University of Groningen

On recursive temporal difference and eligibility traces

Baldi, Simone; Liu, Di; Zhang, Zichen

Published in:
IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society

DOI:
[10.1109/IECON43393.2020.9254411](https://doi.org/10.1109/IECON43393.2020.9254411)

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version
Publisher's PDF, also known as Version of record

Publication date:
2020

[Link to publication in University of Groningen/UMCG research database](#)

Citation for published version (APA):

Baldi, S., Liu, D., & Zhang, Z. (2020). On recursive temporal difference and eligibility traces. In *IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society* (pp. 501-506). IEEE.
<https://doi.org/10.1109/IECON43393.2020.9254411>

Copyright

Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license. More information can be found on the University of Groningen website: <https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment>.

Take-down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): <http://www.rug.nl/research/portal>. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.

On recursive temporal difference and eligibility traces

Simone Baldi, Di Liu, Zichen Zhang

Abstract—This work studies a new reinforcement learning method in the framework of Recursive Least-Squares Temporal Difference (RLS-TD). Differently from the standard mechanism of eligibility traces, leading to RLS-TD(λ), in this work we show that the forgetting factor commonly used in gradient-based estimation has a similar role to the mechanism of eligibility traces. We adopt an instrumental variable perspective to illustrate this point and we propose a new algorithm, namely — RLS-TD with forgetting factor (RLS-TD-f). We test the proposed algorithm in a Policy Iteration setting, i.e. when the performance of an initially stabilizing controller must be improved. We take the cart-pole benchmark as experimental platform: extensive experiments show that the proposed RLS-TD algorithm exhibits larger performance improvements in the largest portion of the state space.

Index Terms—Reinforcement learning; temporal difference; least-squares; instrumental variable method; eligibility traces.

I. INTRODUCTION

In recent years, reinforcement learning has been an active research area not only in machine learning [1], but also in control engineering [2], [3], and complex optimization and decision-making problems [4], [5], [6], [7]. Among the several reinforcement learning methods, temporal difference (TD) learning is one of the most popular one. The method dates back to 1988, when Sutton presented a method to estimate the value function of a Markov Decision Process (MDP) [8]: the method was named TD(λ) algorithm, where the parameter $0 < \lambda < 1$ refers to the mechanism of *eligibility traces*. The TD(λ) was originally proposed for problems with finite state and action spaces. Since many real-world applications have large or infinite state space, value function approximation methods should be preferred. When combined with nonlinear value function approximators, TD(λ) can not guarantee convergence in general [9]. However, for TD(λ) with linear function approximators, also called linear TD(λ), Dayan showed convergence with $0 < \lambda < 1$ [10].

Based on the theory of linear least-squares estimation, Bradtke and Barto proposed two temporal difference algorithms, called Least-Squares TD(0) (LS-TD(0)) and Recursive Least-Squares TD(0) (RLS-TD(0)), respectively [11]. The main difference between the least-squares and recursive

least-squares version is whether the estimate of the value function is computed from a batch of data, or updated every time a new data arrives [12]. Later, Boyan proposed a class of linear temporal difference learning algorithms called LS-TD(λ) [13], where the mechanism of eligibility traces was embedded in the least-squares estimation: when RLS-TD(0) employs the mechanism of eligibility traces, the result is RLS-TD(λ), which was proposed and analyzed formally in [14].

The TD algorithm is the basic block of many reinforcement learning methods. For example, in [15], the author investigated the application of natural gradient descent to Bellman error based reinforcement learning algorithms. An algorithm known as Complementary Temporal Difference Learning (CTDL) was proposed in [16], which combines a Deep Neural Networks (DNN) with a Self-Organizing Map (SOM) updated by the TD error. See also [17], [18], [19] and references therein. Because TD is the basic block of many reinforcement learning methods, it is very important to study possible extensions and insights to the method. An open problem in the TD method regards the mechanism of eligibility traces. In fact, it is not clear how to select the parameter of the eligibility trace, and it is still an open question to give a clear interpretation to such mechanism [1]. In this work, we show that the mechanism of eligibility traces is analogous to the use of forgetting factor in parameter estimation methods.

With this observation, and considering the gradient structure of TD algorithm, we propose a new TD algorithm namely, RLS-TD with forgetting factor (RLS-TD-f). The effectiveness of the algorithm is tested in the cart-pole benchmark: it is shown that the proposed method overcomes RLS-TD(λ) over large portions of the state space.

The rest of the paper is organized as follows. In Sect. II, we introduce various elements and important concepts of reinforcement learning. Sect. III recalls TD(λ), LS-TD(λ) and RLS-TD(λ). In Sect. IV, we introduce the instrumental variable method to provide a new optimization-based perspective of RLS-TD. The new algorithm — RLS-TD with forgetting factor is proposed. In Sect. V, we test the new method on the cart-pole benchmark and compare it with RLS-TD(λ). Finally, Sect. VI contains concluding remarks.

II. BACKGROUND ON REINFORCEMENT LEARNING

Reinforcement learning problems involve learning what to do — how to map situations to actions — so as to maximize a numerical reward signal. Essentially, a reinforcement learning problem describes a closed-loop problem because

This work was supported by the Fundamental Research Funds for the Central Universities grant 3207012004A2, the special guiding fund for double first-class grant 3307012001A, and the China Scholarship Council grant 201906090146. (Corresponding author: Di Liu)

S. Baldi is with School of Cyber Science and Engineering, Southeast University, and with School of Mathematics Southeast University, Nanjing 210096, China (e-mail: s.baldi@tudelft.nl).

D. Liu is with School of Cyber Science and Engineering, Southeast University, Nanjing 210096, China (e-mail: liud923@126.com).

Z. Zhang is with School of Mathematics Southeast University, Nanjing 210096, China (e-mail: zzcll.lucky@163.com).

the learning system's actions influence its later inputs (cf. Fig. 1).

Fig. 1: The agent–environment interaction.

The *value function* is a crucial function in reinforcement learning since it specifies what is good in the long run.

$$V_\pi(s) = E_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | S_t = s \right] \quad (1)$$

where $r_{t+1}, r_{t+2}, \dots$ is the sequence of rewards received after time step t , γ is a parameter, $0 < \gamma \leq 1$, called the discount rate, $E_\pi[\cdot]$ denotes the expected value of a random variable given that the agent follows policy π . The value function is defined with respect to a particular policy, because the rewards the agent can expect to receive depend on what actions it will take. We call the function V_π the state-value function for policy π .

Optimal value functions satisfy the Bellman equation

$$\begin{aligned} V_*(s) &= \max_a E[r_{t+1} + \gamma V_*(S_{t+1}) | S_t = s, A_t = a] \\ &= \max_{a \in \mathcal{A}(s)} \sum_{s', r} p(s', r | s, a) [r + \gamma V_*(s')]. \end{aligned} \quad (2)$$

The reason why researchers are interested in optimal value functions rather than optimal policies is that, once one has V_* , it is relatively easy to determine an optimal policy via (2).

III. BACKGROUND ON TEMPORAL DIFFERENCE

The temporal difference (TD) method was initially proposed in [8] as a way to estimate the value function. The idea of TD is to calculate the value of the state through the value of the next state and form an iterative formula, known in literature as TD(0) formula

$$\begin{aligned} V(s_t) &\leftarrow V(s_t) + \alpha \delta_t \\ \delta_t &= r_{t+1} + \gamma V(s_{t+1}) - V(s_t) \end{aligned} \quad (3)$$

where δ_t is the TD error, α is the learning step size, γ is the reward discount rate.

Since the state space of optimal control problems is usually large or infinite in practice, function approximators such as linear function approximators with basis function $\Phi(s) = (\phi(s_1), \phi(s_2), \dots, \phi(s_n))^T$ can be used to approximate the value function [14], [20]

$$V_t(s) = \Phi_t(s)^T \theta_t \quad (4)$$

where $\theta_t = (\theta_1, \theta_2, \dots, \theta_n)^T$ is the weight vector. The TD(0) learning rule for linear function approximator becomes

$$\theta_{t+1} = \theta_t + \alpha(r_{t+1} + \gamma \Phi_{t+1}^T \theta_t - \Phi_t^T \theta_t) \Phi_t. \quad (5)$$

TD methods can be viewed as gradient descent (hill climbing) in the space of the modifiable parameters (weights) [8]. That is, TD aims to minimize an error measure $J(\theta)$

$$J(\theta_t) = \sum_{k=1}^t [r_k - (\phi_k^T - \gamma \phi_{k+1}^T) \theta_t]^2 \quad (6)$$

over the space of weights by repeatedly incrementing the weight vector θ_t along the direction in which $J(\theta)$ decreases most steeply. Denoting with $\nabla_\theta J(\theta)$, the gradient of $J(\theta)$ with respect to θ , (5) can be written as

$$\theta_{t+1} = \theta_t + \alpha \Delta \theta_t, \quad \Delta \theta_t = -\alpha \nabla_\theta J(\theta_t). \quad (7)$$

A. The eligibility trace mechanism

Eligibility traces are one of the basic mechanisms of reinforcement learning [21]. There is no clear interpretation of the eligibility trace mechanism: a popular interpretation [1] is that the eligibility trace is a temporary record of the occurrence of an event, such as the visiting of a state or the taking of an action. The trace marks the memory parameters associated with the event as eligible for undergoing learning changes. When a TD error occurs, only the eligible states or actions are assigned credit or blame for the error.

Temporal difference can be combined with eligibility trace, resulting in the TD(λ) algorithm:

$$\begin{aligned} \theta_{t+1} &= \theta_t + \alpha(r_{t+1} + \gamma \Phi_{t+1}^T \theta_t - \Phi_t^T \theta_t) e_t, \\ e_t &= \gamma \lambda e_{t-1} + \delta_t \end{aligned} \quad (8)$$

where λ is a parameter that can be adjusted artificially, $\lambda \in [0, 1]$. When $\lambda = 1$, the value change rate of each state in the past has a complete impact on θ_{t+1} , which can be regarded as Monte Carlo process; when $\lambda = 0$, the value change rate of each state in the past has no impact on θ_{t+1} , which can be regarded as TD(0) process; when $\lambda \in (0, 1)$, each state in the past has an impact on adjustment θ_{t+1} in a declining trend, the greater λ , the greater the impact.

A convenient form of TD(λ) that exploits linear function approximator [9] is shown in Algorithm 1.

On each transition, the algorithm computes the scalar one-step TD error $r_t + (\gamma \Phi(s_{t+1}) - \Phi(s_t))^T \theta_t$, and apportions that error among all state features according to their respective eligibilities e_t .

B. Least-squares with eligibility trace

Boyan [13], pointed out some shortcomings of TD(λ) algorithm and put forward an improved method — Least-Squares TD(λ) (LS-TD(λ)). As compared to TD(λ), LS-TD(λ) builds from experience matrix A (of dimension $K \times K$, where K is the number of features) and vector b (of dimension K):

$$b = \sum_{i=0}^L e_i r_i \quad A = \sum_{i=0}^L e_i (\gamma \Phi(s_{i+1}) - \Phi(s_i))^T. \quad (9)$$

After n independent trajectories have been observed, θ_λ can be estimated as $A^{-1}b$. The complete LS-TD(λ) algorithm is specified in Algorithm 2.

Algorithm 1: TD(λ) algorithm:

Given: (1)a simulation model for a proper policy π in MDP $\mathcal{S}$;
(2)a featurizer $\Phi : \mathcal{S} \rightarrow \mathbb{R}^K$ mapping states to feature vectors, $\Phi(END) \stackrel{def}{=} 0$;
(3)a parameter $\lambda \in [0, 1]$;
(4)learning step size: α ;
Output: a coefficient vector θ for which $V_\pi(s) \approx \phi^T(s)\theta$.
Set $\theta := 0$ (or an arbitrary initial estimate), $t := 0$.
for $n:=1,2,\dots$ **do**
 Set $\delta:=0$.
 Choose a start state $s_t \in \mathcal{S}$.
 Set $e_t := \phi(s_t)$.
 while $s_t \neq END$, **do**
 Simulate one step of the process, producing a reward r_t and next state s_{t+1} .
 Set $\delta := \delta + (r_t + (\gamma\phi(s_{t+1}) - \phi(s_t))^T \theta)$.
 Set $e_{t+1} := \gamma\lambda e_t + \phi(s_{t+1})$.
 Set $t := t + 1$.
 Set $\theta := \theta + \alpha\delta e$

Algorithm 2: LS-TD(λ) algorithm:

Given: (1)a simulation model for a proper policy π in MDP $\mathcal{S}$;
(2)a featurizer $\Phi : \mathcal{S} \rightarrow \mathbb{R}^K$ mapping states to feature vectors, $\Phi(END) \stackrel{def}{=} 0$;
(3)a parameter $\lambda \in [0, 1]$;
Output: a coefficient vector θ for which $V_\pi(s) \approx \phi^T(s)\theta$.
Set $A := 0$, $b := 0$, $t := 0$.
for $n:=1,2,\dots$ **do**
 Choose a start state $s_t \in \mathcal{S}$.
 Set $e_t := \phi(s_t)$.
 while $s_t \neq END$, **do**
 Simulate one step of the chain, producing a reward r_t and next state s_{t+1} .
 Set $A := A + e_t(\phi(s_t) - \gamma\phi(s_{t+1}))^T$.
 /* outer product */
 Set $b := b + e_t r_t$.
 Set $e_{t+1} := \gamma\lambda e_t + \phi(s_{t+1})$.
 Set $t := t + 1$.
 Whenever updated coefficients are desired: Set $\theta := A^{-1}b$

When $\lambda = 0$, LS-TD(0) reduces to Bradtke and Barto's LS-TD algorithm, which they derived using regression with instrumental variables [11].

C. Recursive least squares with eligibility trace

Least-squares methods require the computation of a matrix inverse at each time step, i.e. they have a computational complexity of $O(K^3)$, assuming that the state representations are of length K . Recursive Least-Squares (RLS) techniques have been used in [11] to derive a modified algorithm, RLS-TD(λ), with computational complexity of $O(K^2)$.

$$\theta_t = \theta_{t-1} + \frac{P_{t-1}}{1 + (\phi_t - \gamma\phi_{t+1})^T P_{t-1} e_t} \delta_t e_t \quad (10)$$

$$P_t = P_{t-1} - \frac{P_{t-1} e_t (\phi_t - \gamma\phi_{t+1})^T P_{t-1}}{1 + (\phi_t - \gamma\phi_{t+1})^T P_{t-1} e_t} \quad (11)$$

$$\delta_t = r_t - (\phi_t - \gamma\phi_{t+1})^T \theta_{t-1} \quad (12)$$

where the e_t is the eligibility trace, updated recursively according to

$$e_{t+1} = \gamma\lambda e_t + \phi_{t+1}. \quad (13)$$

It is recognized in literature that there are no clear rules to select λ , and the value of λ should be selected appropriately to obtain the best approximation error [22].

IV. AN INSTRUMENTAL VARIABLE PERSPECTIVE OF RECURSIVE TEMPORAL DIFFERENCE

A. Gradient structure of TD algorithm

Going back to the standard TD method (5) for a while, [8] showed that the learning rule for TD(λ) (i.e. with eligibility

trace mechanism) can also be written as a gradient update

$$\begin{aligned} \theta_{t+1} &= \theta_t + \alpha [r_t + \gamma V_t(s_{t+1}) - V_t(s_t)] \sum_{k=1}^t \lambda^{t-k} \nabla_{\theta_t} V_t(s_k) \\ &= \theta_t + \alpha \Delta \theta_t \end{aligned} \quad (14)$$

where

$$\Delta \theta_t = [r_t + \gamma V_t(s_{t+1}) - V_t(s_t)] \sum_{k=1}^t \lambda^{t-k} \nabla_{\theta_t} V_t(s_k). \quad (15)$$

It is worth noticing that the summation in term in (15) essentially keeps $\Delta \theta_t$ fixed, whereas it goes through all the recently visited states s_k . This is an attempt to separate the effects of changing the parameters from the effects of moving through the state space [11]. That is, (14) reveals a similar structure as the LS cost (6) where θ_t is fixed and the data $\hat{\omega}_k$ change in the past [23]. The problem that we want to investigate in the following is to find the corresponding version of (6) for the update (14)-(15).

In order to explain this point, let us consider the goal of linear least-squares approximation of some function $R : \mathbb{R}^n \rightarrow \mathbb{R}$, given samples of $\omega_t \in \mathbb{R}^n$ and outputs $r_t \in \mathbb{R}$. If only the observations r_t are corrupted by noise, then we have the following situation:

$$r_t = \omega_t^T \theta^* + \eta_t \quad (16)$$

where θ^* is the vector of true (but unknown) parameters and η_t is the output observation noise. Given the minimization of (6) (assuming $\omega_t = \phi_t - \gamma\phi_{t+1}$), it is well known that the optimal θ_t can be found by setting the partial derivative

of J_t with respect to θ_t equal to zero,

$$\theta_t = [\sum_{k=1}^t \omega_k \omega_k^T]^{-1} [\sum_{k=1}^t \omega_k r_k]. \quad (17)$$

Relation (19) models the situation in which observation errors occur only on r_t . If the input observations ω_t are also noisy, the following errors-in-variables situation [12] applies:

$$r_t = \hat{\omega}_t \theta^* - \zeta_t^T \theta^* + \eta_t. \quad (18)$$

Substituting $\hat{\omega}_t$ directly for ω_t in (17) has the effect of making the noise dependent upon the current state. This introduces a bias, and θ_t no longer converges to θ^* [12], [11]. One way around this problem is to introduce an instrumental variables ϕ_t , i.e. a vector that is correlated with the input ω_t , but uncorrelated with the observation noise ζ_t . The following equation is a modification of (17) that uses the instrumental variables and the noisy inputs:

$$\theta_t = [\sum_{k=1}^t \phi_k \hat{\omega}_k^T]^{-1} [\sum_{k=1}^t \phi_k r_k]. \quad (19)$$

It was shown in [11] that LS-TD(0) can be interpreted using the instrumental variable method with $\hat{\omega}_k = \phi_k - \gamma \phi_{k+1}$. In the following, we will use the instrumental variable method to provide a new perspective into the LS-TD(0), LS-TD(λ) and their recursive versions.

B. Instrumental variable method as a minimizer

Let us first derive LS-TD(0) and RLS-TD(0) according to the instrumental variable method. This perspective is new as it provides an optimization cost for the RLS-TD(0) which has not appeared in previous TD literature. Let us consider the following minimizer in line with the LS-TD(0) in [11]

$$\theta = (\Phi_k^T \hat{\Omega}_k + P_0^{-1})^{-1} (P_0^{-1} \theta_0 + \Phi_k^T r_k) \quad (20)$$

where $\Phi_k = [\phi_1, \phi_2, \dots, \phi_k]$ is the instrumental variable vector, $\hat{\Omega}_k = [\hat{\omega}_1, \hat{\omega}_2, \dots, \hat{\omega}_k]$ is the regressor vector, $\hat{\omega}_k = \phi_k - \gamma \phi_k$, $r_k = [r_1, r_2, \dots, r_k]$ is the reward vector and $\theta = [\theta_1, \theta_2, \dots, \theta_k]$ is the weight vector. The parameters P_0 and θ_0 will be defined later.

We want to find the cost corresponding to the instrumental variable minimizer (20). Because (20) is a minimizer, it must make the gradient of an appropriate cost function J equal to 0. From reverse calculation, we can get that the cost function

$$J(\theta) = \frac{1}{2} (r_k - \Phi_k \theta)^T (r_k - \Phi_k \theta) - \frac{1}{2} (\Phi_k \theta)^T [(\Phi_k - \hat{\Omega}_k) \theta] + \frac{1}{2} (\theta - \theta_0)^T P_0^{-1} (\theta - \theta_0). \quad (21)$$

Instead of using (20) to calculate the minimizer, we want to do it in a recursive way [23] in order to update the minimizer when new data arrive,

$$P_k = P_{k-1} - \frac{P_{k-1} \phi_k \hat{\omega}_k^T P_{k-1}}{m_k^2 + \hat{\omega}_k^T P_{k-1} \phi_k} \quad (22)$$

$$\theta_k = \theta_{k-1} + P_k \phi_k \left[\frac{r_k - \theta_{k-1}^T \hat{\omega}_k}{m_k^2} \right]. \quad (23)$$

where $m_k^2 \geq 1$ is the normalizing signal designed to bound ϕ_k from above.

C. Proposed method

Let us now consider the cost

$$J(\theta) = \frac{1}{2} \sum_{k=1}^t a_k \beta^{t-k} \frac{[r_k - \theta_k^T \phi_k]^2}{m_k^2} - \frac{1}{2} \sum_{k=1}^t a_k \beta^{t-k} \frac{(\phi_k \theta_k^T)^T [(\phi_k - \hat{\omega}_k) \theta_k^T]}{m_k^2} + \frac{1}{2} \beta^t (\theta_k - \theta_0)^T P_0^{-1} (\theta_k - \theta_0) \quad (24)$$

where a_k is a non-negative sequence of weighting coefficients, $\beta \in (0, 1)$ is a design sequence.

Remark 1: Differently from (21), the cost (24) has a forgetting factor to strengthen the influence of current data and reduce the influence of historical data. This is essentially the same mechanism in (14)-(15). Therefore, cost (24) can be interpreted as the extension to (6) in the eligibility trace sense.

The algorithm for generating θ_k , the estimate of θ^* , is (derivation follows along similar lines as [23])

$$P_k = \frac{1}{\beta} [P_{k-1} - \frac{a_k P_{k-1} \phi_k \hat{\omega}_k^T P_{k-1}}{m_k^2 \beta + a_k \hat{\omega}_k^T P_{k-1} \phi_k}] \quad (25)$$

$$\theta_k = \theta_{k-1} + \sqrt{a_k} P_k \phi_k \left[\frac{r_k - \theta_{k-1}^T \hat{\omega}_k}{m_k^2} \right]. \quad (26)$$

It is worth noticing the differences and similarities between the RLS-TD(λ) (10)-(13) and the proposed RLS-TD with forgetting factor (25)-(26). Such methods will now be compared experimentally.

V. VALIDATION

The balancing control of inverted pendulums is a typical nonlinear control problem, widely studied not only in control theory but also in artificial intelligence [1]. Fig. 2 shows the cart-pole balancing control system, which consists of a cart moving horizontally and a pole with one end fixed at the cart. The variable x is the horizontal position of the cart. The variable ϑ is the angle of the pole with respect to the vertical ($\vartheta = \pi$ is the upright position) and F is force applied to the cart. The dynamics of the system can be described by

$$\ddot{x} = \frac{1}{m_c + m_p \sin^2 \vartheta} [F + m_p \sin \vartheta (l \dot{\vartheta}^2 + g \cos \vartheta)] \quad (27)$$

Fig. 2: The cart-pole balancing control system

$$\ddot{\vartheta} = \frac{1}{l(m_c + m_p \sin^2 \vartheta)} \times [-F \cos \vartheta - m_p l \dot{\vartheta}^2 \cos \vartheta \sin \vartheta - (m_c + m_p)g \sin \vartheta] \quad (28)$$

where g is the acceleration due to the gravity, which is $9.81m/s^2$.

According to [1], we set the mass of the cart $m_c = 1.0kg$, the mass of the pole $m_p = 0.1kg$, the pole length $l = 0.5m$. The reward for the problem is defined as

$$r = -x^2 - \vartheta^2 - \dot{x}^2 - \dot{\vartheta}^2 - \rho F^2 \quad (29)$$

representing a quadratic cost. Where ρ is a design constant. In the simulation experiment, we set $\rho = 0.1$. Furthermore,

$$\Phi_t(s) = [x_t^2, \dot{x}_t^2, (\vartheta_t - \pi)^2, \dot{\vartheta}_t^2, x_t \dot{x}_t, x_t(\vartheta_t - \pi), \quad (30)$$

$$x_t \dot{\vartheta}_t, \dot{x}_t(\vartheta_t - \pi), \dot{x}_t \dot{\vartheta}_t, (\vartheta_t - \pi) \dot{\vartheta}_t]^T$$

$$\theta_t = [\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7, \theta_8, \theta_9, \theta_{10}]^T \quad (31)$$

representing a quadratic value function. We consider a Policy Iteration setting, i.e. improving the performance of an initially stabilizing controller. The initially stabilizing controller is selected according to θ is obtained by a linear quadratic regulator.

In RLS-TD(λ) algorithm, we set discount rate $\gamma = 0.95$, $\lambda = 0.8$, which are the standard values suggested in Sutton's and Barto's book. In RLS-TD with forgetting factor algorithm, we set discount rate $\gamma = 0.95$ (as in Sutton's and Barto's book), weighting coefficient $a_k = 1$, forgetting factor $\beta = 0.95$, and normalizing signal $m_k^2 = 1 + \Phi_t^T \Phi_t$. In both algorithms, we set $P_0 = 10I$. The initial condition of the system are: $x = 0; \dot{x} = 0; \vartheta = \pi + \pi/36$ (i.e. 185°); $\dot{\vartheta} = 0$.

Furthermore, in order to check if the algorithms are effectively learning, we "offline" apply the resulting estimate at time t and calculate the corresponding cost. This is represented in Fig. 3 and 4, where it can be seen that the offline reward is indeed improving. The proposed method seems to improve more as compared to the initial stabilizing controller. Also notice that the learning is more effective when the system states are far from the equilibrium: as the system approaches the equilibrium, the learning becomes smaller and might even become a bit worse. Dependence of learning on the quality of data is a well known phenomenon in learning schemes [11].

In the offline simulation experiment, the reward range of RLS-TD(λ) algorithm is $-0.04102 \sim -0.03708$, converging to -0.03735 ; the reward range of the new algorithm is $-0.04025 \sim -0.03687$, converging to -0.03697 .

Some important information about the experimental results is shown in the Table I.

TABLE I: Comparison of experimental results

	RLS-TD(λ)	RLS-TD-f
Integral norm of $\chi^T \chi$ (state)	23.36	21.58
Integral norm of ρF^2 (input)	38.40	33.00
Largest offline learning reward	-0.0371	-0.0369
Steps needed for offline learning convergence	767	687

In the table, the integral norm is the integral from 0 to t (in line with the linear quadratic regulator theory), and

Fig. 3: Offline reward of RLS-TD(λ).

Fig. 4: Offline reward of proposed RLS-TD-f.

$\chi = [x, \dot{x}, \vartheta, \dot{\vartheta}]^T$. It can be seen that the cost of the RLS-TD with forgetting factor algorithm is typically smaller.

In order to validate these benefits over a wider portion of the state space, we select four groups of initial conditions according to the following combinations of initial conditions.

The set of values of initial position of the cart (i.e. x) is $-2m, -1.5m, \dots, 1.5m, 2m$. There are 9 values from $-2m$ to $2m$, and the step span is $0.5m$. The set of values of initial angle of the pole from the vertical (i.e. ϑ) is $-5^\circ, -3.75^\circ, \dots, 3.75^\circ, 5^\circ$. There are 9 values from -5° to 5° , and the step span is 1.25° . The set of values of initial speed of the cart (i.e. $\dot{x}$) is $-1m/s, -0.75m/s, \dots, 0.75m/s, 1m/s$. There are 9 values from $-1m/s$ to $1m/s$, and the step span is $0.25m/s$. The set of values of initial angular speed of the pole (i.e. $\dot{\vartheta}$) is $-10^\circ/s, -7.5^\circ/s, \dots, 7.5^\circ/s, 10^\circ/s$. There are 9 values from $-10^\circ/s$ to $10^\circ/s$, and the step span is $2.5^\circ/s$.

All these initial conditions are combined into 4 groups of $9 \times 9 = 81$ initial conditions. In order to compare the approaches, we count how many times each algorithm overcomes the other one in terms of offline final reward and offline convergence. The results are summarized in Table II. The table shows that the proposed algorithm has a better improvement of the initially stabilizing performance in 70.4% of the cases, but only in 32.4% of the cases, it achieves the convergence effect. Remarkably, the better reward is achieved while obtaining better online regulation and control effort (smaller norm of the state in 53.1% of the

cases and smaller norm of the control input in 57.1% of the cases): this can partially explain the longer convergence

time, since it is known in parameter estimation literature that smaller signals will make convergence slower.

TABLE II: Comparison results for each group of experiments.

	Final reward		Convergence		Norm state		Norm input	
	RLS-TD(λ)	RLS-TD-f	RLS-TD(λ)	RLS-TD-f	RLS-TD(λ)	RLS-TD-f	RLS-TD(λ)	RLS-TD-f
1st group	15/81 (18.5%)	66 /81 (81.5%)	62 /81 (76.5%)	19/81 (23.5%)	31/81 (38.3%)	50 /81 (61.7%)	38/81 (46.9%)	43 /81 (53.1%)
2nd group	30/81 (37.0%)	51 /81 (63.0%)	60 /81 (74.1%)	21/81 (25.9%)	41 /81 (50.6%)	40/81 (49.4%)	54 /81 (66.7%)	27/81 (33.3%)
3rd group	19/81 (23.5%)	62 /81 (76.5%)	58 /81 (71.6%)	23/81 (28.4%)	20/81 (24.7%)	61 /81 (75.3%)	34/81 (42.0%)	47 /81 (58.0%)
4th group	32/81 (39.5%)	49 /81 (60.5%)	39/81 (48.1%)	42 /81 (51.9%)	60 /81 (74.1%)	21/81 (25.6%)	13/81 (16.0%)	68 /81 (84.0%)
Total	96/324 (29.6%)	228 /324 (70.4%)	219 /324 (67.6%)	105/324 (32.4%)	152/324 (46.9%)	172 /324 (53.1%)	139/324 (42.9%)	185 /324 (57.1%)

VI. CONCLUSION

In this work we have shown that the forgetting factor commonly used in least-squares algorithm has a similar role to the eligibility trace in the TD algorithm. We adopted an instrumental variable perspective to illustrate this point and we proposed a new algorithm, namely — RLS-TD with forgetting factor. The purpose of introducing forgetting factor is to give different weights to the original data and the new data, so that the algorithm can respond to the changes of input process. Consequently, the similarity between forgetting factor and eligibility trace is that the influence of the past data on the current value is considered to some extent. Using extensive experiments on a cart-pole benchmark, we have shown that the proposed algorithm can better improve the initially stabilizing control performance: remarkably, this is done while achieving better online regulation and control effort.

REFERENCES

- [1] Richard S Sutton and Andrew G Barto. Reinforcement learning: an introduction cambridge. MA: MIT Press.[Google Scholar], 1998.
- [2] Dimitri P Bertsekas. *Reinforcement learning and optimal control*. Athena Scientific, 2019.
- [3] G. Rishwaraj, S. G. Ponnambalam, and C. K. Loo. Heuristics-based trust estimation in multiagent systems using temporal difference learning. *IEEE Transactions on Cybernetics*, 47(8):1925–1935, 2017.
- [4] Gerald Tesauro. Practical issues in temporal difference learning. In *Advances in neural information processing systems*, pages 259–266, 1992.
- [5] C. Keerthisinghe, G. Verbič, and A. C. Chapman. A fast technique for smart home management: Adp with temporal difference learning. *IEEE Transactions on Smart Grid*, 9(4):3291–3303, 2018.
- [6] J. Dragoun and V. Šmídl. Adaptive control of lcl filter with time-varying parameters using reinforcement learning. In *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*, volume 1, pages 267–272, 2019.
- [7] Z. Chen, L. Li, X. Hu, B. Yan, and C. Yang. Temporal-difference learning-based stochastic energy management for plug-in hybrid electric buses. *IEEE Transactions on Intelligent Transportation Systems*, 20(6):2378–2388, 2019.
- [8] Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.
- [9] John N Tsitsiklis and Benjamin Van Roy. Analysis of temporal-difference learning with function approximation. In *Advances in neural information processing systems*, pages 1075–1081, 1997.
- [10] Peter Dayan. The convergence of td (λ) for general λ . *Machine learning*, 8(3-4):341–362, 1992.
- [11] Steven J Bradtke and Andrew G Barto. Linear least-squares algorithms for temporal difference learning. *Machine learning*, 22(1-3):33–57, 1996.
- [12] Peter C Young. *Recursive estimation and time-series analysis: an introduction*. Springer Science & Business Media, 2012.
- [13] Justin A Boyan. Technical update: Least-squares temporal difference learning. *Machine learning*, 49(2-3):233–246, 2002.
- [14] Xin Xu, Han-gen He, and Dewen Hu. Efficient reinforcement learning using recursive least-squares methods. *Journal of Artificial Intelligence Research*, 16:259–292, 2002.
- [15] William Dabney and Philip S. Thomas. Natural temporal difference learning. In *Twenty-eighth Aaai Conference on Artificial Intelligence*, 2014.
- [16] Sam Blakeman and Denis Mareschal. A complementary learning systems approach to temporal difference learning. *arXiv: Neurons and Cognition*, 2019.
- [17] A. Ramaswamy and S. Bhatnagar. Stability of stochastic approximations with “controlled markov” noise and temporal difference learning. *IEEE Transactions on Automatic Control*, 64(6):2614–2620, 2019.
- [18] X. Niu, J. Xu, Q. Ren, and Q. Wang. Locomotion learning for an anguilliform robotic fish using central pattern generator approach. *IEEE Transactions on Industrial Electronics*, 61(9):4780–4787, 2014.
- [19] K. Yeh, I. Wu, C. Hsueh, C. Chang, C. Liang, and H. Chiang. Multi-stage temporal difference learning for 2048-like games. *IEEE Transactions on Computational Intelligence and AI in Games*, 9(4):369–380, 2017.
- [20] Adithya M Devraj and Sean P Meyn. Differential td learning for value function approximation. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 6347–6354. IEEE, 2016.
- [21] 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.
- [22] Biao Yin, Mahjoub Dridi, and Abdellah El Moudni. Recursive least-squares temporal difference learning for adaptive traffic signal control at intersection. *Neural Computing and Applications*, 31(2):1013–1028, 2019.
- [23] Petros Ioannou and Barış Fidan. *Adaptive control tutorial*. SIAM, 2006.