BACKSTEPPING TEMPORAL DIFFERENCE LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Off-policy learning ability is an important feature of reinforcement learning (RL) for practical applications. However, even one of the most elementary RL algorithms, temporal-difference (TD) learning, is known to suffer form divergence issue when the off-policy scheme is used together with linear function approximation. To overcome the divergent behavior, several off-policy TD learning algorithms have been developed until now. In this work, we provide a unified view of such algorithms from a purely control-theoretic perspective. Our method relies on the backstepping technique, which is widely used in nonlinear control theory.

1 INTRODUCTION

Since Mnih et al. (2015), which has demonstrated that deep reinforcement learning (RL) outperforms human in several video games (Atari 2600 games), significant advances has been made in RL theory and algorithms. For instance, Van Hasselt et al. (2016); Lan et al. (2020); Chen et al. (2021) proposed some variants of the so-called deep Q-network (Mnih et al., 2015) that achieves higher scores in Atari games than the original deep Q-network. An improved deep RL was developed in Badia et al. (2020) that performs better than average human scores across 57 Atari games. Not only performing well in video games, but Schrittwieser et al. (2020) also have shown that an RL agent can self-learn chess, Go, and Shogi. Furthermore, RL has shown great success in real world applications, e.g., robotics (Kober et al., 2013), healthcare (Gottesman et al., 2019), and recommendation systems (Chen et al., 2019).

Despite the practical success of deep RL, there is still a gap between theory and practice. One of the notorious phenomena is the deadly triad (Sutton & Barto, 2018), the diverging issue of the algorithm when function approximation, off-policy learning, and bootstrapping are used together. One of the most fundamental algorithms, the so-called temporal-difference (TD) learning (Sutton, 1988), is known to diverge under the deadly triad, and several works have tried to fix this issue for decades. In particular, the seminar works Sutton et al. (2008; 2009) introduced the so-called gradient-TD (GTD), gradient-TD2 (GTD2), and TD learning with correction (TDC), which are off-policy, and have been proved to be convergent with linear function approximation. More recently, Ghiassian et al. (2020) suggested regularized version of TDC called TD learning with regularized correction (TDRC), and showed its favorable features under off-policy settings. Moreover, Lee et al. (2021) developed several variants of GTD based on primal dual formulation.

On the other hand, backstepping control (Khalil, 2015) is a popular method in designing stable controllers for nonlinear systems with special structures. The design technique offers a wide range of stable controllers, and is proved to be robust under various settings. It has been used in various fields including quadrotor helicopters (Madani & Benallegue, 2006), mobile robots (Fierro & Lewis, 1997), and ship control (Fossen & Strand, 1999). Using backstepping control technique, in this paper, we develop a new convergent off-policy TD-learning which is a single time-scale algorithm.

In particular, the goal of this paper is to introduce a new unifying framework to design off-policy TD-learning algorithms under linear function approximation. The main contributions are summarized as follows:

- We propose a systemic way to generate off-policy TD-learning algorithms including GTD2 and TDC from control theoretic perspective.
- Using our framework, we derive a new TD-learning algorithm, which we call backstepping TD (BTD).

- We experimentally verify its convergence and performance under various settings including where off-policy TD has known to be unstable.

In particular, most of the previous works on off-policy TD-learning algorithms (e.g., GTD2 and TDC) are derived based on optimization perspectives starting with an objective function. Then, the convergence is proved by proving stability of the corresponding O.D.E. models. In this paper, we follow reversed steps, and reveal that an off-policy TD-learning algorithm (called backstepping TD) can be derived based on control theoretic motivations. In particular, we develop stable O.D.E. models first using the backstepping technique, and then recover back the corresponding off-policy TD-learning algorithms. The new analysis reveals connections between off-policy TD-learning and notions in control theory, and provides additional insights on off-policy TD-learning with simple concepts in control theory. This sound theoretical foundation established in this paper can potentially motivate further analysis and developments of new algorithms.

Finally, we briefly summarize TD learning algorithms that guarantee convergence under linear function approximation. GTD (Sutton et al., 2008), GTD2 and TDC (Sutton et al., 2009) have been introduced to approximate gradient on mean squared projected Belllman error. Furthermore, GTD and GTD2 has been discovered to solve minimax optimization problem (Macua et al., 2014; Liu et al., 2020). Such sadde-point view point of GTD has led to many interesting results including Du et al. (2017); Dai et al. (2018); Lee et al. (2021). TDRC (Ghiassian et al., 2020) adds an additional term similar to regularization term to one-side of parameter update, and tries to balance between the performance of TD and stability of TDC. TDC++ (Ghiassian et al., 2020) also adds an additional regularization term on both sides of the parameter update. Even though TDRC shows good performance, it introduces additional parameter condition to ensure convergence, whereas TDC++ does not.

2 Preliminaries

2.1 Nonlinear system theory

Nonlinear system theory will play an important role throughout this paper. Here, we briefly review basics of nonlinear systems. Let us consider the continuous-time nonlinear system

$$\dot{x}_t = f(x_t, u_t), \quad x_0 \in \mathbb{R}^n, \tag{1}$$

where $x_0 \in \mathbb{R}^n$ is the initial state, $t \in \mathbb{R}, t \geq 0$ is the time, $x_t \in \mathbb{R}^n$ is the state, $u_t \in \mathbb{R}^n$ is the control input, and $f : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ is a nonlinear mapping. An important concept in dealing with nonlinear systems is the equilibrium point. Considering the state-feedback law $u_t = \mu(x_t)$, the system can be written as $\dot{x}_t = f(x_t, u_t) = f(x_t, \mu(x_t)) =: f(x_t)$, and a point $x = x^e$ in the state-space is said to be an equilibrium point of (1) if it has the property that whenever the state of the system starts at x^e, it will remain at x^e (Khalil, 2015). For $\dot{x}_t = f(x_t)$, the equilibrium points are the real roots of the equation $f(x) = 0$. The equilibrium point x^e is said to be globally asymptotically stable if for any initial state $x_0 \in \mathbb{R}^n$, $x_t \to x^e$ as $t \to \infty$.

An important control design problem is to construct a state-feedback law $u_t = \mu(x_t)$ such that the origin becomes the globally asymptotically stable equilibrium point of (1). To design a state-feedback law to meet such a goal, control Lyapunov function plays a central role, which is defined in the following definition.

Definition 2.1 (Control Lyapunov function (Sontag, 2013))**.** *A positive definite function $V : \mathbb{R}^n \to \mathbb{R}$ is called a control Lyapunov function (CLF) if for all $x \neq 0$, there exists a corresponding control input $u \in \mathbb{R}^m$ that satisfies the inequality, $\nabla_x V(x)^\top f(x, u) < 0$ for all $x \neq 0$.*

Once such a CLF is found, then it guarantees that there exists the control law that stabilizes the system. Moreover, the corresponding state-feedback control law can be extracted from the CLF, e.g., $\mu(x) = \arg\min_u \nabla_x V(x)^\top f(x, u)$ provided that the minimum exists and unique. The concept of control Lyapunov function will be used in the derivations of our main results. For the autonomous system, $\dot{x}_t = f(x_t)$, and Lypaunov function $V : \mathbb{R}^n \to \mathbb{R}$, Lie derivative is defined as $\mathcal{L}_f V(x) := \nabla_x V(x)^\top f(x)$ so that $\dot{V}(x_t) = \mathcal{L}_f V(x_t)$ along the solution.

2.2 STOCHASTIC APPROXIMATION AND O.D.E. APPROACH

Including Q-learning (Watkins & Dayan, 1992) and TD-learning (Sutton, 1988), reinforcement learning algorithms can be considered as stochastic approximation (Robbins & Monro, 1951) described by

$$x_{k+1} = x_k + \alpha_k(f(x_k) + \epsilon_k), \tag{2}$$

where $f : \mathbb{R}^n \to \mathbb{R}^n$ is a nonlinear mapping, and ϵ_k is an i.i.d. noise. Borkar and Meyn theorem (Borkar & Meyn, 2000) is a well-known method to bridge the asymptotic convergence of stochastic approximation and the stability of its corresponding O.D.E. model, which can be expressed as

$$\dot{x}_t = f(x_t), \quad x_0 \in \mathbb{R}^n, \tag{3}$$

where $x_0 \in \mathbb{R}^n$ is initial state, and $t \in \mathbb{R}$, $t \geq 0$ is the time.

Borkar and Meyn theorem (Borkar & Meyn, 2000) states that under the conditions in Assumption 6.1 in the Appendix, global asymptotic stability of the O.D.E. (3) leads to asymptotic convergence of the stochastic approximation update (2), which is formally stated in the following lemma.

Lemma 2.1 (Borkar and Meyn theorem (Borkar & Meyn, 2000))**.** *Suppose that Assumption 6.1 in the Appendix holds, and consider the stochastic approximation in (2). Then, for any initial $x_0 \in \mathbb{R}^n$, $\sup_{k\geq 0} ||x_k|| < \infty$ with probability one. In addition , $x_k \to x^e$ as $k \to \infty$ with probability one, where x^e is the unique equilibrium point of the O.D.E. in (3).*

The main idea of Borkar and Meyn theorem is as follows: iterations of a stochastic recursive algorithm follow the solution of its corresponding O.D.E. in the limit when the step-size satisfies the so-called Robbins-Monro condition (Robbins & Monro, 1951) in (33) in the Appendix. Therefore, by proving asymptotic stability of the O.D.E., we can induce convergence of the original algorithm. In this paper, we will use an O.D.E. model of TD-learning, which is expressed as a linear time-invariant system.

2.3 BACKSTEPPING CONTROL

This section provides the concept of the backstepping control (Kokotovic, 1992; Khalil, 2015), which will be the main tool in this paper to derive TD-learning algorithms. The backstepping technique is a popular tool for generating a CLF (control Lyapunov function) for nonlinear systems with specific structures. In particular, let us start with the following general nonlinear system:

$$\begin{aligned} \dot{\lambda}_t &= f(\lambda_t) + g(\lambda_t)x_t \\ \dot{x}_t &= u_t, \end{aligned} \tag{4}$$

where $\lambda_t \in \mathbb{R}^m, x_t \in \mathbb{R}^m$ are the states, $u_t \in \mathbb{R}^m$ is the input, and $f : \mathbb{R}^m \to \mathbb{R}^m$ and $g : \mathbb{R}^m \to \mathbb{R}$ are continuous functions. The first system is a nonlinear system with a particular affine structure, and the second system is simply an integrator. It can be seen as a cascade interconnection of two systems, where the second system's state is injected to the input of the first system. The backstepping control technique gives us a systematic way to generate a CLF for such particular nonlinear systems provided that the first system admits a CLF independently. To this end, we suppose that the first system admits a CLF. Through the backstepping approach, designing a stable control law for the above system can be summarized in the following steps:

Step 1. Consider x_t in (4) as virtual input $x_s(\lambda_t)$ (state-feedback controller), and consider the following system: $\dot{\lambda}_t = f(\lambda_t) + g(\lambda_t)x_s(\lambda_t)$. Design $x_s(\lambda_t)$ such that the above system admits a CLF V, i.e., it admits a positive definite and radially unbounded function V such that its time derivative is negative definite, i.e.,$\dot{V}(\lambda_t) < 0, \forall \lambda_t \neq 0$.

Step 2. Denote the error between the virtual state-feedback controller $x_s(\lambda_t)$ and state variable x_t as $z_t := x_t - x_s(\lambda_t)$. Now, rewrite the original O.D.E. in (4) with the new variable (λ_t, z_t): $\frac{d}{dt}\begin{bmatrix}\lambda_t \\ z_t\end{bmatrix} = \begin{bmatrix} f(\lambda_t) + g(\lambda_t)x_s(\lambda_t) + g(\lambda_t)z_t \\ u_t - \dot{x}_s(\lambda_t)\end{bmatrix}$

Step 3. Design the control input u_t such that the above system is stable. One popular choice is to consider the CLF $V_c(\lambda_t, z_t) := V(\lambda_t) + ||z_t||^2/2$, where $V(\lambda_t)$ is defined in Step 1. Then choose u_t such that the time derivative of $V_c(\lambda_t, z_t)$ to be negative definite.

A simple example of designing a stabilizing control law through the backstepping technique is given in Appendix Section 6.3.

2.4 MARKOV DECISION PROCESS

In this paper, we consider a Markov decision process (MDP) characterized by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma, r)$, where $\mathcal{S} := \{1, 2, \ldots, |\mathcal{S}|\}$ stands for the set of finite state space, $|\mathcal{S}|$ denotes the size of $\mathcal{S}$, $\mathcal{A} := \{1, 2, \ldots, |\mathcal{A}|\}$ denotes the set of finite action space, $|\mathcal{A}|$ is the size of $\mathcal{A}$, $\gamma \in (0, 1)$ is the discount factor, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ denotes the Markov transition kernel, and $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ means the reward function. In particular, if an agent at state $s \in \mathcal{S}$, takes action $a \in \mathcal{A}$, then the current state transits to the next state $s' \in \mathcal{S}$ with probability $\mathcal{P}(s, a, s')$, and the agent receives reward $r(s, a, s')$. Each element of the state to state transition matrix under policy π, denoted by $P^\pi \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}$ is $[P^\pi]_{ij} := \sum_{a\in\mathcal{A}} \pi(a|i)\mathcal{P}(i, a, j), \quad 1 \le i, j \le |\mathcal{S}|$, where $[P^\pi]_{ij}$ corresponds to i-th row and j-th column element of matrix P^π. Moreover, the stationary state distribution induced by policy μ, is denoted as $d^\mu : \mathcal{S} \to [0, 1]$, i.e., ${d^\mu}^\top P^\mu = {d^\mu}^\top$. With the above setup, we define the following matrix notations:

$$D^\mu := \begin{bmatrix} d^\mu(1) & & \\ & \ddots & \\ & & d^\mu(|\mathcal{S}|) \end{bmatrix} \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}, \quad R^\pi = \begin{bmatrix} \mathbb{E}_{a\sim\pi}[r(s, a, s')|s = 1] \\ \mathbb{E}_{a\sim\pi}[r(s, a, s')|s = 2] \\ \vdots \\ \mathbb{E}_{a\sim\pi}[r(s, a, s')|s = |\mathcal{S}|] \end{bmatrix} \in \mathbb{R}^{|\mathcal{S}|},$$

where D^μ is a diagonal matrix of the state distribution induced by behavior policy μ, each element of R^π is the expected reward under policy π at the corresponding state. The policy evaluation problem aims to approximate the value function at state $s \in \mathcal{S}$, $v^\pi(s) := \mathbb{E}\left[\sum_{k=0}^\infty \gamma^k r(S_k, A_k, S_{k+1}) \middle| S_0 = s, \pi\right]$, where the trajectory is generated under policy $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$. In this paper, we consider the linear function approximation to approximate the value function $v^\pi(s)$. In particular, we parameterize the value function $v^\pi(s)$ with $\phi^\top(s)\xi$, where $\phi : \mathcal{S} \to \mathbb{R}^n$ is a pre-selected feature vector with $\phi(s) := [\phi_1(s) \quad \cdots \quad \phi_n(s)], \phi_1, \ldots, \phi_n : \mathcal{S} \to \mathbb{R}$ are feature functions, and $\xi \in \mathbb{R}^n$ is the learning parameter. The goal of the policy evaluation problem is then to approximate the value function $v^\pi(s)$ using this linear parameterization, i.e., $\phi^\top(s)\xi \approx v^\pi(s)$. Moreover, using the matrix notation $\Phi := [\phi(1), \phi(2), \cdots, \phi(|\mathcal{S}|)]^\top \in \mathbb{R}^{|\mathcal{S}|\times n}$, called the feature matrix, the linear parameterization can be written in the vector form $\Phi\xi$. We also assume that Φ is full column rank matrix throughout the paper, which is a standard assumption (Sutton et al., 2008; 2009; Ghiassian et al., 2020; Lee et al., 2021).

2.5 TEMPORAL DIFFERENCE LEARNING

This section provides a brief background on TD-learning (Sutton, 1988). Suppose that we have access to stochastic samples of state s_k from the state stationary distribution induced by the behavior policy μ, i.e., $s_k \sim d^\mu(\cdot)$, and action is chosen under behavior policy μ, i.e., $a_k \sim \mu(\cdot|s_k)$. Then, we observe the next state s'_k following $s'_k \sim \mathcal{P}(\cdot, a_k, s_k)$, and receive the reward $r_k := r(s_k, a_k, s'_k)$. Using the simplified notations for the feature vectors $\phi_k := \phi(s_k), \quad \phi'_k = \phi(s'_k)$. the TD-learning update at time step k with linear function approximation can be expressed as $\xi_{k+1} = \xi_k + \alpha_k\rho_k\delta_k(\xi_k)$, where $\alpha_k > 0$ is the step-size, $\delta_k(\xi_k) := r_k + \gamma\phi'^\top_k\xi_k - \phi^\top_k\xi_k$ is called the temporal difference or temporal difference error (TD-error), and $\rho_k := \rho(s_k, a_k) = \frac{\pi(a_k|s_k)}{\mu(a_k|s_k)}$ is called the importance sampling ratio (Precup et al., 2001). The importance sampling ratio reweights the TD-error to handle the mismatch between the behavior policy μ and target policy π. It is known that TD-learning with linear function approximation and off-policy learning scheme does not guarantee convergence in general. The above stochastic approximation aims to find fixed point of the following projected Bellman equation, which is, after some manipulations, expressed as:

$$\Phi^\top D^\mu \Phi\xi^* - \gamma X^\top D^\mu P^\pi X\xi^* = \Phi^\top D^\mu R^\pi. \tag{5}$$

To simplify the expressions, let use introduce one more piece of notations:

$$A := \mathbb{E}_{s\sim d^\mu(s), s'\sim P^\pi(s'|s)}[\phi(s)(\phi(s) - \gamma\phi(s'))^\top] = \Phi^\top D^\mu \Phi - \gamma D^\mu P^\pi \Phi \in \mathbb{R}^{n\times n},$$
$$b := \mathbb{E}_{s\sim d^\mu(s), a\sim\pi(a|s), s'\sim P(s'|s,a)}[r(s, a, s')\phi(s)] = \Phi^\top D^\mu R^\pi \in \mathbb{R}^{n\times 1}.$$

Even though we can use arbitrary distribution, for simplicity we assume stationary distribution of μ. Now, we can rewrite (5) compactly as

$$A\xi^* = b. \tag{6}$$

The corresponding O.D.E. for TD-learning can be written as $\dot{\xi}_t = A\xi_t - b, \xi_0 \in \mathbb{R}^n$. Using the coordinate transform $x_k := \xi_k - \xi^*$, we get the O.D.E. $\dot{x}_t = Ax_t, x_0 \in \mathbb{R}^n$, whose origin is globally asymptotically stable equilibrium point if $\rho(s,a) = \frac{\pi(a|s)}{\mu(a|s)} = 1$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$. Throughout the paper we will use the vector $x_k := \xi_k - \xi^*$ to represent the coordinate transform of ξ_k to the origin, and will use ξ_t and x_t to denote the corresponding continuous-time counterparts of ξ_k and x_k, respectively.

2.6 Gradient temporal difference learning

To fix the instability issue of off-policy TD-learning under linear function approximation, Sutton et al. (2008) and Sutton et al. (2009) introduced various stable off-policy TD-learning algorithms, called GTD (gradient TD-learning), GTD2, and TDC (temporal difference correction). The idea behind these algorithms is to minimize the mean-square error of projected Bellman equation (MSPBE) $\min_{\xi \in \mathbb{R}^n} \frac{1}{2}||\Phi^\top D^\mu(R^\pi + \gamma P^\pi \Phi\xi - \Phi\xi)||^2_{(\Phi^\top D^\mu \Phi)^{-1}}$, where $||x||_D := \sqrt{x^\top Dx}$. The core idea of the algorithms is to introduce an additional variable $\lambda_k \in \mathbb{R}^n$ to approximate the stochastic gradient descent method for MSPBE as an objective function. In particular, GTD2 update can be written as

$$\lambda_{k+1} = \lambda_k + \alpha_k(-\phi_k^\top \lambda_k + \rho_k \delta_k(\xi_k))\phi_k$$
$$\xi_{k+1} = \xi_k + \alpha_k(\phi_k^\top \lambda_k \phi_k - \rho_k \gamma \phi_k^\top \lambda_k \phi_k').$$

We denote λ_t to denote continuous time part of λ_k. Since the fixed point for λ_k is zero, it doesn't require coordinate transformation. It is a single time-scale algorithm because it uses a single step-size α_k. The corresponding O.D.E. is expressed as $\dot{\lambda}_t = -C\lambda_t - Ax_t, \dot{x}_t = A^\top \lambda_t$, where $C := \mathbb{E}_{s\sim d^\mu(s)}[\phi(s)\phi^\top(s)] = \Phi^\top D^\mu \Phi \in \mathbb{R}^{n\times n}$. Similarly, TDC update can be written as

$$\lambda_{k+1} = \lambda_k + \alpha_k(-\phi_k^\top \lambda_k + \rho_k \delta_k(\xi_k))\phi_k \tag{7}$$
$$\xi_{k+1} = \xi_k + \beta_k(-\rho_k \gamma \phi_k^\top \lambda_k \phi_k' + \rho_k \delta_k(\xi_k)\phi_k), \tag{8}$$

where the step-sizes, α_k and β_k, satisfy $\alpha_k/\beta_k \to 0$ as $k \to \infty$ and the Robbins and Monro step-size condition (Robbins & Monro, 1951) in (33) in Appendix. It is a two time-scale algorithm because it uses two time-steps, α_k and β_k.

3 Designing TD-learning through backstepping

First, we briefly explain the motivation for our algorithmic development. Borkar and Meyn theorem (Borkar & Meyn, 2000) in Lemma 2.1 is a typical tool to prove convergence of Q-learning (Borkar & Meyn, 2000; Lee & He, 2019) and TD-learning (Sutton et al., 2009; Lee et al., 2021). Most of the previous works on off-policy TD-learning algorithms (e.g., GTD2 and TDC) first start with an objective function, and then derive GTD algorithms based on optimization perspectives. Then, the convergence is proved using the corresponding O.D.E. models and stability theory of linear time-invariant systems. A natural question arises is, can we derive off-policy TD-learning algorithms following a reversed step? In other words, can we develop a stable O.D.E. model first using tools in control theory, and then recover back the corresponding off-policy TD-learning algorithms? In this paper, we reveal that a class of off-policy TD-learning algorithms can be derived based on purely control theoretic motivations following such a reversed process. By doing so, this work provides additional insights on off-policy TD-learning algorithms and gives a sound theoretical foundation on off-policy TD-learning algorithms for further analysis and developments of new algorithms.

Designing stabilizing control laws for continuous-time nonlinear system has been successful over the past decades (Khalil, 2015). One such technique, so called backstepping, is a popular controller design method in non-linear control literature (Khalil, 2015). With the help of the backstepping method (Khalil, 2015), we design stabilizing control laws for continuous-time systems, and then the

corresponding off-policy TD-learning algorithms are derived, and are shown to be convergent via Borkar and Meyn theorem (Borkar & Meyn, 2000) in Lemma 2.1. The brief procedure is explained in the following steps:

Step 1) Choose an appropriate continuous-time dynamic model such that (a) we can recover the TD-fixed point ξ^* in (6) via its equilibrium point; (b) the corresponding stochastic approximation algorithm can be implementable only through transitions of MDP and accessible data.

Step 2) Using the backstepping method, design a control input to stabilize the dynamic model chosen in Step 1).

3.1 BACKSTEPPING TD

Now, we introduce a new off-policy TD-learning algorithm, which we call Backstepping TD (BTD). Firstly, we will develop a stabilizing control law for the following the continuous-time system:

$$\dot{\lambda}_t = (-C + \eta A)\lambda_t - Ax_t \tag{9}$$
$$\dot{x}_t = u_t, \tag{10}$$

which well explains our motivation. Considering only in (9), we have an unstable system matrix $-C + \eta A$ and our objective becomes to design stabilizing control law u_t using the backstepping techniques. In particular, our aim is to design an appropriate control input u_t for the above system such that the origin is the unique asymptotically stable equilibrium point, i.e., $(\lambda_t, x_t) \to 0$ as $t \to \infty$ for any $(\lambda_0, x_0) \in \mathbb{R}^n \times \mathbb{R}^n$. The overall procedure is depicted in Figure 1 in the Appendix, and we show how to choose the control input u_t in the following lemma.

Lemma 3.1. *Consider the O.D.E. in (9) and (10). If we choose the control input $u_t := (A^\top + \eta^2 A - \eta C)\lambda_t - \eta A x_t$, then the above O.D.E. has globally asymptotically stable origin, i.e., $(\lambda_t, x_t) \to (0, 0)$ as $t \to \infty$ for any $(\lambda_0, x_0) \in \mathbb{R}^n \times \mathbb{R}^n$.*

Proof. In this proof, we follow the steps given in the backstepping scheme in Section 3. First, substituting x_t in (9) with a virtual controller $x_s(\lambda_t)$, we will design a control law $x_s(\lambda_t)$ that stabilizes the following new virtual system:

$$\dot{\lambda}_t = (-C + \eta A)\lambda_t - Ax_s(\lambda_t). \tag{11}$$

Even though matrix C is positive definite, due to matrix A, the system may be unstable. One natural choice of the virtual controller is $x_s(\lambda_t) = \eta\lambda_t$. Plugging into (11) leads to $\dot{\lambda}_t = -C\lambda_t$. The system now has a globally asymptotically stable origin due to the positive definiteness of matrix C. It is straightforward to verify the global asymptotic stability of the above system with the following Lyapunov function:

$$V(\lambda_t) := \frac{||\lambda_t||_2^2}{2}. \tag{12}$$

With this result in mind, we now consider the original O.D.E. in (9) and (10). Applying simple algebraic manipulations yield $\dot{\lambda}_t = -C\lambda_t - A(x_t - \eta\lambda_t), \quad \dot{x}_t = u_t$. The error between x_t and the virtual controller $x_s(\lambda_t)$ can be expressed as new variable z_t, which is $z_t := x_t - x_s(\lambda_t) = x_t - \eta\lambda_t$. Rewriting the O.D.E. in (9) and (10) with (λ_t, z_t) coordinates, we have

$$\begin{aligned} \dot{\lambda}_t &= -C\lambda_t - Az_t \\ \dot{z}_t &= u_t + \eta C\lambda_t + \eta A z_t. \end{aligned} \tag{13}$$

To prove the global asymptotic stability of the above system, consider the function $V_c(\lambda_t, z_t) := V(\lambda_t) + \frac{1}{2}||z_t||_2^2$ where $V(\lambda_t)$ is defined in (12). The time derivative of the Lyapunov function along the system's solution becomes

$$\begin{aligned} \dot{V}_c(\lambda_t, z_t) &= \lambda_t^\top(-C\lambda_t - Az_t) + z_t^\top(u_t + \eta C\lambda_t + \eta A z_t) \\ &= -||\lambda_t||_C^2 + z_t^\top(-A^\top\lambda_t + u_t + \eta C\lambda_t + \eta A z_t). \end{aligned}$$

By taking u_t as $u_t = A^\top \lambda_t - \eta C \lambda_t - \eta A z_t$, the corresponding closed-loop system is $\frac{d}{dt}\begin{bmatrix} \lambda_t \\ z_t \end{bmatrix} = \begin{bmatrix} -C & -A \\ A^\top & 0 \end{bmatrix}\begin{bmatrix} \lambda_t \\ z_t \end{bmatrix} =: f(\lambda_t, z_t)$, and we have $\dot{V}_c = -||\lambda_t||_C^2 \leq 0$ for all $(\lambda_t, z_t) \neq (0,0)$. Since the inequality is not strict, Lyapunov theory cannot be directly applied. Therefore, we will use LaSall'es invariance principle in Lemma 6.1 in Appendix. Define the Lie derivative $\mathcal{L}_f V(\lambda, z) := -\|\lambda\|_C^2$ so that $\dot{V}_c(\lambda_t, z_t) = \mathcal{L}_f V(\lambda_t, z_t)$ along the solution. Consider a solution $(\lambda_t, z_t), t \geq 0$ and the set $S := \{(\lambda, z) \mid \mathcal{L}_f V(\lambda, z) = 0\} = \{(\lambda, z) \mid \lambda = 0\}$. Suppose that the solution, $(\lambda_t, z_t), t \geq 0$, is inside S, i.e., $(\lambda_t, z_t) \in S, t \geq 0$. Then, we should have $\lambda \equiv 0$, which implies from (13) that $z \equiv 0$. Therefore, S can only contain the trivial solution $(\lambda, z) \equiv (0,0)$. Therefore, from LaSall'es invariance principle in Lemma 6.1 and noting that V_c is radially unbounded, the closed-loop system admits the origin as a globally asymptotically stable equilibrium point. Using the relation $z_t := x_t - \eta \lambda_t$, we can also easily conclude that the closed-loop system in the original coordinate (λ_t, x_t) admits the origin as a globally asymptotically stable equilibrium point. □

Using the relation $z_t := x_t - \eta \lambda_t$, the control input in the original coordinate (λ_t, x_t) can be written as $u_t := A^\top \lambda_t - \eta C \lambda_t - \eta A z_t = (A^\top + \eta^2 A - \eta C)\lambda_t - \eta A x_t$. Plugging this input into the original open-loop system in (9) and (10), the closed-loop system in the original coordinate (λ_t, x_t) can written as

$$\dot{\lambda}_t = (-C + \eta A)\lambda_t - A x_t \tag{14}$$

$$\dot{x}_t = (A^\top + \eta^2 A - \eta C)\lambda_t - \eta A x_t, \tag{15}$$

whose origin is also globally asymptotically stable according to Lemma 3.1. Recovering back from x_t to ξ_t, we have $\frac{d}{dt}\begin{bmatrix} \lambda_t \\ \xi_t \end{bmatrix} = \begin{bmatrix} -C + \eta A & -A \\ A^\top + \eta^2 A - \eta C & -\eta A \end{bmatrix}\begin{bmatrix} \lambda_t \\ \xi_t \end{bmatrix} + \begin{bmatrix} b \\ \eta b \end{bmatrix}$. The corresponding stochastic approximation of the O.D.E. in Theorem 3.1 becomes

$$\lambda_{k+1} = \lambda_k + \alpha_k(((-1+\eta)\phi_k^\top - \eta\rho_k\gamma\phi_k'^\top)\lambda_k + \rho_k\delta_k(\xi_k))\phi_k \tag{16}$$

$$\xi_{k+1} = \xi_k + \alpha_k(((-\eta+\eta^2)\phi_k^\top - \eta^2\rho_k\gamma\phi_k'^\top)\lambda_k\phi_k + \eta\rho_k\delta_k(\xi_k)\phi_k + (\phi_k^\top\lambda_k\phi_k - \rho_k\gamma\phi_k^\top\lambda_k\phi_k')). \tag{17}$$

The equilibrium point of the above O.D.E. is $(0, \xi^*)$. Hence, we only need to transform the coordinate of ξ_t to $x_t = \xi_t - \xi^*$, which results to the O.D.E. in (14) and (15). With the above result, we are now ready to prove convergence of Algorithm 1. The proof simply follows from Borkar and Meyn theorem in Lemma 2.1, of which the details can be found in Sutton et al. (2009).

Theorem 3.1. *Under the step size condition (33), with Algorithm 1 in Appendix, $\xi_k \to \xi^*$ as $k \to \infty$ with probability one, where ξ^* is the fixed point of (6).*

Proof. The proof is done by checking Assumption 6.1 in Appendix. □

Remark 3.1. *Theorem 3.1 doesn't require any condition on η. Therefore, we can set $\eta = 0$, which results to GTD2 developed in Sutton et al. (2009).*

3.2 Recovering single time-scale TDC

In this section, we derive a single-time scale version of TDC (Sutton et al., 2009) through the back-stepping design in the previous section. TDC (Sutton et al., 2009) was originally developed as a two-time scale algorithm in Sutton et al. (2009). Even though the two time-scale method provides theoretical guarantee for a larger class of algorithms, the single time-scale scheme provides more simplicity in practice, and shows faster convergence empirically. Subsequently, Maei (2011) provided a single-time scale version of TDC by multiplying a large enough constant $\eta > 0$ to the faster time scale part (7), which leads to

$$\lambda_{k+1} = \lambda_k + \beta_k\eta(-\phi_k^\top\lambda_k + \rho_k\delta_k(\xi_k))\phi_k \tag{18}$$

$$\xi_{k+1} = \xi_k + \beta_k(-\rho_k\gamma\phi_k^\top\lambda_k\phi_k' + \rho_k\delta_k(\xi_k)\phi_k), \tag{19}$$

where

$$\eta > \max\left\{0, -\lambda_{\min}\left(C^{-1}\frac{A+A^\top}{2}\right)\right\}. \tag{20}$$

Here, we derive another version of single-time TDC by multiplying a constant to the slower time-scale part in (8), which results in

$$\lambda_{k+1} = \lambda_k + \alpha_k(-\phi_k^\top\lambda_k + \rho_k\delta_k(\xi_k))\phi_k \tag{21}$$

$$\xi_{k+1} = \xi_k + \alpha_k\beta(\phi_k^\top\lambda_k\phi_k - \rho_k\gamma\phi_k^\top\lambda_k\phi_k' - +\rho_k\delta_k(\xi_k)\phi_k), \tag{22}$$

where β satisfies

$$\begin{cases} 0 < \beta < -\frac{\lambda_{\min}(C)}{\lambda_{\min}(A)} & \text{for} \quad \lambda_{\min}(A) < 0 \\ 0 < \beta & \text{for} \quad \lambda_{\min}(A) > 0 \end{cases}. \tag{23}$$

We can derive the above algorithm following similar steps as in Section 3. Let us first consider the following dynamic model:

$$\dot{\lambda}_t = -C\lambda_t - Ax_t \tag{24}$$

$$\dot{x}_t = u_t \tag{25}$$

Using the backstepping technique, we can prove that the above system admits the origin as a global asymptotically stable equilibrium point with the control input $u_t := \beta\left((A^\top - C)\lambda_t - A\xi_t\right)$, which is shown in the following lemma:

Lemma 3.2. *Consider the O.D.E. in (24) and (25). Suppose that we choose the control input $u_t := \beta\left((A^\top - C)\lambda_t - A\xi_t\right)$, and β satisfies condition (23). Then, the above O.D.E. has globally asymptotically stable origin, i.e., $(\lambda_t, x_t) \to (0,0)$ as $t \to \infty$.*

The proof of Lemma 3.2 is given in Appendix Section 6.4.1. By Borkar and Meyn theorem in Lemma 2.1, we can readily prove the convergence of Algorithm 2 in Appendix, which uses stochastic recursive update (21) and (22).

Theorem 3.2. *Consider Algorithm 2 in Appendix. Under the step size condition (33), and if β satisfies (23), $\xi_k \to \xi^*$ as $k \to \infty$ with probability one, where ξ^* is the fixed point of (6).*

We will call the Algorithm 4 as TDC-slow, and single-time version of TDC suggested by Maei (2011) as TDC-fast. Other than the multiplication of a constant reflecting two-time scale property, we can make TDC into a single-time algorithm, which we call a single time-scale TDC2, while the original version in Maei (2011) will be called the single time-scale TDC. The derivation is given in Appendix Section 6.5. The performance of such versions of TDC are evaluated in Appendix Section 6.9.1. Even though not one of the algorithms outperforms each other, TDC-slow and TDC2 shows better performance in general.

3.3 Generalizing TDC++

This section provides versions of TDC++ (Ghiassian et al., 2020). Ghiassian et al. (2020) suggested another version of TDC so called TDC++. With an additional regularization term ξ_k on both updates of TDC in (7) and (8), the update is written as follows:

$$\lambda_{k+1} = \lambda_k + \alpha_k\eta(-\phi_k^\top\lambda_k + \rho_k\delta_k(\xi_k))\phi_k - \beta\lambda_k) \tag{26}$$

$$\xi_{k+1} = \xi_k + \alpha_k(-\rho_k\gamma\phi_k^\top\lambda_k\phi_k' - \beta\lambda_k + \rho_k\delta_k(\xi_k)\phi_k), \tag{27}$$

where $\eta > 0$ satisfies (20) and $\beta > 0$ is a new parameter. Note that TDC++ can be simply viewed as variant of TDC by adding the term $\beta\lambda_k$ in the update, which can be seen as a regularization term. Therefore, letting $\beta = 0$ yields the original TDC. In this paper, we prove that our controller design leads to the following update:

$$\lambda_{k+1} = \lambda_k + \alpha_k\eta(-\phi_k^\top\lambda_k + \rho_k\delta_k(\xi_k))\phi_k - \beta\lambda_k) \tag{28}$$

$$\xi_{k+1} = \xi_k + \alpha_k(-\rho_k\gamma\phi_k^\top\lambda_k\phi_k' + (1-\kappa\eta)\phi_k^\top\lambda_k\phi_k - \kappa\beta\eta\lambda_k + \rho_k\kappa\eta\delta_k(\xi_k)\phi_k), \tag{29}$$

where κ and β are new parameters and when $\kappa = 1/\eta$ it becomes TDC++. The difference with the original TDC++ can be seen in their corresponding O.D.E. forms. The corresponding O.D.E. for (26) and (27) (original TDC++) can be expressed as: $\frac{d}{dt}\begin{bmatrix}\lambda_t \\ x_t\end{bmatrix} = \begin{bmatrix}-\eta(C+\beta I) & -\eta A \\ A^\top - C - \beta I & -A\end{bmatrix}\begin{bmatrix}\lambda_t \\ x_t\end{bmatrix}$. Meanwhile, the O.D.E. corresponding to (28) and (29) (new TDC++) becomes $\frac{d}{dt}\begin{bmatrix}\lambda_t \\ x_t\end{bmatrix} = \begin{bmatrix}-\eta(C+\beta I) & -\eta A \\ A^\top - \kappa\eta(C+\beta I) & -\kappa\eta A\end{bmatrix}\begin{bmatrix}\lambda_t \\ x_t\end{bmatrix}$. We experiment under different of κ and η to examine the behavior of new TDC++. The result shows that in general, smaller κ leads to better performance. The results are given in Appendix Section 6.9.

Lemma 3.3. *Consider the following O.D.E.:*

$$\dot{\lambda}_t = -\eta(C+\beta I)\lambda_t - \eta A x_t \tag{30}$$

$$\dot{x}_t = u_t. \tag{31}$$

Suppose that we choose the control input $u_t := (A^\top - \kappa\eta(C+\beta I))\lambda_t - \kappa\eta A x_t$. Assume $\eta > 0$ and β and κ satisfies the following condition: $\beta + \kappa\lambda_{\min}(A) > \lambda_{\min}(C)$. Then, the above O.D.E. has globally asymptotically stable origin, i.e., $(\lambda_t, x_t) \to (0,0)$ as $t \to \infty$.

The proof is given in Appendix Section 6.4.2. By Borkar and Meyn theorem in Lemma 2.1, we can readily prove the convergence of stochastic recursive update with (28) and (29) whose pseudo code is given in Algorithm 5 in Appendix.

Theorem 3.3. *Consider Algorithm 5 in Appendix. Under the step-size condition (33) and if η satisfies (20), then $\xi_k \to \xi^*$ as $k \to \infty$ with probability one, where ξ^* is the TD fixed point in (6).*

Remark 3.2. *We can replace the regularization term with nonlinear terms satisfying certain conditions. The details are given in Appendix Section 6.6.*

4 Experiments

In this section, we verify the performance and convergence of the proposed BTD under standard benchmarks to evaluate performance of off-policy TD-learning algorithms, including Baird environment (Baird, 1995), RandomWalk (Sutton et al., 2009) with different features, and Boyan chain (Boyan, 2002). The details about the environments are given in Appendix Section 6.7. From the experiments, we see how BTD behaves under different coefficients $\eta \in \{-1, -0.5, 0, 0.5, 1\}$. We measure the Root Mean-Squared Projected Bellman Error (RMSPBE) as the performance metric, and every results are averaged over 100 runs. As can be seen in Table 1, the result with $\eta = 0.5$ shows the best performance except at Baird, where $\eta = 0$, corresponding to GTD2 performs best. Overall results are given in Appendix Section 6.8.

Table 1: Backstepping TD, step-size = 0.01

Env \ η	-0.5	-0.25	0	0.25	0.5
Boyan	1.513 ± 0.666	1.481 ± 0.656	1.452 ± 0.647	1.428 ± 0.64	$\mathbf{1.408 \pm 0.635}$
Dependent	0.112 ± 0.192	0.097 ± 0.163	0.086 ± 0.142	0.079 ± 0.128	$\mathbf{0.076 \pm 0.122}$
Inverted	0.205 ± 0.254	0.173 ± 0.218	0.151 ± 0.193	0.139 ± 0.177	$\mathbf{0.136 \pm 0.172}$
Tabular	0.168 ± 0.28	0.147 ± 0.238	0.133 ± 0.208	0.124 ± 0.191	$\mathbf{0.122 \pm 0.188}$
Baird	0.1 ± 0.641	0.09 ± 0.629	$\mathbf{0.085 \pm 0.625}$	0.087 ± 0.628	0.092 ± 0.637

5 Conclusion

In this paper we have proposed a new framework to design off-policy TD-learning algorithms under linear function approximation from control-theoretic view. Our framework discovers new type of off-policy TD learning algorithms which includes GTD2, and with appropriate parameter selection, it performs better than GTD2. Furthermore, we reformulated TDC into single time-scale algorithm and generalized TDC++. Future research directions would be extending this framework to non-linear function approximation setting.

REFERENCES

Adrià Puigdomènech Badia, Bilal Piot, Steven Kapturowski, Pablo Sprechmann, Alex Vitvitskyi, Zhaohan Daniel Guo, and Charles Blundell. Agent57: Outperforming the atari human benchmark. In *International Conference on Machine Learning*, pp. 507–517. PMLR, 2020.

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

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

Justin A Boyan. Technical update: Least-squares temporal difference learning, 2002.

Xinshi Chen, Shuang Li, Hui Li, Shaohua Jiang, Yuan Qi, and Le Song. Generative adversarial user model for reinforcement learning based recommendation system. In *International Conference on Machine Learning*, pp. 1052–1061. PMLR, 2019.

Xinyue Chen, Che Wang, Zijian Zhou, and Keith Ross. Randomized ensembled double q-learning: Learning fast without a model. *arXiv preprint arXiv:2101.05982*, 2021.

Bo Dai, Albert Shaw, Lihong Li, Lin Xiao, Niao He, Zhen Liu, Jianshu Chen, and Le Song. Sbeed: Convergent reinforcement learning with nonlinear function approximation. In *International Conference on Machine Learning*, pp. 1125–1134. PMLR, 2018.

Simon S Du, Jianshu Chen, Lihong Li, Lin Xiao, and Dengyong Zhou. Stochastic variance reduction methods for policy evaluation. In *International Conference on Machine Learning*, pp. 1049–1058. PMLR, 2017.

Rafael Fierro and Frank L Lewis. Control of a nonholomic mobile robot: Backstepping kinematics into dynamics. *Journal of robotic systems*, 14(3):149–163, 1997.

Thor I Fossen and Jan P Strand. Tutorial on nonlinear backstepping: Applications to ship control. 1999.

Sina Ghiassian, Andrew Patterson, Shivam Garg, Dhawal Gupta, Adam White, and Martha White. Gradient temporal-difference learning with regularized corrections. In *International Conference on Machine Learning*, pp. 3524–3534. PMLR, 2020.

Omer Gottesman, Fredrik Johansson, Matthieu Komorowski, Aldo Faisal, David Sontag, Finale Doshi-Velez, and Leo Anthony Celi. Guidelines for reinforcement learning in healthcare. *Nature medicine*, 25(1):16–18, 2019.

Hassan K Khalil. *Nonlinear control*, volume 406. Pearson New York, 2015.

Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

Petar V Kokotovic. The joy of feedback: nonlinear and adaptive. *IEEE Control Systems Magazine*, 12(3):7–17, 1992.

Qingfeng Lan, Yangchen Pan, Alona Fyshe, and Martha White. Maxmin q-learning: Controlling the estimation bias of q-learning. *arXiv preprint arXiv:2002.06487*, 2020.

Donghwan Lee and Niao He. A unified switching system perspective and ode analysis of q-learning algorithms. *arXiv preprint arXiv:1912.02270*, 2019.

Donghwan Lee, Han-Dong Lim, Jihoon Park, and Okyong Choi. Versions of gradient temporal difference learning. *arXiv preprint arXiv:2109.04033*, 2021.

Bo Liu, Ji Liu, Mohammad Ghavamzadeh, Sridhar Mahadevan, and Marek Petrik. Finite-sample analysis of proximal gradient td algorithms. *arXiv preprint arXiv:2006.14364*, 2020.

Sergio Valcarcel Macua, Jianshu Chen, Santiago Zazo, and Ali H Sayed. Distributed policy evaluation under multiple behavior strategies. *IEEE Transactions on Automatic Control*, 60(5):1260–1274, 2014.

Tarek Madani and Abdelaziz Benallegue. Backstepping control for a quadrotor helicopter. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3255–3260. IEEE, 2006.

Hamid Reza Maei. Gradient temporal-difference learning algorithms. 2011.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Doina Precup, Richard S Sutton, and Sanjoy Dasgupta. Off-policy temporal-difference learning with function approximation. In *ICML*, pp. 417–424, 2001.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pp. 400–407, 1951.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

Eduardo D Sontag. *Mathematical control theory: deterministic finite dimensional systems*, volume 6. Springer Science & Business Media, 2013.

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

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Richard S Sutton, Csaba Szepesvári, and Hamid Reza Maei. A convergent o (n) algorithm for off-policy temporal-difference learning with linear function approximation. *Advances in neural information processing systems*, 21(21):1609–1616, 2008.

Richard S Sutton, Hamid Reza Maei, Doina Precup, Shalabh Bhatnagar, David Silver, Csaba Szepesvári, and Eric Wiewiora. 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, 2009.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3):279–292, 1992.