,a) + \mathbb{E}_\mu[\sum_{t=0}^{\infty} (\lambda\gamma)^t \delta_t^{\pi,\sigma}] \tag{8}$$

where the λ is the parameter takes the from TD(0) to Monte Carlo version as usual. When $\sigma = 0$, $\mathcal{T}^{\pi,\mu}_{\sigma=0,\lambda}$ is restricted to $\mathcal{T}^{\pi,\mu}_\lambda$ [Harutyunyan, 2016], when $\sigma = 1$, $\mathcal{T}^{\pi,\mu}_{\sigma=1,\lambda}$ is restricted to λ-operator.

The next theorem provides a basic property of $\mathcal{T}^{\pi,\mu}_{\sigma,\lambda}$.

Theorem 1. *The operator $\mathcal{T}^{\pi,\mu}_{\sigma,\lambda}$ is a γ-contraction: for any q_1, q_2,*

$$\|\mathcal{T}^{\pi,\mu}_{\sigma,\lambda} q_1 - \mathcal{T}^{\pi,\mu}_{\sigma,\lambda} q_2\|_\infty \le \gamma \|q_1 - q_2\|_\infty.$$

Furthermore, for any initial Q_0, the sequence $\{Q\}_{k=0}^{\infty}$ is generated by the iteration

$$Q_{k+1} = \mathcal{T}^{\pi,\mu}_{\sigma,\lambda} Q_k,$$

then $\{Q_k\}_{k=0}^{\infty}$ converges to the unique fixed point of $\mathcal{T}^{\pi,\mu}_{\sigma,\lambda}$.

Proof. Unfolding the operator $\mathcal{T}^{\pi,\mu}_{\sigma,\lambda}$, we have

$$\begin{aligned}\mathcal{T}^{\pi,\mu}_{\sigma,\lambda} q =&\sigma \underbrace{\Big(q + B[\mathcal{T}^\mu q - q]\Big)}_{\mathcal{T}^\mu_\lambda q}\\ &+ (1-\sigma)\underbrace{\Big(q + B[\mathcal{T}^\pi q - q]\Big)}_{\mathcal{T}^{\pi,\mu}_\lambda q},\end{aligned} \tag{9}$$

where $B = (I - \gamma\lambda\mathcal{P}^\mu)^{-1}$. Based the fact that both $\mathcal{T}^\mu_\lambda$ [Bertsekas *et al.*, 2012] and $\mathcal{T}^{\pi,\mu}_\lambda$ [Harutyunyan, 2016; Munos *et al.*, 2016] are γ-contraction operators, and $\mathcal{T}^{\pi,\mu}_{\sigma,\lambda}$ is the convex combination of above operators, thus $\mathcal{T}^{\pi,\mu}_{\sigma,\lambda}$ is a γ-contraction. □

3.2 Upper Error Bound of Policy Evaluation

In this section we discuss the ability of policy evaluation iteration $Q_{k+1} = \mathcal{T}^{\pi,\mu}_{\sigma,\lambda} Q_k$ in Theorem 1. Our results show that when μ and π are sufficiently close, the ability of the policy evaluation iteration increases gradually as the σ decreases from 1 to 0.

Lemma 1. *If a positive sequence $\{a_k\}_{k=1}^{\infty}$ satisfies $a_{k+1} \le \alpha a_k + \beta$, then for any $|\alpha| < 1$, we have*

$$a_k - \frac{\beta}{1-\alpha} \le \alpha^k (a_1 - \frac{\beta}{1-\alpha}).$$

Furthermore, for any $\epsilon > 0$, $\exists K, s.t, \forall k > K$ has the following estimation

$$|a_k| \le \frac{\beta}{1-\alpha} + \epsilon.$$

Theorem 2 (Upper error bound of policy evaluation)**.** *Consider the policy evaluation algorithm $Q_{k+1} = \mathcal{T}^{\pi,\mu}_{\sigma,\lambda} Q_k$, if the behavior policy μ is ϵ-away from the target policy π, in the sense that $\max_{s\in\mathcal{S}} \|\pi(s,a) - \mu(s,a)\|_1 \le \epsilon$, $\epsilon < \frac{1-\gamma}{\lambda\gamma}$, and $\gamma(1+2\lambda) < 1$, then $\exists K, s.t, \forall k > K$, the policy evaluation sequence$\{Q_k\}_{k=0}^{\infty}$ satisfies*

$$\|Q_{k+1} - q^\pi\|_\infty \le \sigma\epsilon\Big[\frac{M + \gamma C_\pi}{1 - \gamma(1+2\lambda)} + 1\Big],$$

where for a given policy π, M, C_π is determined by the learning system.

Proof. Firstly, we provide an equation which could be used later:

$$\begin{aligned}
& q + B[\mathcal{T}^{\mu} q - q] - q^{\pi} \\
&= B[(I - \gamma\lambda\mathcal{P}^{\mu})(q - q^{\pi}) + \mathcal{T}^{\mu} q - q] \\
&= B[-\mathcal{T}^{\pi} q^{\pi} + \mathcal{T}^{\mu} q^{\pi} - \mathcal{T}^{\mu} q^{\pi} + \mathcal{T}^{\mu} q + \gamma\lambda\mathcal{P}^{\mu}(q^{\pi} - q)] \\
&= B\Big[\underbrace{\mathcal{R}^{\mu} - \mathcal{R}^{\pi} + \gamma(\mathcal{P}^{\mu} - \mathcal{P}^{\pi})q^{\pi}}_{-\mathcal{T}^{\pi} q^{\pi} + \mathcal{T}^{\mu} q^{\pi}} + \underbrace{\gamma\mathcal{P}^{\mu}(q - q^{\pi})}_{-\mathcal{T}^{\mu} q^{\pi} + \mathcal{T}^{\mu} q} \\
&+ \gamma\lambda\mathcal{P}^{\mu}(q^{\pi} - q)\Big]. \qquad (10)
\end{aligned}$$

Rewrite the policy evaluation iteration:

$$Q_{k+1} = \sigma\mathcal{T}^{\mu}_{\lambda} Q_k + (1-\sigma)\mathcal{T}^{\pi,\mu}_{\lambda} Q_k.$$

Note q^{π} is fixed point of $\mathcal{T}^{\pi,\mu}_{\lambda}$ [Harutyunyan, 2016], then we merely consider next estimator:

$$\begin{aligned}
& \|Q_{k+1} - q^{\pi}\|_{\infty} \\
= \;& \sigma\|Q_k + B[\mathcal{T}^{\mu} Q_k - Q_k] - q^{\pi}\|_{\infty} \\
\le \;& \sigma\Big[\frac{\epsilon(M + \gamma\|q^{\pi}\|)}{1-\gamma\lambda} + \frac{\gamma(1+\lambda)}{1-\gamma\lambda}\|Q_k - q^{\pi}\|_{\infty}\Big] \\
\overset{\text{Lemma1}}{\le} \;& \sigma\epsilon\Big[\frac{M + \gamma C_{\pi}}{1-\gamma(1+2\lambda)} + 1\Big].
\end{aligned}$$

The first equation is derived by replacing q in (10) with Q_k. Since μ is ϵ-away from π, the first inequality is determined the following fact:

$$\begin{aligned}
\|\mathcal{R}^{\pi}_s - \mathcal{R}^{\mu}_s\|_{\infty} &= \max_{a\in\mathcal{A}}\{|(\pi(s,a) - \mu(s,a))\mathcal{R}^a_s|\} \\
&\le \epsilon|\mathcal{A}|\max_a|\mathcal{R}^a_s| = \epsilon M_s, \\
\|\mathcal{R}^{\pi} - \mathcal{R}^{\mu}\|_{\infty} &\le \epsilon M,
\end{aligned}$$

where $M_s = |\mathcal{A}|\max_a|\mathcal{R}^a_s|$ is determined by the reinforcement learning system and is independent of π, μ. $M = \max_{s\in\mathcal{S}} M_s$. For the given policy π, $\|q^{\pi}\|$ is a constant that is determined by learning system, we denote it C_{π}. □

Remark 1: *The proof in Theorem 2 strictly dependent on the assumption that ϵ is smaller but never to be zero, where the ϵ is a bound of discrepancy between the behavior policy μ and target policy π. That is to say, the ability of the prediction of policy evaluation iteration is dependent on the gap between μ and π. A more profound discussion about this gap is related to λ-ϵ trade off [Harutyunyan,2016] which is beyond the scope in this paper.*

4 $Q^{\pi}(\sigma,\lambda)$ Control Algorithm

In this section, we present $Q^{\pi}(\sigma,\lambda)$ [2] algorithm for control. We analyze the off-line version of $Q^{\pi}(\sigma,\lambda)$ which converges to optimal value function exponentially fast.

Considering the typical iteration (Q_k, π_k), where $\pi_k$is calculated by the following two steps: $\{\mu_k\}_{k\ge0}$ is a sequence of corresponding behavior policies,

*Step*1: *policy evaluation*

$$Q_{k+1} = \mathcal{T}^{\pi_k,\mu_k}_{\sigma,\lambda} Q_k,$$

*Step*2: *policy improvement*

$$\mathcal{T}^{\pi_{k+1}} Q_{k+1} = \mathcal{T}^{*} Q_{k+1},$$

that is π_{k+1} is greedy policy with repect to Q_{k+1}. We call the approach introduced by above step1 and step2 $Q^{\pi}(\sigma,\lambda)$ *control algorithm.*

In the following Theorem 3, we presents the convergence rate of $Q^{\pi}(\sigma,\lambda)$ control algorithm.

Theorem 3 (Convergence of $Q^{\pi}(\sigma,\lambda)$ Control Algorithm)**.** *Considering the sequence $\{(Q_k,\pi_k)\}_{k\ge0}$ generated by the $Q^{\pi}(\sigma,\lambda)$ control algorithm, for the given $\lambda,\gamma\in(0,1)$, then*

$$\|Q_{k+1} - q^{*}\|_{\infty} \le \eta\|Q_k - q^{*}\|_{\infty},$$

where $\eta = \frac{\gamma(1+\lambda-2\lambda\sigma)}{1-\lambda\gamma}$.

Particularly, if $\lambda < \frac{1-\gamma}{2\gamma}$, then $0 < \eta < 1$ and sequence $\{Q_k\}_{k\ge1}$ converges to q^{*} exponentially fast:

$$\|Q_{k+1} - q^{*}\|_{\infty} = O(\eta)^{k+1}.$$

Proof. By the definition of $\mathcal{T}^{\pi,\mu}_{\sigma,\lambda}$,

$$\mathcal{T}^{\pi,\mu}_{\sigma,\lambda} q = \sigma\mathcal{T}^{\mu}_{\lambda} q + (1-\sigma)\mathcal{T}^{\pi,\mu}_{\lambda} q$$

we have[3]:

$$\begin{aligned}
& \|Q_{k+1} - q^{*}\|_{\infty} \\
\le \;& \sigma\|\mathcal{T}^{\mu_k}_{\lambda}(Q_k - q^{*})\|_{\infty} + (1-\sigma)\|\mathcal{T}^{\pi_k,\mu_k}_{\lambda}(Q_k - q^{*})\|_{\infty} \\
\le \;& \Big(\sigma\frac{\gamma(1-\lambda)}{1-\lambda\gamma} + (1-\sigma)\frac{\gamma(1+\lambda)}{1-\lambda\gamma}\Big)\|Q_k - q^{*}\|_{\infty} \\
= \;& \underbrace{\frac{\gamma(1+\lambda-2\lambda\sigma)}{1-\lambda\gamma}}_{\eta}\|Q_k - q^{*}\|_{\infty}
\end{aligned}$$

□

```
Algorithm1:On-line $Q^{\pi}(\sigma,\lambda)$ algorithm
Require:Initialize $Q_0(s,a)$ arbitrarily, $\forall s\in\mathcal{S},\forall a\in\mathcal{A}$
Require:Initialize $\mu_k$ to be the behavior policy
Parameters: step-size $\alpha_t\in(0,1]$
Repeat (for each episode):
  $Z(s,a) = 0\ \forall s\in\mathcal{S},\forall a\in\mathcal{A}$
  $Q_{k+1}(s,a) = Q_k(s,a)\ \forall s\in\mathcal{S},\forall a\in\mathcal{A}$
  Initialize state-action pair $(S_0, A_0)$
  For $t$ = 0 , 1, 2, $\cdots$ $T_k$:
    Obersive a sample $(R_t, S_{t+1}, A_{t+1}) \sim \mu_k$
    $\delta^{\sigma,\pi_k}_t = R_t + \gamma\Big\{(1-\sigma)\mathbb{E}_{\pi_k}[Q_{k+1}(S_{t+1},\cdot)]$
        $+\sigma Q_{k+1}(S_{t+1},A_{t+1})\Big\} - Q_{k+1}(S_t,A_t)$
    For $\forall s\in\mathcal{S},\forall a\in\mathcal{A}$:
      $Z(s,a) = \gamma\lambda Z(s,a)$+$\mathbb{I}\{(S_t,A_t) = (s,a)\}$
      $Q_{k+1}(s,a) = Q_{k+1}(s,a) + \alpha_k\delta^{\sigma,\pi_k}_t Z(s,a)$
    End For
    $S_{t+1} = S_t$, $A_{t+1} = A_t$
    If $S_{t+1}$ is terminal:
      Break
  End For
```

[2] The work in [Dumke, 2017] proposes a similar method, called $Q(\sigma,\lambda)$ control algorithm. Our $Q^{\pi}(\sigma,\lambda)$ is different from theirs due to the different version of eligibility trace updating.

[3]The section inequality is based on the next two results: [Munos *et al.*, 2016] Theorem2 and [Bertsekas *et al.*, 2012] Proposition6.3.10.

5 On-line Implementation of $Q^\pi(\sigma, \lambda)$

We have discussed the contraction property of mixed-sampling operator $\mathcal{T}^{\pi,\mu}_{\sigma,\lambda}$, and via it we have introduced the $Q^\pi(\sigma, \lambda)$ control algorithm. Both of the iterations in Theorem 2 and Theorem 3 are the version of offline. In this section, we give the on-line version of $Q^\pi(\sigma, \lambda)$ and discuss its convergence.

5.1 On-line Learning

Off-line learning is too expensive due to the learning process must be carried out at the end of an episode, however, on-line learning updates value function with a lower computational cost, better performance. There is a simple interpretation of equivalence between off-line learning and on-line learning which means that, by the end of an episode, the total updates of the *forward view*(*off-line learning*) is equal to the total updates of the *backward view*(*on-line learning*) [Sutton and Barto, 1998]. By the view of equivalence[4], on-line learning can be seen as an implementation of offline algorithm in an inexpensive manner. Another interpretation of on-line learning was provided by [Singh and Sutton, 1996], TD learning with accumulate trace comes to approximate *every-visit Monte-Carlo* method and TD learning with replace trace comes to approximate *first-visit Monte-Carlo* method.

The iterations in Theorem 2 and Theorem 3 are the version of expectations. In practice, we can only access to the trajectory $\{(S_t, A_t, R_t)\}_{t\ge 0}$. By statistical approaches, we can utilize the trajectory to estimate the value function. Algorithm 1 corresponds to online form of $Q^\pi(\sigma, \lambda)$.

5.2 On-line Learning Convergence Analysis

We make some common assumptions which are similar to [Bertsekas and Tsitsiklis, 1996; Harutyunyan, 2016].

Assumption 1. $\sum_{t\ge 0} P[(S_t, A_t) = (s, a)] > 0$, *minimum visit frequency, which implies a complete exploration.*

Assumption 2. *For every historical chain $\mathcal{F}_t$ in a MDPs, $P[N_t(s,a) \ge k|\mathcal{F}_t] \le \gamma\rho^k$, where ρ is a positive constants, k is a positive integer and $N_t(s,a)$ denotes the times of the pair (s,a) is visited in the t-th trajectory $\mathcal{F}_t$.*

For the convenience of expression, we give some notations firstly. Let $Q^o_{k,t}$ be the vector obtained after t iterations in the k-th trajectory, and the superscript o emphasizes online learning. We denote the k-th trajectory as $\{(S_t, A_t, R_t)\}_{t\ge 0}$ sampled by the policy μ_k. Then the online update rules can be expressed as follows: $\forall\ (s,a) \in \mathcal{S}\times\mathcal{A}$:

$$\begin{aligned} Q^o_{k,0}(s,a) &= Q^o_k(s,a) \\ \delta^o_{k,t} &= R_t + \gamma\Big\{(1-\sigma)\mathbb{E}_{\pi_k}[Q^o_{k,t}(S_{t+1},\cdot)] \\ &\quad +\sigma Q^o_{k,t}(S_{t+1}, A_{t+1})\Big\} - Q^o_{k,t}(S_t, A_t) \\ Q^o_{k,t+1}(s,a) &= Q^o_{k,t}(s,a) + \alpha_k(s,a) z^o_{k,t}(s,a)\delta^o_{k,t} \\ Q^o_{k+1}(s,a) &= Q^o_{k,T_k}(s,a), \end{aligned}$$

where T_k is the length of the k-th trajectory.

[4]The true online learning was firstly introduced by [Seijen and Sutton, 2014], more details in [Van Seijen *et al.*, 2016].

Theorem 4. *Based on the Assumption 1 and Assumption 2, step-size α_t satisfying, $\sum_t \alpha_t = \infty, \sum_t \alpha_t^2 < \infty$, π_k is greedy with respect to Q^o_k, then $Q^o_k \xrightarrow{w.p.1} q^*$ as $k \to \infty$, where $w.p.1$ is short for with probability one.*

Proof. After some simple algebra:

$$\begin{aligned} & Q^o_{k+1}(s,a) \\ = \ & Q^o_k(s,a) + \widetilde{\alpha_k}\Big(G^o_k(s,a) - \frac{N_k(s,a)}{\mathbb{E}[N_k(s,a)]}Q^o_k(s,a)\Big), \\ & G^o_k(s,a) \\ = \ & \frac{1}{\mathbb{E}[N_k(s,a)]}\sum_{t=0}^{T_k} z^o_{k,t}(s,a)\delta^o_{k,t}, \end{aligned}$$

where $\widetilde{\alpha_k} = \mathbb{E}[N_k(s,a)]\alpha_k(s,a)$.

We rewrite the off-line update:

$$\begin{aligned} & Q^f_{k+1}(s,a) \\ = \ & Q^f_k(s,a) + \alpha_k\Big(G^f_k(s,a) - \frac{N_k(s,a)}{\mathbb{E}[N_k(s,a)]}Q^f_k(s,a)\Big), \end{aligned}$$

$$G^f_k = \frac{1}{\mathbb{E}[N_k(s,a)]}\sum_{t=0}^{N_k(s,a)} Q^\lambda_{k,t}(s,a),$$

where $Q^\lambda_{k,t}(s,a)$ is the λ-returns at time t when the pair (s,a) is visited in the k-th trajectory, the superscript f in $Q^f_{k+1}(s,a)$ emphasizes the forward (off-line) update. $N_k(s,a)$ denotes the times of the pair (s,a) visited in the k-th trajectory. We define $Res_k(s,a)$ as the *residual* between on-line estimate G^o_k and off-line estimate $G^f_k(s,a)$ in the k-th trajectory:

$$Res_k(s,a) = Q^o_k(s,a) - Q^f_k(s,a).$$

Set $\Delta_k(s,a) = Q^o_k(s,a) - q^*(s,a)$ which is the error between after k-episode the learned value function Q^o_k and optimal value function q^*, then we consider the next random iterative process:

$$\Delta_{k+1}(s,a) = (1-\hat{\alpha_k}(s,a))\Delta_k(s,a) + \hat{\beta}_k F_k(s,a), \quad (11)$$

where

$$\hat{\alpha_k}(s,a) = \frac{N_k(s,a)\alpha_k(s,a)}{\mathbb{E}_{\mu_k}[N_k(s,a)]}, \hat{\beta}_k(s,a) = \alpha_k(s,a),$$

$$F_k(s,a) = G^o_k(s,a) - \frac{N_k(s,a)}{\mathbb{E}_{\mu_k}[N_k(s,a)]}q^*(s,a).$$

Step1:Upper bound on $Res_k(s,a)$:

$$\max_{(s,a)}\Big|\mathbb{E}_{\mu_k}[Res_k(s,a)]\Big| \le C_k \max_{(s,a)}\Big|Q^o_{k+1}(s,a) - q^*\Big|, \quad (12)$$

where $C_k \xrightarrow{w.p.1} 0$.

$$\begin{aligned} & Res_{k,t}(s,a) \\ = \ & \frac{1}{\mathbb{E}[N_k(s,a)]}\sum_{m=0}^{t}\Big[z^o_{k,m}(s,a)\delta^o_{k,m} - Q^\lambda_{k,m}(s,a)\Big], \end{aligned}$$

where $0 \le t \le T_k$. $Res_{k,t}(s,a)$ is the difference between the total on-line updates of first t steps and the first t times off-line update in k-th trajectory. By induction on t, we have:

$$\|Res_{k,t+1}(s,a)\| \le \alpha_M C(\Delta_{k,t} + \|Res_{k,t}(s,a)\|),$$

where C is a consist and $\Delta_{k,t} = \|Q^o_{k,t}(s,a) - q^*(s,a)\|$, $\alpha_M = \max_{0\le t\le T_k}\{\alpha_t(s,a)\}$. Based on the condition of step-size in the Theorem 4, $\alpha_M \xrightarrow{w.p.1} 0$, then we have (12).

Step2: $\max_{(s,a)} \mathbb{E}_{\mu_k}[F_k(s,a)] \le \gamma \max_{(s,a)} \|\Delta_k(s,a)\|$.

In fact:

$$\begin{aligned} F_k(s,a) &= G^f_k(s,a) + Res_k(s,a) \\ &\quad - \frac{N_k(s,a)}{\mathbb{E}_{\mu_k}[N_k(s,a)]} q^*(s,a), \\ \mathbb{E}_{\mu_k}[F_k(s,a)] &= \frac{\sum_{t=0}^{N_k(s,a)} \mathbb{E}_{\mu_k}[Q^\lambda_{k,t}(s,a) - q^*]}{\mathbb{E}_{\mu_k}[N_k(s,a)]} \\ &\quad + Res_k(s,a). \end{aligned}$$

From the property of eligibility trace(more details refer to [Bertsekas *et al.*, 2012]) and Assumption 2, we have:

$$\begin{aligned} &\left|\mathbb{E}_{\mu_k}[Q^\lambda_{k,t}(s,a) - q^*]\right| \\ \le\ & P[N_k(s,a) \ge t]\mathbb{E}_{\mu_k}\left|Q^\lambda_k(s,a) - q^*\right| \\ \le\ & \gamma\rho^t \max_{(s,a)} |Q^f_k(s,a) - q^*|. \end{aligned}$$

Then according to (11), for some $t > 0$:

$$\begin{aligned} \mathbb{E}_{\mu_k}[F_k(s,a)] &\le (\gamma\rho^t + \alpha_t) \max_{(s,a)} \|\Delta_k(s,a) \\ &\le \gamma \max_{(s,a)} \|\Delta_k(s,a)\|. \end{aligned}$$

Step3: $Q^o_k \xrightarrow{w.p.1} q^*$ as $k \to \infty$.
Considering the iteration (11) and Theorem 1 in [Jaakkola *et al.*, 1994], then we have $Q^o_k \xrightarrow{w.p.1} q^*$. □

6 Experiments

6.1 Experiment for Prediction Capability

In this section, we test the prediction abilities of $Q^\pi(\sigma,\lambda)$ in *19-state random walk* environment [De Asis *et al.*, 2018]. The agent at each state has two action : *left* and *right*, and taking each action with equal probability. We compare the root-mean-square(RMS) error as a function of episodes, σ varies dynamically σ from 0 to 1 with steps of 0.2. In this experiment, we set behavior policy μ=[0.5-D,0.5+D], $D \sim U$[-0.01,0.01], where U is uniform distribution.

Results in Figure 1 show that the performance of $Q^\pi(\sigma,\lambda)$ increases gradually as the σ decreases from 1 to 0, which just verifies the upper error bound in Theorem2.

6.2 Experiment for Control Capability

We test the control capability of $Q^\pi(\sigma,\lambda)$ in the classical episodic task, *mountain car* [Sutton and Barto, 1998]. Because the state space in this environment is continuous, we use *tile coding* function approximation [Sutton and Barto, 1998], and use the version 3 of Sutton's tile coding software (n.d.) with 8 tilings.

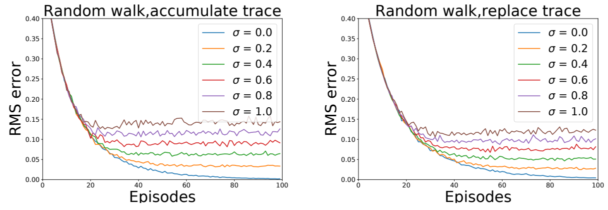

Figure 1: Root-mean-square(RMS) error of state values as a function of episodes in 19-state random walk, we consider both accumulating trace and replacing trace. The plot shows the prediction ability of $Q(\sigma,\lambda)$ varying σ, when λ is fixed to 0.8,$\gamma = 1$.

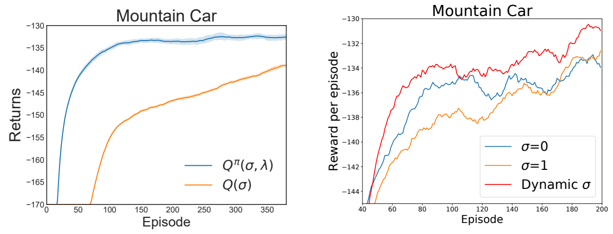

Figure 2: The plot shows the the average returns per episode. A right-centered moving average with a window of 20 successive episodes was employed in order to smooth the results. The right plot shows the result of $Q^\pi(\sigma,\lambda)$ comparing with $Q(\sigma)$. The left plot shows the result of $Q^\pi(\sigma,\lambda)$ comparing with $Q^\pi(\sigma)$ ($\sigma = 0$) and Sarsa(λ) ($\sigma = 1$). $\gamma = 0.99$, step-size $\alpha = 0.3$.

In the right part of Figure 2, we collect the data by varing σ from 0 to 1 with steps of 0.02. Results show that $Q^\pi(\sigma,\lambda)$ significantly converges faster than $Q(\sigma)$. In the left part of Figure 2, results show that the $Q^\pi(\sigma,\lambda)$ with σ in an intermediate value can outperform $Q^\pi(\lambda)$ and Sarsa(λ).

Table1 and Table2 summarize the average return after 50 and 200 episodes. In order to gain more insight into the nature of the results, we run σ from 0 to 1 with steps of 0.02, we take the statistical method according to [De Asis *et al.*, 2018], lower (LB) and upper (UB) 95% confidence interval bounds are provided to validate the results. Dynamic σ in the table mean that σ does not need to remain constant throughout every episode [De Asis et al., 2018].In this experiment, σ is a function of time, a normal distribution, $\sigma_t \sim \mathcal{N}(0.5, 0.1^2)$. The average return after only 50 episodes could be interpreted as a measure of initial performance, whereas the average return after 200 episodes shows how well an algorithm is capable of learning[De Asis *et al.*, 2018]. Results show that $Q^\pi(\sigma,\lambda)$ with a intermediate value had the best final performance.

Algorithm	Mean	UB	LB
$Q^\pi(\sigma=0,\lambda)$,$Q^\pi(\lambda)$	**-193.56**	-179.33	-197.06
$Q^\pi(\sigma=1,\lambda)$,$Sarsa(\lambda)$	-196.84	-182.69	-200.42
$Q^\pi(\sigma=0.5,\lambda)$	-195.99	-181.60	-199.62
Dynamic σ	-195.01	**-177.14**	**-195.01**

Table 1: Average return per episode after 50 Episodes.

Algorithm	Mean	UB	LB
$Q^{\pi}(\sigma=0,\lambda)$,$Q^{\pi}(\lambda)$	-144.04	-138.49	-146.44
$Q^{\pi}(\sigma=1,\lambda)$,$Sarsa(\lambda)$	-145.72	-140.04	-148.21
$Q^{\pi}(\sigma=0.5,\lambda)$	-143.71	-137.92	-146.12
Dynamic σ	**-142.62**	**-137.24**	**-145.09**

Table 2: Average return per episode after 200 episodes.

7 Conclusion

In this paper we presented a new method, called $Q^{\pi}(\sigma,\lambda)$, which unifies Sarsa(λ) and $Q^{\pi}(\lambda)$. We solved a upper error bound of $Q^{\pi}(\sigma,\lambda)$ for the ability of policy evaluation. Furthermore, we discussed the convergence of $Q^{\pi}(\sigma,\lambda)$ control algorithm to q^* under some conditions. The proposed approach was compared with one-step and multi-step TD learning methods and the results demonstrated its effectiveness.

Acknowledgements

This work was partly supported by National Key Research and Development Program of China (2017YFB1002503), Natural Science Foundation of China (No.61772460), and Zhejiang Provincial Natural Science Foundation of China (LR15F020001).

References

[Bertsekas and Tsitsiklis, 1996] Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro Dynamic Programming*. Athena scientific Belmont, MA, 1996.

[Bertsekas *et al.*, 2012] Dimitri P Bertsekas, Dimitri P Bertsekas, Dimitri P Bertsekas, and Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 2. Athena scientific Belmont, MA, 2012.

[De Asis *et al.*, 2018] Kristopher De Asis, J Fernando Hernandez-Garcia, G Zacharias Holland, and Richard S Sutton. Multi-step reinforcement learning: A unifying algorithm. *AAAI*, 2018.

[Dumke, 2017] Markus Dumke. Double q(σ) and q(σ,λ): Unifying reinforcement learning control algorithms. *arXiv preprint arXiv:1711.01569*, 2017.

[Harutyunyan, 2016] Rémi Harutyunyan. Q (λ) with off-policy corrections. In *Algorithmic Learning Theory: 27th International Conference, ALT 2016, Bari, Italy, October 19-21, 2016, Proceedings*, volume 9925, page 305. Springer, 2016.

[Jaakkola *et al.*, 1994] Tommi Jaakkola, Michael I Jordan, and Satinder P Singh. Convergence of stochastic iterative dynamic programming algorithms. In *Advances in neural information processing systems*, pages 703–710, 1994.

[Kearns and Singh, 2000] Michael J Kearns and Satinder P Singh. Bias-variance error bounds for temporal difference updates. In *COLT*, pages 142–147, 2000.

[Munos *et al.*, 2016] Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 1054–1062, 2016.

[Rummery and Niranjan, 1994] Gavin A Rummery and Mahesan Niranjan. *On-line Q-learning using connectionist systems*, volume 37. University of Cambridge, Department of Engineering, 1994.

[Seijen and Sutton, 2014] Harm Seijen and Rich Sutton. True online td (lambda). In *International Conference on Machine Learning*, pages 692–700, 2014.

[Singh and Dayan, 1998] Satinder Singh and Peter Dayan. Analytical mean squared error curves for temporal difference learning. *Machine Learning*, 32(1):5–40, 1998.

[Singh and Sutton, 1996] Satinder P Singh and Richard S Sutton. Reinforcement learning with replacing eligibility traces. *Machine learning*, 22(1-3):123–158, 1996.

[Singh *et al.*, 2000] Satinder Singh, Tommi Jaakkola, Michael L Littman, and Csaba Szepesvári. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine learning*, 38(3):287–308, 2000.

[Sutton and Barto, 1998] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[Sutton and Barto, 2017] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 2017.

[Sutton, 1984] Richard S Sutton. Temporal credit assignment in reinforcement learning. *PhD thesis, University of Massachusetts*, 1984.

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

[Sutton, 1996] Richard S Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. In *Advances in neural information processing systems*, pages 1038–1044, 1996.

[Van Seijen *et al.*, 2009] Harm Van Seijen, Hado Van Hasselt, Shimon Whiteson, and Marco Wiering. A theoretical and empirical analysis of expected sarsa. In *Adaptive Dynamic Programming and Reinforcement Learning, 2009. ADPRL'09. IEEE Symposium on*, pages 177–184. IEEE, 2009.

[Van Seijen *et al.*, 2016] Harm Van Seijen, A Rupam Mahmood, Patrick M Pilarski, Marlos C Machado, and Richard S Sutton. True online temporal-difference learning. *Journal of Machine Learning Research*, 17(145):1–40, 2016.

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