Target-Based Temporal-Difference Learning

Donghwan Lee [1] **Niao He** [2]

Abstract

The use of target networks has been a popular and key component of recent deep Q-learning algorithms for reinforcement learning, yet little is known from the theory side. In this work, we introduce a new family of target-based temporal difference (TD) learning algorithms that maintain two separate learning parameters – the target variable and online variable. We propose three members in the family, the *averaging TD, double TD*, and *periodic TD*, where the target variable is updated through an averaging, symmetric, or periodic fashion, respectively, mirroring those techniques used in deep Q-learning practice. We establish asymptotic convergence analyses for both *averaging TD* and *double TD* and a finite sample analysis for *periodic TD*. In addition, we provide some simulation results showing potentially superior convergence of these target-based TD algorithms compared to the standard TD-learning. While this work focuses on linear function approximation and policy evaluation setting, we consider this as a meaningful step towards the theoretical understanding of deep Q-learning variants with target networks.

1. Introduction

Deep Q-learning (Mnih et al., 2015) has recently captured significant attentions in the reinforcement learning (RL) community for outperforming human in several challenging tasks. Besides the effective use of deep neural networks as function approximators, the success of deep Q-learning is also indispensable to the utilization of a separate target network for calculating target values at each iteration. In practice, using target networks is proven to substantially improve the performance of Q-learning algorithms, and is gradually adopted as a standard technique in modern implementations of Q-learning.

To be more specific, the update of Q-learning with target network can be viewed as follows:

$$\theta_{t+1} = \theta_t + \alpha(y_t - Q(s_t, a_t; \theta_t))\nabla_\theta Q(s_t, a_t; \theta_t)$$

where $y_t = r(s_t, a_t) + \gamma \max_a Q(s_{t+1}, a; \theta'_t)$, θ_t is the online variable, and θ'_t is the target variable. Here the state-action value function $Q(s, a; \theta)$ is parameterized by θ. The update of the online variable θ_t resembles the stochastic gradient descent step. The term $r(s_t, a_t)$ stands for the intermediate reward of taking action a_t in state s_t, and y_t stands for the target value under the target variable, θ'_t. When the target variable is set to be the same as the online variable at each iteration, this reduces to the standard Q-learning algorithm (Watkins & Dayan, 1992), and is known to be unstable with nonlinear function approximations. Several choices of target networks are proposed in the literature to overcome such instability: (i) periodic update, i.e., the target variable is copied from the online variable every $\tau > 0$ steps, as used for deep Q-learning (Gu et al., 2016; Mnih et al., 2015; 2016; Wang et al., 2016); (ii) symmetric update, i.e., the target variable is updated symetrically as the online variable; this is first introduced in double Q-learning (Hasselt, 2010; Van Hasselt et al., 2016); and (iii) Polyak averaging update, i.e., the target variable takes weighted average over the past values of the online variable; this is used in deep deterministic policy gradient (Heess et al., 2015; Lillicrap et al., 2015) as an example. In the following, we simply refer these as target-based Q-learning algorithms.

While the integration of Q-learning with target networks turns out to be successful in practice, its theoretical convergence analysis remains largely an open yet challenging question. As an intermediate step towards the answer, in this work, we first study target-based temporal difference (TD) learning algorithms and establish their convergence analysis. TD algorithms (Sutton, 1988; Sutton et al., 2009a;b) are designed to evaluate a given policy and are the fundamental building blocks of many RL algorithms. Comprehensive surveys and comparisons among TD-based policy evaluation algorithms can be found in (Dann et al., 2014). Motivated by the target-based Q-learning algorithms (Mnih et al., 2015; Wang et al., 2016), we introduce a target variable into the

[1]Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, USA [2]Department of Industrial and Enterprise Systems Engineering, University of Illinois at Urbana-Champaign, USA. Correspondence to: Donghwan Lee <donghwan@illinois.edu>, Niao He <niaohe@illinois.edu>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

TD framework and develop a family of target-based TD algorithms with different updating rules for the target variable. In particular, we propose three members in the family, the *averaging TD, double TD*, and *periodic TD*, where the target variable is updated through an averaging, symmetric or periodic fashion, respectively. Meanwhile, similar to the standard TD-learning, the online variable takes stochastic gradient steps of the Bellman residual loss function while freezing the target variable. As the target variable changes slowly compared to the online variable, target-based TD algorithms are prone to improve the stability of learning especially if large neural networks are used, although this work will focus on TD with linear function approximators.

Theoretically, we prove the asymptotic convergence of *averaging TD* and *double TD* and establish a finite sample analysis for the *periodic TD*. Practically, we also run some simulations showing superior convergence of the proposed target-based TD algorithms compared to the standard TD-learning. In particular, our empirical case studies demonstrate that the target TD-learning algorithms outperform the standard TD-learning in the long run with better accuracy and lower variances, despite their slower convergence at the very beginning. Moreover, our analysis reveals an important connection between the TD-learning and the target-based TD-learning. We consider the work as a meaningful step towards the theoretical understanding of deep Q-learning with general nonlinear function approximation.

Related work. The first target-based reinforcement learning was proposed in (Mnih et al., 2015) for policy optimization problems with nonlinear function approximation, where only empirical results were given. To our best knowledge, target-based reinforcement learning for policy evaluation has not been specifically studied before. A somewhat related family of algorithms are the gradient TD (GTD) learning algorithms (Dai et al., 2017; Mahadevan et al., 2014; Sutton et al., 2009a;b), which minimize the projected Bellman residual through the primal-dual algorithms. The GTD algorithms share some similarities with the proposed target-based TD-learning algorithms in that they also maintain two separate variables – the primal and dual variables, to minimize the objective. Apart from this connection, the GTD algorithms are fundamentally different from the averaging TD and double TD algorithms that we propose. The proposed periodic TD algorithm can be viewed as approximately solving least squares problems across cycles, making it closely related to two families of algorithms, the least-square TD (LSTD) learning algorithms (Bertsekas, 1995; Bradtke & Barto, 1996) and the least squares policy evaluation (LSPE) (Bertsekas & Yu, 2009; Yu & Bertsekas, 2009). But they also distinct from each other in terms of the subproblems and subroutines used in the algorithms. Particularly, the periodic TD executes stochastic gradient descent steps while LSTD uses the least-square parameter estimation method to minimize the projected Bellman residual. On the other hand, LSPE directly solves the subproblems without successive projected Bellman operator iterations. Moreover, the proposed periodic TD algorithm enjoys a simple finite-sample analysis based on existing results on stochastic approximation.

2. Preliminaries

In this section, we briefly review the basics of the TD-learning algorithm with linear function approximation. We first list a few notations used throughout the paper.

Notation $\|x\|_D := \sqrt{x^T D x}$ for any positive-definite D; $\lambda_{\min}(A)$ and $\lambda_{\max}(A)$ denotes the minimum and maximum eigenvalues of a symmetric matrix A, respectively.

2.1. Markov Decision Process (MDP)

A discounted Markov decision process is characterized by the tuple $\mathcal{M} := (\mathcal{S}, \mathcal{A}, P, r, \gamma)$, where $\mathcal{S}$ is a finite state space, $\mathcal{A}$ is a finite action space, $P(s, a, s') := \mathbb{P}[s'|s, a]$ represents the (unknown) state transition probability from state s to s' given action a, $r : \mathcal{S} \times \mathcal{A} \to [0, \sigma]$ is a uniformly bounded stochastic reward, and $\gamma \in (0, 1)$ is the discount factor. If action a is selected with the current state s, then the state transits to s' with probability $P(s, a, s')$ and incurs a random reward $r(s, a) \in [0, \sigma]$ with expectation $R(s, a)$. A stochastic policy is a distribution $\pi \in \Delta_{|\mathcal{S}|\times|\mathcal{A}|}$ representing the probability $\pi(s, a) = \mathbb{P}[a|s]$, P^π denotes the transition matrix whose (s, s') entry is $\mathbb{P}[s'|s] = \sum_{a\in\mathcal{A}} P(s, a, s')\pi(s, a)$, and $d \in \Delta_{|\mathcal{S}|}$ denotes the stationary distribution of the state $s \in \mathcal{S}$ under policy π, i.e., $d = dP^\pi$. The following assumption is standard in the literature.

Assumption 1 *We assume that* $d(s) > 0$ *for all* $s \in \mathcal{S}$.

We also define $r^\pi(s)$ and $R^\pi(s)$ as the stochastic reward and its expectation given the policy π and the current state s, i.e. $R^\pi(s) := \sum_{a\in\mathcal{A}} \pi(s, a)R(s, a)$. The infinite-horizon discounted value function given policy π is

$$J^\pi(s) := \mathbb{E}\left[\sum\nolimits_{k=0}^{\infty} \gamma^k r(s_k, a_k) \middle| s_0 = s\right],$$

where $s \in \mathcal{S}$, $\mathbb{E}$ stands for the expectation taken with respect to the state-action-reward trajectories.

2.2. Linear Function Approximation

Given pre-selected basis (or feature) functions $\phi_1, \ldots, \phi_n : \mathcal{S} \to \mathbb{R}$, $\Phi \in \mathbb{R}^{|\mathcal{S}|\times n}$ is defined as

$$\Phi := \begin{bmatrix} \phi(1)^T \\ \vdots \\ \phi(|\mathcal{S}|)^T \end{bmatrix} \in \mathbb{R}^{|\mathcal{S}|\times n}, \text{ where } \phi(s) := \begin{bmatrix} \phi_1(s) \\ \vdots \\ \phi_n(s) \end{bmatrix} \in \mathbb{R}^n.$$

Here $n \ll |\mathcal{S}|$ is a positive integer and $\phi(s)$ is a feature vector. It is standard to assume that the columns of Φ do not have any redundancy up to linear combinations. We make the following assumption.

Assumption 2 *Φ has full column rank.*

2.3. Reinforcement Learning (RL) Problem

In this paper, the goal of RL with the linear function approximation is to find the weight vector $\theta \in \mathbb{R}^n$ such that $J_\theta := \Phi\theta$ approximates the true value function J^π. This is typically done by minimizing the *mean-square Bellman error* loss function (Sutton et al., 2009a)

$$\min_{\theta\in\mathbb{R}^n} l(\theta) := \frac{1}{2}\mathbb{E}_s[([\mathbb{E}_{s',r}[r(s,a)+\gamma J_\theta(s')] - J_\theta(s)])^2] = \frac{1}{2}\|R^\pi + \gamma P^\pi\Phi\theta - \Phi\theta\|_D^2, \quad (1)$$

where D is defined as a diagonal matrix with diagonal entries equal to a stationary state distribution d under the policy π. Note that due to Assumption 1, $D \succ 0$. In typical RL setting, the model is unknown, while only samples of the state-action-reward are observed. Therefore, the problem can only be solved in stochastic way using the observations. In order to formally analyze the sample complexity, we consider the following assumption on the samples.

Assumption 3 *There exists a Sampling Oracle (SO) that takes input (s,a) and generates a new state s' with probabilities $P(s,a,s')$ and a stochastic reward $r(s,a) \in [0,\sigma]$.*

This oracle model allows us to draw i.i.d. samples (s,a,r,s') from $s \sim d(\cdot), a \sim \pi(s,\cdot), s' \sim P(s,a,\cdot)$. While such an i.i.d. assumption may not necessarily hold in practice, it is commonly adopted for complexity analysis of RL algorithms in the literature (Bhandari et al., 2018; Dalal et al., 2018; Sutton et al., 2009a;b). It's worth mentioning that several recent works also provide complexity analysis when only assuming Markovian noise or exponentially β-mixing properties of the samples (Antos et al., 2008; Bhandari et al., 2018; Dai et al., 2018; Srikant & Ying., 2019). For sake of simplicity, this paper only focuses on the i.i.d. sampling case.

A naive idea for solving 1 is to apply the stochastic gradient descent steps, $\theta_{k+1} = \theta_k - \alpha_k\tilde{\nabla}_\theta l(\theta_k)$, where $\alpha_k > 0$ is a step-size and $\tilde{\nabla}_\theta l(\theta_k)$ is a stochastic estimator of the true gradient of l at $\theta = \theta_k$, $\nabla_\theta l(\theta_k) = \mathbb{E}_{s,a}\big[(\mathbb{E}_{s',r}[r(s,a)+\gamma J_{\theta_k}(s')] - J_{\theta_k}(s))^T(\mathbb{E}_{s'}[\gamma\nabla_\theta J_{\theta_k}(s')] - \nabla_\theta J_{\theta_k}(s))\big]$. This approach is called the residual method (Baird, 1995). Its main drawback is the double sampling issue (Bertsekas & Tsitsiklis, 1996, Lemma 6.10, pp. 364): to obtain an unbiased stochastic estimation of $\nabla_\theta l(\theta_k)$, we need two independent samples given any pair $(s,a) \in \mathcal{S}\times\mathcal{A}$. This is possible under Assumption 3, but hardly implementable in most real applications.

2.4. Standard TD-Learning

In the standard TD-learning (Sutton, 1988), the gradient term $\mathbb{E}_{s'}[\gamma\nabla_\theta J_{\theta_k}(s')]$ in the last line ($\nabla_\theta l(\theta_k)$) is omitted (Bertsekas & Tsitsiklis, 1996, pp. 369). The resulting update rule is $\theta_{k+1} = \theta_k - \alpha_k\eta(\theta_k)$, where $\eta(\theta_k) := -(r(s,a) + \gamma J_{\theta_k}(s') - J_{\theta_k}(s))\nabla_\theta J_{\theta_k}(s)$. While the algorithm avoids the double sampling problem and is simple to implement, a key issue here is that the stochastic gradient $\eta(\theta_k)$ does not correspond to the true gradient of the loss function $l(\theta)$ or any other objective functions, making the theoretical analysis rather subtle. Asymptotic convergence of the TD-learning was given in the original paper (Sutton, 1988) in tabular case and in Tsitsiklis & Van Roy (1997) with linear function approximation. Finite-time convergence analysis was recently established in Bhandari et al. (2018); Dalal et al. (2018); Srikant & Ying. (2019).

Remark. The TD-learning can also be interpreted as minimizing the modified loss function at each iteration

$$l(\theta;\theta') := \frac{1}{2}\mathbb{E}_{s,a}[(\mathbb{E}_{s',r}[r(s,a)+\gamma J_{\theta'}(s')] - J_\theta(s))^2],$$

where θ stands for an online variable and θ' stands for a target variable. At each iteration step k, it sets the target variable to the value of current online variable and performs a stochastic gradient step, $\theta_{k+1} = \theta_k - \alpha_k\,\tilde{\nabla}_\theta l(\theta;\theta_k)\Big|_{\theta=\theta_k}$. A full algorithm is described in Algorithm 1.

Algorithm 1 Standard TD-Learning

1: Initialize θ_0 randomly and set $\theta'_0 = \theta_0$.
2: **for** iteration $k = 0, 1, \ldots$ **do**
3: Sample $s \sim d(\cdot)$ and $a \sim \pi(s,\cdot)$
4: Sample s' and $r(s,a)$ from SO
5: Let $g_k = \phi(s)(r(s,a) + \gamma\phi(s')^T\theta'_k - \phi(s)^T\theta_k)$
6: Update $\theta_{k+1} = \theta_k - \alpha_k g_k$
7: Update $\theta'_{k+1} = \theta_{k+1}$
8: **end for**

Inspired by the the recent target-based deep Q-learning algorithms (Mnih et al., 2015), we consider several alternative updating rules for the target variable that are less aggressive and more general. This then leads to the so-called target-based TD-learning. One of the potential benefits is that by slowing down the update for the target variable, we can reduce the correlation of the target value, or the variance in the gradient estimation, which would then improve the stability of the algorithm. To this end, we introduce three variants of target-based TD: averaging TD, double TD, and periodic TD, each of which corresponds to a different strategy of the

target update. In the following sections, we discuss these algorithms in details and provide their convergence analysis.

3. Averaging TD-Learning (A-TD)

We start by integrating TD-learning with the Polyak averaging strategy for target variable update. This is motivated by the recent deep Q-learning (Mnih et al., 2015) and DDPG (Lillicrap et al., 2015). It's worth pointing out that such a strategy has been commonly used in the deep Q-learning framework, but the convergence analysis remains absent to our best knowledge. Here we first study this for the TD-learning. The basic idea is to minimize the modified loss, $l(\theta;\theta')$, with respect to θ while freezing θ', and then enforce $\theta' \to \theta$ (target tracking). Roughly speaking, the tracking step, $\theta' \to \theta$, is executed with the update

$$\theta_{k+1} = \theta_k - \alpha_k \left.\tilde{\nabla}_\theta l(\theta;\theta'_k)\right|_{\theta=\theta_k},$$
$$\theta'_{k+1} = \theta'_k + \alpha_k \delta(\theta_k - \theta'_k),$$

where $\delta > 0$ is the parameter used to adjust the update speed of the target variable and $\tilde{\nabla}_\theta l(\theta;\theta'_k)$ is a stochastic estimation of $\nabla_\theta l(\theta;\theta'_k)$. A full algorithm is summarized in Algorithm 2, which is called *averaging TD* (A-TD).

Compared to the standard TD-learning in Algorithm 1, the only difference comes from the target variable update in the last line of Algorithm 2. In particular, if we set $\alpha_k = 1/\delta$ and replace θ_k with θ_{k+1} in the second update, then it reduces to the TD-learning.

Algorithm 2 Averaging TD-Learning (A-TD)

1: Initialize θ_0 and θ'_0 randomly.
2: **for** iteration $k = 0, 1, \ldots$ **do**
3: Sample $s \sim d(\cdot)$ and $a \sim \pi(s,\cdot)$
4: Sample s' and $r(s,a)$ from SO
5: Let $g_k = \phi(s)(r(s,a) + \gamma\phi(s')^T\theta'_k - \phi(s)^T\theta_k)$
6: Update $\theta_{k+1} = \theta_k - \alpha_k g_k$
7: Update $\theta'_{k+1} = \theta'_k + \alpha_k\delta(\theta_k - \theta'_k)$
8: **end for**

Next, we prove its convergence under certain assumptions. The convergence proof is based on the ODE (ordinary differential equation) approach (Bhatnagar et al., 2012), which is standard technique used in the RL literature (Sutton et al., 2009b). In the approach, a stochastic recursive algorithm is converted to the corresponding ODE, and the stability of the ODE is used to prove the convergence. The ODE associated with A-TD is $\dot{\theta} = -\Phi^T D\Phi\theta + \gamma\Phi^T DP^\pi\Phi\theta' + \Phi^T DR^\pi$ and $\dot{\theta}' = \delta\theta - \delta\theta'$. We arrive at the following convergence result.

Theorem 1 *Assume that with a fixed policy π, the Markov chain is ergodic and the step-sizes satisfy*

$$\alpha_k > 0, \quad \sum_{k=0}^{\infty}\alpha_k = \infty, \quad \sum_{k=0}^{\infty}\alpha_k^2 < \infty. \tag{2}$$

Then, $\theta'_k \to \theta^*$ and $\theta_k \to \theta^*$ as $k \to \infty$ with probability one, where

$$\theta^* = -(\Phi^T D(\gamma P^\pi - I)\Phi)^{-1}\Phi^T DR^\pi. \tag{3}$$

Remark 1 *Note that θ^* in (3) is not identical to the optimal solution of the original problem in (1). Instead, it is the solution of the projected Bellman equation defined as $\Phi\theta = \mathbf{F}(\Phi\theta)$, where $\mathbf{F}$ is the projected Bellman operator defined by $\mathbf{F}(\Phi\theta) := \Pi(R^\pi + \gamma P^\pi\Phi\theta)$, where Π is the projection onto the range space of Φ, denoted by $R(\Phi)$: $\Pi(x) := \arg\min_{x' \in R(\Phi)} \|x - x'\|_D^2$. The projection can be performed by the matrix multiplication: we write $\Pi(x) := \Pi x$, where $\Pi := \Phi(\Phi^T D\Phi)^{-1}\Phi^T D$.*

Theorem 1 implies that both the target and online variables of the A-TD converge to θ^* which solves the projected Bellman equation. The proof of Theorem 1 is provided in Appendix A of the supplemental material based on the stochastic approximation approach, where we apply the Borkar and Meyn theorem (Bhatnagar et al., 2012, Appendix D). Alternatively, the multi-time scale stochastic approximation (Bhatnagar et al., 2012, pp. 23) can be used with slightly different step-size rules. Due to the introduction of target variable updates, deriving a finite-sample analysis for the modified TD-learning is far from straightforward (Bhandari et al., 2018; Dalal et al., 2018). We will leave this for future investigation.

4. Double TD-Learning (D-TD)

In this section, we introduce a natural extension of the A-TD, which has a more symmetric form. The algorithm mirrors the double Q-learning (Van Hasselt et al., 2016), but with a notable difference. Here, both the online variable and target variable are updated in the same fashion by switching roles. To enforce $\theta' \to \theta$, we also add a correction term $\delta(\theta - \theta')$ to the gradient update. The algorithm is summarized in Algorithm 3, and referred to as the *double TD*-learning (D-TD).

We provide the convergence of the D-TD with linear function approximation below. The proof is similar to the proof of Theorem 1, and is contained in Appendix B of the supplemental material. Noting that asymptotic convergence for double Q-learning has been established in (Hasselt, 2010) for tabular case, but no result is yet known when linear function approximation is used.

Theorem 2 *Assume that with a fixed policy π, the Markov chain is ergodic and the step-sizes satisfy (2). Then, $\theta_k \to \theta^*$ and $\theta'_k \to \theta^*$ as $k \to \infty$ with probability one.*

Algorithm 3 Double TD-Learning (D-TD)

Initialize θ_0 and θ'_0 randomly.
for iteration $k = 0, 1, \dots$ **do**
  Sample $s \sim d(\cdot)$ and $a \sim \pi(s, \cdot)$
  Sample s' and $r(s, a)$ from SO
  Let $g_k = \phi(s)(r(s,a) + \gamma\phi(s')^T\theta'_k - \phi(s)^T\theta_k) + \delta(\theta'_k - \theta_k)$
  Let $g'_k = \phi(s)(r(s,a) + \gamma\phi(s')^T\theta_k - \phi(s)^T\theta'_k) + \delta(\theta_k - \theta'_k)$
  Update $\theta_{k+1} = \theta_k - \alpha_k g_k$
  Update $\theta'_{k+1} = \theta'_k - \alpha_k g'_k$
end for

If D-TD uses identical initial values for the target and online variables, then the two updates remain identical, i.e., $\theta_k = \theta'_k$ for $k \geq 0$. In this case, D-TD is equivalent to the TD-learning with a variant of the step-size rule. In practice, this problem can also be resolved if we use different samples for each update, and the convergence result will still apply to this variation of D-TD.

Compared to the corresponding form of the double Q-learning (Hasselt, 2010), D-TD has two modifications. First, we introduce an additional term, $\delta(\theta'_k - \theta_k)$ or $\delta(\theta_k - \theta'_k)$, linking the target and online parameter to enforce a smooth update of the target parameter. This covers double Q-learning as a special case by setting $\delta = 0$. Moreover, the D-TD updates both target and online parameters in parallel instead of randomly. This approach makes more efficient use of the samples in a slight sacrifice of the computation cost. The convergence of the randomized version is proved with slight modification of the corresponding proof (see Appendix C of the supplemental material for details).

5. Periodic TD-Learning (P-TD)

In this section, we propose another version of the target-based TD-learning algorithm, which more resembles that used in the deep Q-learning (Mnih et al., 2015). It corresponds to the periodic update form of the target variable, which differs from previous sections. Roughly speaking, the target variable is only periodically updated as follows:

$$\theta_{k+1} = \theta_k - \alpha_k \, \tilde{\nabla}_\theta l(\theta; \theta_{k-(k \bmod L)})\Big|_{\theta=\theta_k},$$

where $\tilde{\nabla}_\theta l(\theta; \theta_{k-(k \bmod L)})$ is a stochastic estimator of the gradient $\nabla_\theta l(\theta; \theta_{k-(k \bmod L)})$. The standard TD-learning is recovered by setting $L = 1$.

Alternatively, one can interpret every L iterations of the update as contributing to minimizing the modified loss function

$$\min_\theta l(\theta; \theta') := \frac{1}{2}\mathbb{E}_{s,a}[(\mathbb{E}_{s',r}[r(s,a) + \gamma J_{\theta'}(s')] - J_\theta(s))^2],$$

while freezing the target variable. In other words, the above subproblem is approximately solved at each iteration through L steps of stochastic gradient descent. We formally present the algorithmic idea in a more general way as depicted in Algorithm 4 and call it the *periodic TD* algorithm (P-TD).

Algorithm 4 Periodic TD-Learning (P-TD)

Initialize θ_0 randomly and set $\theta'_0 = \theta_0$.
Set positive integers T and the subroutine iteration steps, L_k, for $k = 0, 1, \dots, T-1$.
Set stepsizes, $\{\beta_t\}_{t=0}^{\infty}$, for the subproblem.
for iteration $k = 0, 1, \dots, T-1$ **do**
  Update $\theta_{k+1} = \texttt{SGD}(\theta_k, \theta'_k, L_k)$ such that

$$\mathbb{E}[\|\theta_{k+1} - \theta^*_{k+1}\|_2^2] \leq \varepsilon_{k+1},$$

  where $\theta^*_{k+1} := \arg\min_{\theta \in \Theta} l(\theta; \theta'_k)$.
  Update $\theta'_{k+1} = \theta_{k+1}$
end for
Return θ_{T+1}

procedure SGD(θ_k,θ'_k,L_k)
  ▷ Subroutine: Stochastic gradient decent steps
  Initialize $\theta_{k,0} = \theta_k$.
  for iteration $t = 0, 1, \dots, L_k - 1$ **do**
    Sample $s \sim d(\cdot)$ and $a \sim \pi(s, \cdot)$
    Sample s' and $r(s, a)$ from SO
    Let $g_t = \phi(s)(r(s,a) + \gamma\phi(s')^T\theta'_k - \phi(s)^T\theta_{k,t})$
    Update $\theta_{k,t+1} = \theta_{k,t} - \beta_t g_t$
  end for
  Return θ_{k,L_k}
end procedure

For the P-TD, given a fixed target variable θ'_k, the subroutine, $\texttt{SGD}(\theta_k, \theta'_k, L_k)$, runs stochastic gradient descent steps L_k times in order to approximately solve the subproblem $\arg\min_{\theta \in \mathbb{R}^n} l(\theta; \theta'_k)$, for which an unbiased stochastic gradient estimator is obtained by using observations. Upon solving the subproblem after L_k steps, the next target variable is replaced with the next online variable. This makes it similar to the original deep Q-learning (Mnih et al., 2015) as it is periodic if L_k is set to a constant. Moreover, P-TD is also closely related to the TD-learning Algorithm 1. In particular, if $L_k = 0$ for all $k = 0, 1, \dots, T-1$, then P-TD corresponds to the standard TD.

Based on the standard results in Bottou et al. (2018, Theorem 4.7), the SGD subroutine converges to the optimal solution, $\theta^*_{k+1} := \arg\min_{\theta \in \mathbb{R}^n} l(\theta; \theta'_k)$. But as we only apply a finite number L_k steps of SGD, the subroutine will return an approximate solution with a certain error bound ε_k in expectation, i.e., $\mathbb{E}[\|\theta_{k+1} - \theta^*_{k+1}\|_2^2 | \theta_k] \leq \varepsilon_{k+1}$.

Below, we establish a finite-time convergence analysis of P-TD. We first characterize the expected error of the solution.

Theorem 3 *Consider Algorithm 4. We have*

$$\mathbb{E}[\|\Phi\theta_T - \Phi\theta^*\|_D]$$
$$\leq \|\Phi\|_D \sqrt{\max_{s\in\mathcal{S}} d(s)} \sum_{k=1}^{T-1} \gamma^{T-k}\sqrt{\varepsilon_k} + \gamma^T \mathbb{E}[\|\Phi\theta_0 - \Phi\theta^*\|_D].$$

Moreover,

$$\mathbb{P}[\|\Phi\theta_T - \Phi\theta^*\|_D \geq \tau] \leq \frac{\gamma^T \mathbb{E}[\|\Phi\theta_0 - \Phi\theta^*\|_D]}{\tau} + \frac{\|\Phi\|_D \sqrt{\max_{s\in\mathcal{S}} d(s)}}{\tau} \sum_{k=1}^{T-1} \gamma^{T-k}\sqrt{\varepsilon_k}.$$

The result implies that P-TD achieves an ϵ-optimal solution with high probability by approaching $T \to \infty$ and controlling the error bounds ε_k. In particular, if $\varepsilon_k = \varepsilon$ for all $k \geq 0$, then $\mathbb{P}[\|\Phi\theta_T - \Phi\theta^*\|_D \geq \tau] \leq \frac{\|\Phi\|_D \sqrt{\max_{s\in\mathcal{S}} d(s)}\sqrt{\varepsilon}}{\tau(1-\gamma)} + \frac{\gamma^T \mathbb{E}[\|\Phi\theta_0 - \Phi\theta^*\|_D]}{\tau}$. One can see that the error is essentially decomposed into two terms, one from the approximation errors induced from SGD procedures and one from the contraction property of solving the subproblems, which can also be viewed as solving the projected Bellman equations. Full details of the proof can be found in Appendix D of the supplemental material.

To analyze the approximation error from the SGD procedure, existing convergence results in Bottou et al. (2018, Theorem 4.7) can be applied with some modifications.

Proposition 1 *Suppose that the SGD method in Algorithm 4 is run with a stepsize sequence such that, for all $t \geq 0$, $\beta_t = \frac{\beta}{\kappa+t+1}$ for some $\beta > 1/\lambda_{\min}(\Phi^T D\Phi)$ and $\kappa > 0$ such that*

$$\beta_0 = \frac{\beta}{\kappa+2} \leq \frac{1}{\sqrt{\lambda_{\max}(\Phi^T D\Phi\Phi^T D\Phi)}(\xi_3+1)},$$

Then, for any $0 \leq t \leq L_k - 1$, we have

$$\mathbb{E}[\|\theta^*_{k+1} - \theta_{k,t}\|_2^2 | \theta_k] \leq \frac{2}{\lambda_{\min}(\Phi^T D\Phi)} \frac{\chi_1 + \chi_2\|\theta_k - \theta^*\|_2^2}{\kappa+t+1},$$

where

$$\chi_1 := (\xi_1 + \xi_2\|\theta^*\|_2^2)\chi_3 + \frac{(\kappa+1)}{2}\|R^\pi + P^\pi\Phi\theta^* - \Phi\theta^*\|_D^2,$$
$$\chi_2 := \frac{\xi_2\chi_3}{2(\beta\lambda_{\min}(\Phi^T D\Phi) - 1)} + \frac{(\kappa+1)}{2}\lambda_{\max}((P^\pi\Phi - \Phi)^T D(P^\pi\Phi - \Phi)),$$

and

$$\chi_3 := \frac{\beta^2\sqrt{\lambda_{\max}(\Phi^T D\Phi\Phi^T D\Phi)}}{2(\beta\lambda_{\min}(\Phi^T D\Phi) - 1)},$$

$$\xi_1 := 3\sigma^2\|\Phi\|_2^2 + 2(1+\xi_3)^2\|\Phi^T D R^\pi\|_2^2,$$
$$\xi_2 := 3\|\Phi\|_2^4 + 2(1+\xi_3)^2\lambda_{\max}(\Phi^T(P^\pi)^T D\Phi\Phi^T D P^\pi\Phi),$$
$$\xi_3 := \frac{3\|\Phi\|_2^4}{\lambda_{\min}(\Phi^T D\Phi\Phi^T D\Phi)}.$$

Proposition 1 ensures that the subroutine iterate, θ_k, converges to the solution of the subproblem at the rate of $\mathcal{O}(1/L_k)$. Combining Proposition 1 with Theorem 3, the overall sample complexity is derived in the following proposition. We defer the proofs to Appendix E and Appendix F of the supplemental material.

Proposition 2 (Sample Complexity) *The ϵ-optimal solution, $\mathbb{E}[\|\theta_T - \theta^*\|_D] \leq \epsilon$, is obtained by Algorithm 4 with at most*

$$\rho_1(\rho_2\epsilon^{-2} + 4\chi_2)\ln(\epsilon^{-1})$$

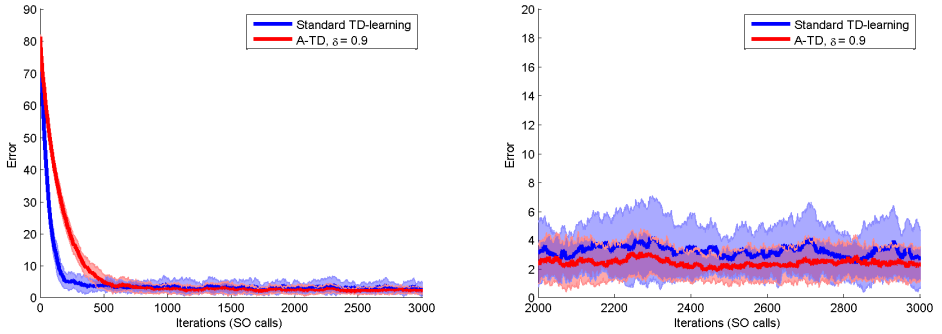
number of SO calls , where

$$\rho_1 := \frac{2\|\Phi\|_D^2}{\lambda_{\min}(\Phi^T D\Phi)^2(1-\gamma)^2\ln\gamma^{-1}},$$
$$\rho_2 := \chi_1\lambda_{\min}(\Phi^T D\Phi) + \chi_2\mathbb{E}[\|\Phi\theta_0 - \Phi\theta^*\|_D^2],$$

and χ_1 and χ_2 are defined in Proposition 1.

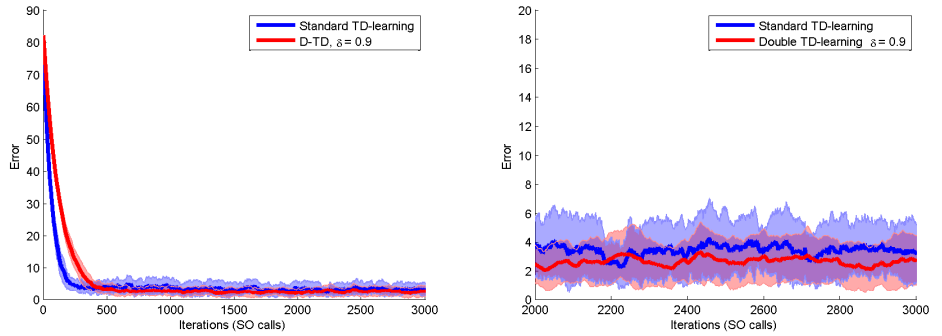
As a result, the overall sample complexity of P-TD is bounded by $\mathcal{O}((1/\epsilon^2)\ln(1/\epsilon))$. As mentioned earlier, non-asymptotic analysis for even the standard TD algorithm is only recently developed in a few work (Bhandari et al., 2018; Dalal et al., 2018; Srikant & Ying., 2019). Our sample complexity result on P-TD, which is a target-based TD algorithm, matches with that developed in Bhandari et al. (2018) with similar decaying step-size sequence, up to a log factor. Yet, our analysis is much simpler and builds directly upon existing results on stochastic gradient descent. Moreover, from the computational perspective, although P-TD runs in two loops, it is as the efficient as standard TD.

P-TD also shares some similarity with the least squares temporal difference (LSTD, Bradtke & Barto (1996)) and its stochastic approximation variant (fLSTD-SA, Prashanth et al. (2014)). LSTD is a batch algorithm that directly estimates the optimal solution as described in (3) through samples, which can also be viewed as exactly computing the solution to a least squares subproblem. fLSTD-SA alleviates the computation burden by applying the stochastic gradient descent (the same as TD update) to solve the subproblems. The key difference between fLSTD-SA and P-TD lies in that the objective for P-TD is adjusted by the target variables across cycles. Lastly, P-TD is also closely related to and can be viewed as a special case of the least-squares fitted Q-iteration (Antos et al., 2008). Both of them solves a similar least squares problems using target values. However, for P-TD, we are able to directly apply the stochastic gradient descent to address the subproblems to near-optimality.



(a) Error evolution over $[0, 3000]$ (b) Error evolution over $[2000, 3000]$

Figure 1: (a) Blue line: error evolution of the standard TD-learning with the step-size $\alpha_k = 1000/(k + 10000)$; Red line: error evolution of A-TD with the step-size $\alpha_k = 1000/(k + 10000)$ and $\delta = 0.9$. The shaded areas depict empirical variances obtained with several realizations. (a) Error over the interval $[0, 3000]$; (b) Error over the interval $[2000, 3000]$.



(a) Error evolution over $[0, 3000]$ (b) Error evolution over $[2000, 3000]$

Figure 2: Blue line: error evolution of the standard TD-learning with the step-size $\alpha_k = 1000/(k + 10000)$; Red line: error evolution of D-TD with the step-size $\alpha_k = 1000/(k + 10000)$ and $\delta = 0.9$. The shaded areas depict empirical variances obtained with several realizations. (a) Error over the interval $[0, 3000]$; (b) Error over the interval $[2000, 3000]$.

6. Simulations

In this section, we provide some preliminary numerical simulation results showing the efficiency of the proposed target-based TD algorithms. We stress that the main goal of this paper is to introduce the family of target-based TD algorithms with linear function approximation and provide theoretical convergence analysis for target TD algorithms, as an intermediate step towards the understanding of target-based Q-learning algorithms. Hence, our numerical experiments simply focus on testing the convergence, sensitivity in terms of the tuning parameters of these target-based algorithms, as well as effects of using target variables as opposed to the standard TD-learning.

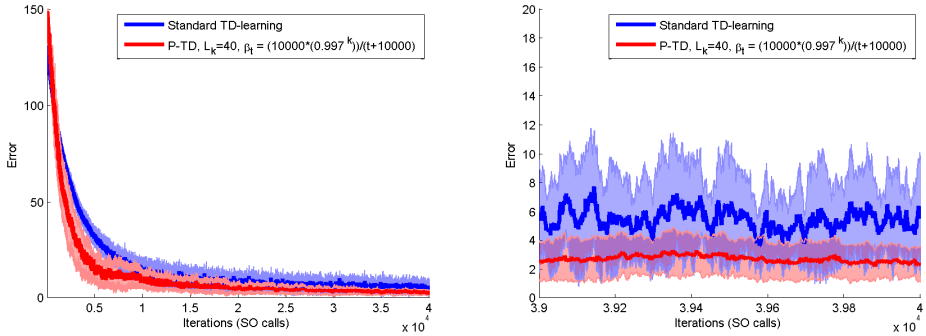
6.1. Convergence of A-TD and D-TD

In this example, we consider an MDP with $\gamma = 0.9$, $|\mathcal{S}| = 10$,

$$P^\pi = \begin{bmatrix} 0.1 & 0.1 & \cdots & 0.1 \\ 0.1 & 0.1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0.1 \\ 0.1 & \cdots & 0.1 & 0.1 \end{bmatrix} \in \mathbb{R}^{10\times 10},$$

and $r^\pi(s) \sim U[0, 20]$, where $U[0, 20]$ denotes the uniform distribution in $[0, 20]$ and $r^\pi(s)$ stands for the reward given policy π and the current state s. The action space and policy are not explicitly defined here. For the linear function approximation, we consider the feature vector with the radial basis function (Geramifard et al., 2013) ($n = 2$), $\phi(s) = \left[\frac{\exp(-(s-0)^2)}{2\times 10^2}, \frac{\exp(-(s-10)^2)}{2\times 10^2}\right] \in \mathbb{R}^2$.

Simulation results are given in Figure 1, which illustrate



(a) Error evolution over $[0, 40000]$ (b) Error evolution over $[39000, 40000]$

Figure 3: Blue line: error evolution of the standard TD-learning with the step-size $\alpha_k = 10000/(k + 10000)$. Red line: error of P-TD with the step-size $\beta_t = (10000 \cdot (0.997)^k)/(10000 + t)$ and $L_k = 40$. The shaded areas depict empirical variances obtained with several realizations. (a) Error over the interval $[0, 30000]$; (b) Error over the interval $[29000, 30000]$.

error evolution of the standard TD-learning (blue line) with the step-size, $\alpha_k = 1000/(k + 10000)$ and the proposed A-TD (red line) with the $\alpha_k = 1000/(k + 10000)$ and $\delta = 0.9$. The design parameters of both approaches are set to demonstrate reasonably the best performance with trial and errors. Additional simulation results in Appendix G of the supplemental material provide comparisons for several different parameters. Figure 1(b) provides the results in the same plot over the interval $[2000, 3000]$. The results suggest that although A-TD with $\delta = 0.9$ initially shows slower convergence, it eventually converges faster than the standard TD with lower variances after certain iterations. With the same setting, comparative results of D-TD are given in Figure 2.

6.2. Convergence of P-TD

In this section, we provide empirical comparative analysis of P-TD and the standard TD-learning. The convergence results of both approaches are quite sensitive to the design parameters to be determined, such as the step-size rules and total number of iterations of the subproblem. We consider the same example as above but with an alternative linear function approximation with the feature vector consisting of the radial basis function, $\phi(s) = \left[\frac{\exp(-(s-0)^2)}{2\times 10^2}, \frac{\exp(-(s-10)^2)}{2\times 10^2}, \frac{\exp(-(s-20)^2)}{2\times 10^2}.\right] \in \mathbb{R}^3$. From our own experiences, applying the same step-size rule, β_t, for every $k \in \{0, 1, \ldots, T-1\}$ yields unstable fluctuations of the error in some cases. For details, the reader is referred to Appendix G of the supplemental material, which provides comparisons with different design parameters. The results motivate us to apply an adaptive step-size rules for the subproblem of P-TD so that smaller and smaller step-sizes are applied as the outer-loop steps increases. In particular, we employ the adaptive step-size rule, $\beta_{k,t} = (10000 \cdot (0.997)^k)/(10000 + t)$ with $L_k = 40$ for P-TD, and the corresponding simulation results are given in Figure 3, where P-TD outperforms the standard TD with the step-size, $\alpha_k = 10000/(k+10000)$, best tuned for comparison. Figure 3(b) provides the results in Figure 3 in the interval $[29000, 30000]$, which clearly demonstrates that the error of P-TD is smaller with lower variances.

7. Conclusion

In this paper, we propose a new family of target-based TD-learning algorithms, including the *averaging TD, double TD*, and *periodic TD*, and provide theoretical analysis on their convergences. The proposed TD algorithms are largely inspired by the recent success of deep Q-learning using target networks and mirror several of the practical strategies used for updating target network in the literature. Simulation results show that integrating target variables into TD-learning can also help stabilize the convergence by reducing variance of and correlations with the target. Our convergence analysis provides some theoretical understanding of target-based TD algorithms. We hope this would also shed some light on the theoretical analysis for target-based Q-learning algorithms and non-linear RL frameworks.

Possible future topics include (1) developing finite-time convergence analysis for A-TD and D-TD; (2) extending the analysis of the target-based TD-learning to the Q-learning case w/o function approximation; and (3) generalizing the target-based framework to other variations of TD-learning and Q-learning algorithms.

Acknowledgment

We thank the anonymous reviewers of ICML 2019 for their insightful comments and acknowledge funding from NSF-CRII-1755829.

References

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

Antsaklis, P. J. and Michel, A. N. A linear systems primer. 2007.

Baird, L. Residual algorithms: reinforcement learning with function approximation. In *Machine Learning Proceedings*, pp. 30–37. 1995.

Bertsekas, D. P. *Dynamic programming and optimal control*. Athena Scientific Belmont, MA, 1995.

Bertsekas, D. P. and Tsitsiklis, J. N. *Neuro-dynamic programming*. Athena Scientific Belmont, MA, 1996.

Bertsekas, D. P. and Yu, H. Projected equation methods for approximate solution of large linear systems. *Journal of Computational and Applied Mathematics*, 227(1):27–50, 2009.

Bhandari, J., Russo, D., and Singal, R. A finite time analysis of temporal difference learning with linear function approximation. *arXiv preprint arXiv:1806.02450*, 2018.

Bhatnagar, S., Prasad, H. L., and Prashanth, L. A. *Stochastic recursive algorithms for optimization: simultaneous perturbation methods*, volume 434. Springer, 2012.

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

Boyd, S. and Vandenberghe, L. *Convex optimization*. Cambridge University Press, 2004.

Bradtke, S. J. and Barto, A. G. Linear least-squares algorithms for temporal difference learning. *Machine Learning*, 22(1):33–57, Mar 1996.

Bubeck, S. et al. Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*, 8(3-4):231–357, 2015.

Chen, C.-T. *Linear System Theory and Design*. Oxford University Press, Inc., 1995.

Dai, B., He, N., Pan, Y., Boots, B., and Song, L. Learning from conditional distributions via dual embeddings. In *Artificial Intelligence and Statistics*, pp. 1458–1467, 2017.

Dai, B., Shaw, A., Li, L., Xiao, L., He, N., Liu, Z., Chen, J., and Song, L. SBEED: Convergent reinforcement learning with nonlinear function approximation. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1125–1134. PMLR, 10–15 Jul 2018.

Dalal, G., Szörényi, B., Thoppe, G., and Mannor, S. Finite sample analyses for TD(0) with function approximation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Dann, C., Neumann, G., and Peters, J. Policy evaluation with temporal differences: A survey and comparison. *Journal of Machine Learning Research*, 15(1):809–883, 2014.

Geramifard, A., Walsh, T. J., Tellex, S., Chowdhary, G., Roy, N., How, J. P., et al. A tutorial on linear function approximators for dynamic programming and reinforcement learning. *Foundations and Trends® in Machine Learning*, 6(4):375–451, 2013.

Gu, S., Lillicrap, T., Sutskever, I., and Levine, S. Continuous deep q-learning with model-based acceleration. In *International Conference on Machine Learning*, pp. 2829–2838, 2016.

Hasselt, H. V. Double Q-learning. In *Advances in Neural Information Processing Systems*, pp. 2613–2621, 2010.

Heess, N., Hunt, J. J., Lillicrap, T. P., and Silver, D. Memory-based control with recurrent neural networks. *arXiv preprint arXiv:1512.04455*, 2015.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Mahadevan, S., Liu, B., Thomas, P., Dabney, W., Giguere, S., Jacek, N., Gemp, I., and Liu, J. Proximal reinforcement learning: A new theory of sequential decision making in primal-dual spaces. *arXiv preprint arXiv:1405.6757*, 2014.

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

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937, 2016.

Prashanth, L. A., Korda, N., and Munos, R. Fast lstd using stochastic approximation: Finite time analysis and

application to traffic control. In Calders, T., Esposito, F., Hüllermeier, E., and Meo, R. (eds.), *Machine Learning and Knowledge Discovery in Databases*, pp. 66–81. Springer Berlin Heidelberg, 2014.

Srikant, R. and Ying., L. Finite-time error bounds for linear stochastic approximation and TD learning. *arXiv preprint arXiv:1902.00923*, 2019.

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

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

Sutton, R. S., Maei, H. R., and Szepesvári, C. A convergent $O(n)$ temporal-difference algorithm for off-policy learning with linear function approximation. In *Advances in neural information processing systems*, pp. 1609–1616, 2009b.

Tsitsiklis, J. N. and Van Roy, B. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42(5):674–690, 1997.

Van Hasselt, H., Guez, A., and Silver, D. Deep reinforcement learning with double Q-learning. In *AAAI*, volume 2, pp. 5. Phoenix, AZ, 2016.

Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., and Freitas, N. Dueling network architectures for deep reinforcement learning. In *International Conference on Machine Learning*, pp. 1995–2003, 2016.

Watkins, C. J. C. H. and Dayan, P. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

Yu, H. and Bertsekas, D. P. Convergence results for some temporal difference methods based on least squares. *IEEE Transactions on Automatic Control*, 54(7):1515–1531, 2009.