Consistent Emphatic Temporal-Difference Learning

by

Jiamin He

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

University of Alberta

Abstract

Off-policy policy evaluation has been a critical and challenging problem in reinforcement learning, and Temporal-Difference (TD) learning is one of the most important approaches for addressing it. There has been significant interest in searching for off-policy TD algorithms which find the same solution that would have been obtained in the on-policy regime. An important property of these algorithms is that their expected update has the same fixed point as that of On-policy TD(λ), which we call *consistency*. Notably, Full IS TD(λ) is the only existing consistent off-policy TD method under general linear function approximation but, unfortunately, has a high variance and is scarcely practical. This notorious high variance issue motivates the introduction of ETD(λ), which tames down the variance but has a biased fixed point. Inspired by these two methods, we propose a new consistent algorithm called *Average Emphatic TD* (AETD(λ)) with a transient bias, which strikes a balance between bias and variance. Further, we unify AETD(λ) with existing algorithms and obtain a new family of consistent algorithms called *Consistent Emphatic TD* (CETD(λ, β, ν)), which can control a smooth bias-variance trade-off by varying the speed at which the transient bias fades. Through theoretical analysis and experiments on a didactic example, we settle the consistency of CETD(λ, β, ν) and demonstrate this theoretical advantage empirically. Moreover, we show that CETD(λ, β, ν) converges faster to the lowest error in a complex task with a high variance.

Preface

Significant portions of this thesis are based on a paper accepted to the UAI 2023 conference. Parts of the content in Chapter 4 are built upon the results presented at the Deep Reinforcement Learning (RL) Workshop at the NeurIPS 2022 conference.

This work was in collaboration with Fengdi Che, Yi Wan, and Rupam Mahmood. The project started with a discussion about the emphatic temporal-difference (TD) learning method for the average-reward setting between Yi and me. During the discussion, we decided to investigate the *average followon trace*. Subsequently, I proved the consistency of the new algorithm that use this trace. After some preliminary results were accepted to the Deep RL Workshop at NeurIPS 2022, Fengdi suggested using a more flexible followon trace that has a sublinear form with a parameter to control the sublinearity. Building on this trace and with help from Rupam, I developed the general followon trace and General Emphatic TD (GETD(β, ν)) for the discounted setting. GETD(β, ν) unifies several existing approaches and includes *Consistent Emphatic TD* (CETD(β, ν)). I settled the stability and consistency analysis of CETD(β, ν). Later, I extended the results to the multi-step bootstrapping case and developed GETD(λ, β, ν), which includes CETD(λ, β, ν). Fengdi, Yi, and I proposed to empirically investigate CETD1(λ, β), CETD3(λ, β), and CETD2(λ, ν), the three instances of CETD(λ, β, ν) respectively. The empirical evaluations of these instances were designed and performed by me. The bias-variance analysis was initiated by Fengdi, which was later redone and improved by me. During the whole process, Rupam advised the project and was crucially involved in most aspects of this work. Other collaborators also contributed valuable feedback across various dimensions.

The collaborative nature of this work required the use of “we” as the first person pronoun in writing this thesis. Nevertheless, I bear sole accountability for any technical or presentation errors.

You have power over your mind - not outside events.
Realize this, and you will find strength.

MARCUS AURELIUS

Acknowledgements

I would like to express my sincerest gratitude to my supervisor Rupam Mahmood, who advised the project leading to this thesis and gave me tremendous support. I have learned numerous lessons from him, including how to think in a disciplined way, organize my thoughts, and communicate them with others. His devotion to artificial intelligence (AI) research has also inspired and encouraged me to pursue AI research as a career goal.

I owe many thanks to my collaborators, Yi Wan and Fengdi Che. Yi helped me initiate the project and also gave me treasured advice on both research and life. Fengdi shared with me a key insight that reinvigorated the project as well as introduced me to skiing, the best winter sport, in my opinion. The countless hours they spent with me discussing the project and their valuable feedback on the drafts have taught me how to do research and made this project possible and fun.

I am profoundly grateful to Richard Sutton, who took me on the journey of reinforcement learning and spent many hours discussing research with me. His philosophy of AI has inflamed my passion for understanding intelligence and ourselves. I am also deeply grateful to Csaba Szepesvári, who introduced me to stoicism, a life-changing philosophy. His kindness and wisdom of stoicism have made the whole journey less stressful and more enjoyable.

I want to thank Marlos Machado and James Wright for serving on my examining committee. Marlos's feedback on the draft of this thesis has been invaluable in improving the thesis. I am also grateful to Martha White and Huizhen Yu for discussing research problems with me. The discussion with them has always been insightful and valuable. I also want to thank Tadashi Kozuno, Shivam Garg, Sina Ghiassian, Shangtong Zhang, and the Digital Research Alliance of Canada. Without them, the experiments for this thesis may take many more years to complete.

I also owe an outstanding debt of gratitude to all my friends in the RLAI lab, Amii, and the University of Alberta for creating a supportive environment and helping me in countless ways.

Finally, I would like to thank my family for their unconditional love and support. And my greatest gratitude goes to my fiancée Ziying Huang. It is her encouragement, support, and love that made this journey possible, enjoyable, and memorable.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Understanding and creating intelligence has been a long-standing challenge over decades. A fundamental characteristic of any intelligent agent is learning: the process of acquiring new knowledge, skills, and abilities. Learning to swim is a great example. Swimming is not innate to humans, so it requires learning and practice to master. Through learning, we can gradually acquire knowledge about the physics of water, skills to coordinate our body movements, and the ability to swim confidently. Similar to the process of learning to swim, we acquire new knowledge, skills, and abilities on a daily basis, from writing in various styles to experimenting with new recipes in the kitchen.

Learning can occur through various signals and in different ways. Continuing with our example of learning to swim, we may learn through trial and error using reward or punishment signals. For instance, we will be rewarded if our movements help us stay afloat and move forward, or punished otherwise. We may also learn by imitating the movements of other swimmers, which allows us to quickly adapt to the correct technique. Additionally, we may benefit from feedback provided by trainers or instructors, which can help us improve our swimming skills.

Of the different forms of learning, trial-and-error learning through reward signals is a foundational way in which we learn to behave. Reinforcement learning is a theoretical framework that focuses on the computational aspect of this type of learning, studying how an agent can learn to take actions in an environment to maximize a reward signal. Research has shown that biological processes in animals and humans are consistent with the principles of reinforcement learning (Lee et al., 2012). The powerful formulation and biological consistency of reinforcement learning have secured its important role in various models of natural and artificial intelligence.

A significant and powerful feature of natural intelligent agents' learning is the ability to learn counterfactual knowledge, that is, to estimate the outcomes of hypothetical scenarios or "counterfactuals" that did not happen in reality. Whether it's athletes analyzing their performance, students reflecting on their exams, or chefs experimenting with new dishes, individuals often consider how things could have been different to make better decisions. This reflection involves imagining different scenarios and evaluating the potential outcomes, leading to a deeper understanding of the choices made and the impact of those choices. By utilizing counterfactual learning, individuals can make better decisions in the future, as well as evaluate the effectiveness of past decisions.

In the context of reinforcement learning, off-policy learning is a key form of counterfactual learning. It allows an agent to optimize a greedy policy while simultaneously executing a more explorative one, and also to estimate the value of policies that differ from the one currently being executed. The former is categorized as an off-policy control problem, while the latter is considered an off-policy prediction problem.

In this thesis, we focus on online off-policy prediction, in which the agent will just estimate the values of a fixed policy while following another fixed policy without storing any samples. This simple setting is an important step in developing techniques and understanding that are useful to building intelligent agents. We present a new algorithm that fills a gap in the online off-policy prediction literature: existing algorithms either are not practical or introduce persistent bias, while our method is practical and consistent, that is, has only transient bias that will fade away. Further, we unify our algorithm with several existing algorithms and obtain a more general algorithm with improved practicality while also building a stronger bridge between existing algorithms. Finally, we performed extensive studies on two examples, revealing insights into the properties and advantages of our algorithm, the first algorithm of its kind to have both practicality and consistency in the setting we study.

1.1 Problem Statement

Off-policy learning is a critical area in reinforcement learning (RL). Particularly, off-policy policy evaluation (OPPE), also known as off-policy prediction, is an essential component in model learning, options learning (Sutton et al., 1999), and life-long learning (Sutton et al., 2022; White et al., 2012). The goal of OPPE is to estimate the value function of a *target policy* with data collected by a different *behavior policy*. We refer to the data collected by the target policy as *on*-policy data and the data collected by the behavior policy *off*-policy data. In this thesis, we consider the problem of *online OPPE with linear function approximation*.

In the online setting, algorithms make learning updates while simultaneously collecting data from the environment. In this thesis, we consider a specific type of algorithm with a strict computational goal, where both memory usage and per-sample computation remain constant regardless of the number of samples. Such algorithms are known as strictly-incremental algorithms (Mahmood, 2017). Temporal-Difference (TD) learning is a widely used family of algorithms that can meet this computational goal, with On-policy TD(λ) being an essential approach to online on-policy prediction (Sutton, 1988).

In online OPPE, significant efforts have been made to obtain the *on-policy fixed point*, to which On-policy TD(λ) converges with on-policy data (Precup et al., 2001; Hallak and Mannor, 2017; Zhang et al., 2020b). The on-policy fixed point is targeted because it can produce a good approximation of the target policy's value function (Tsitsiklis and Van Roy, 1996). Subsequently, we say that an off-policy TD algorithm is *consistent* if it has the same fixed point as the on-policy fixed point. For example, when the feature representation is tabular, Off-policy TD(λ), which is the extension of On-policy TD(λ) to the off-policy case, is consistent (Precup, 2000).

However, in the case of function approximation, Off-policy TD(λ) is not consistent and has been shown to be unstable and divergent in various counterexamples (Baird, 1995; Tsitsiklis and Van Roy, 1996; Sutton and Barto, 2018). In general, when TD learning, off-policy learning, and function approximation are combined, it can lead to the issue of divergence, which is commonly known as *the deadly triad*. Furthermore, the *off-policy fixed point* that Off-policy TD(λ) converges to, if it does, has no known guarantee on its approximation error to the true value function. Instead, it has been shown that it could induce an unbounded error when $\lambda = 0$ (Kolter, 2011).

In contrast, not only are consistent off-policy TD algorithms guaranteed to be stable, but their fixed point also has a benign error bound, which is the on-policy fixed point mentioned above. Therefore, *in this thesis, we pursue consistent off-policy TD algorithms for addressing the problem of online OPPE with linear function approximation.*

Full Importance-Sampling TD (Full IS TD(λ); Precup et al., 2001) is the only consistent off-policy TD algorithm in the linear function approximation case without making any significant assumption, while others require additional assumptions on the structure of the problem, especially the feature representation (Hallak and Mannor, 2017; Liu et al., 2018; Nachum et al., 2019; Zhang et al., 2020a,b). To obtain the on-policy fixed point, Full IS TD(λ) reweights the TD update with a full importance-sampling-ratio (IS-ratio) product, which is the multiplication of the IS ratios at every time step. However, Full IS TD(λ) barely works in practice due to the high variance of the full IS-ratio product.

Following Full IS TD(λ)'s idea of calibrating the fixed point of TD by reweighting the TD update, Emphatic TD (ETD(λ); Sutton et al., 2016) uses an alternative emphatic weighting, which is a geometrically weighted sum of IS-ratio products accumulated from every time step. While the IS-ratio products with fewer IS-ratio terms do mitigate the variance issue, however, they also induce a persistent bias, deviating ETD(λ)'s fixed point from the on-policy TD fixed point. Further, to obtain a smooth bias-variance trade-off, Hallak et al. (2016) proposed ETD(λ, β), which unifies Off-policy TD(λ) and ETD(λ) with a tunable parameter β. Yet, ETD(λ, β) will lose the stability guarantee when β is smaller than an instance-dependent condition number that is difficult to determine.

1.2 Contributions

In this thesis, we propose *Consistent Emphatic TD* (CETD(λ, β, ν)). CETD(λ, β, ν) is the first practical, consistent algorithm under general linear function approximation, filling a gap in off-policy policy evaluation: existing algorithms are either impractical or inconsistent in this setting. We have identified this gap in the literature and addressed it through a series of new algorithms:

- We first propose *one-step Average Emphatic TD* (AETD(0)), a new consistent algorithm inspired by Full IS TD(λ) and ETD(λ) that strikes a better balance between bias and variance (see Chapter 4). AETD(0) renovates the idea of ETD(λ), introducing a transient bias in its followon trace to achieve a lower variance than Full IS TD(λ) while retaining consistency as the bias fades away over time.

- Then we unify AETD(0) with existing algorithms by introducing extra hyperparameters to control a smooth bias-variance trade-off and extend the unification to the multi-step bootstrapping case (see Chapter 5). The unified algorithm called *General Emphatic TD* (GETD(λ, β, ν)) not only provides the first proper connection between existing algorithms, including Off-policy TD(λ), Full IS TD(λ), and ETD(λ, β) but also brings out CETD(λ, β, ν), the first practical, consistent algorithm under general linear function approximation.

- We prove the stability and consistency of CETD(λ, β, ν) and propose its three instances for convenient empirical evaluation and practical use (see Chapter 6).

- Through experiments on a didactic example and a more complex task with high variance, we demonstrate the benefit of CETD(λ, β, ν)'s consistency and its practicality (see Chapter 7).

Chapter 2

Background

In this chapter, we first define infinite horizon Markov decision processes, a mathematical framework commonly used to model decision-making problems. After that, we provide an introduction to temporal-difference (TD) learning, an influential and ubiquitous family of algorithms in RL. We then formally describe the problem we tackle in this thesis, online off-policy policy evaluation (OPPE) with linear function approximation. Finally, we discuss the notorious instability issue of the vanilla off-policy TD method as well as introduce the concept of consistency, a valuable property of off-policy TD algorithms for solving OPPE.

2.1 Infinite Horizon Markov Decision Processes

In different literature on sequential decision making, *Markov Decision Processes* (MDPs) are a convenient mathematical framework. In this thesis, we focus on MDPs with an infinite horizon, which are capable of modeling decision-making problems over extended periods of time. An infinite horizon MDP is defined as a tuple $\langle \mathcal{S}, \mathcal{A}, p, d_0, r, \gamma \rangle$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $p : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition function, $d_0 \in \Delta(\mathcal{S})$ is the initial state distribution, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the deterministic reward function[1], and $0 \leq \gamma < 1$ is the discount factor. Here, $\Delta(\mathcal{X})$ denotes the set of all possible probability distributions over a set $\mathcal{X}$. For mathematical convenience, we only consider MDPs with finite state and action spaces[2].

In RL, the problem defined by an MDP is usually called the *environment*, and the intelligent *agent* will interact with it through a policy defined as $\pi : \mathcal{S} \to \Delta(\mathcal{A})$. The agent begins its interaction with the environment starting from observing an initial state S_0 drawn from the initial

[1] For ease of presentation, we assume the reward function to be deterministic, and all the results can go through in the general case with a stochastic reward function.

[2] Extending the results in this thesis to continuous state and action spaces may encounter significant technical challenges and is left for future work.

state distribution d_0. At each time step $t \geq 0$ ($t \in \mathbb{N}$), the agent selects an action, $A_t \sim \pi(S_t)$, based on the current observed state S_t. After sending action A_t to the environment, the agent receives the next state, $S_{t+1} \sim p(S_t, A_t)$, and the next reward, $R_{t+1} \doteq r(S_t, A_t)$. The agent then chooses the next action and observes the next state and reward, repeating the process iteratively.

Given an infinite trajectory,

$$S_0, A_0, R_1, S_1, A_1, R_2, S_2, \cdots, R_t, S_t, A_t, R_{t+1}, S_{t+1}, \cdots, \tag{2.1}$$

the *discounted return* at time step t is defined as

$$G_t \doteq R_1 + \gamma R_2 + \gamma^2 R_3 + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}. \tag{2.2}$$

As a performance metric of the agent, the *discounted value function* of its policy π is defined as the expected discounted return starting from the given state:

$$v_\pi(s) \doteq \mathbb{E}_\pi \left[G_0 | S_0 = s \right] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{k+1} | S_0 = s \right], \quad \text{for all } s \in \mathcal{S}.$$

When there is no ambiguity, we will drop the modifier "discounted" when referring to the discounted return and the discounted value function. For convenience, we will also identify the value function with a vector $\mathbf{v}_\pi = [v_\pi(s_1), \cdots, v_\pi(s_{|\mathcal{S}|})]^\top \in \mathbb{R}^{|\mathcal{S}|}$. In the control setting, the goal of the agent is to find an optimal policy π^*, which dominates all other policies, i.e., $v_{\pi^*}(s) \geq v_\pi(s)$ for all $s \in \mathcal{S}$ and all possible π. However, in this thesis, we will focus on the prediction setting, in which the agent will just estimate the value function of its fixed policy π.

When the agent possesses the knowledge of the transition and reward functions, it can determine the value function either analytically or through an iterative approach (Bellman, 2003). Both approaches utilize the *Bellman equation*, which is a fundamental equation that relates the value function to the reward and transition functions:

$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \left[r(s,a) + \sum_{s' \in \mathcal{S}} \gamma p(s'|s,a) v_\pi(s') \right], \quad \text{for all } s \in \mathcal{S},$$

where we denote the probability of taking a at s under policy π by $\pi(a|s)$ and the probability of transitioning from s to s' given a as $p(s'|s,a)$.

However, it is typical for the agent to lack knowledge of the transition and reward functions and instead have access only to a stream of transitions defined in (2.1). To estimate values in such situations where only random samples are available, the classical Monte Carlo method is a natural choice. This method forms value estimates by averaging random samples. For example, if the agent has encountered state s at time steps $\{t_i\}_{i=1}^n$ by time step t, then it can obtain an Monte Carlo estimate for $v_\pi(s)$:

$$V(s) = \frac{1}{n}\sum_{i=1}^{n} G_{t_i} \approx v_\pi(s).$$

Nevertheless, there is a problem here: By time step t, G_{t_i}s are not computable as R_{t+1}, R_{t+2}, $\cdots$ have not happened and known to the agent yet. In fact, the agent can never accurately calculate G_t for any t because the horizon is infinite. Thus, the naive Monte Carlo method will not work due to the nature of the problem.

2.2 Temporal-Difference Learning

Rather than attempting to fix the limitations of the naive Monte Carlo method, we consider an alternative approach, *temporal-difference learning*, which is straightforward, consistent with biological learning processes (Schultz et al., 1997; Montague et al., 1996), and demonstrates significantly lower variance (Watkins, 1989). Temporal-Difference (TD) learning (Sutton, 1988) relies on the concept of *TD error*. Specifically, the TD error at time step t is defined as the following:

$$\delta_t \doteq R_{t+1} + \gamma V(S_{t+1}) - V(S_t), \tag{2.3}$$

where V is the current estimated value function. In essence, the TD error captures the discrepancy between the estimated value of a given state, $V(S_t)$, and a more accurate estimate, $R_{t+1}+\gamma V(S_{t+1})$, which we also call the *target* of $V(S_t)$. One noteworthy property of the TD error is that its expected value, when computed using the true value function, is zero: for any $s \in \mathcal{S}$,

$$\begin{aligned}
\mathbb{E}_\pi[\delta_t|S_t = s] &= \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) - v_\pi(S_t)|S_t = s] \\
&= \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1})|S_t = s] - v_\pi(s) \\
&= \sum_{a\in\mathcal{A}} \pi(a|s)\left[r(s,a) + \sum_{s'\in\mathcal{S}} \gamma p(s'|s,a) v_\pi(s')\right] - v_\pi(s) \\
&= 0,
\end{aligned}$$

where in the last equation, we use the Bellman equation. This relationship between the Bellman equation and the TD error can help us get a sense of why TD learning works: when the expectation of the TD error is driven to zero, we will obtain the true value function v_π.

Using the TD error defined in (2.3), *TD(0)*, the most basic TD learning method, performs the below update to the estimated value of the current state S_t:

$$V(S_t) \leftarrow V(S_t) + \alpha\delta_t, \tag{2.4}$$

where $\alpha > 0$ is a scalar step-size parameter. Plugging in δ_t and rearranging the right-hand side of the update, we can obtain the new estimated value of the current state $(1-\alpha)V(S_t) + \alpha\left[R_{t+1} + \gamma V(S_{t+1})\right]$, a linear combination of the old estimated value and the more accurate estimate. By repeatedly updating the value estimates, we can iteratively refine the estimates and approach the true value function v_π, as proven by Sutton (1988) and Dayan (1992).

2.3 Multi-Step Temporal-Difference Learning

A helpful way to look at TD methods is to consider them as bootstrapping methods, through which we can make connections to Monte Carlo methods. When we examine the target of $V(S_t)$ in (2.3), we see that it includes the value estimate of S_{t+1}, $V(S_{t+1})$. Using estimated values in the target of updates to the same kind of estimated value is commonly referred to as *bootstrapping*. Specifically, we can say Update (2.4) corresponds to one-step bootstrapping, as it employs the estimated value after one time step. However, we can also utilize n-step bootstrapping more generally. Suppose the current estimated function is V, we define the *n-step return* at time step t as the following:

$$G_{t:t+n} \doteq R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n V(S_{t+n}), \quad \text{for } n \geq 1. \tag{2.5}$$

Subsequently, the n-step TD method performs the below update to the estimated value of the current state S_t:

$$V(S_t) \leftarrow V(S_t) + \alpha\left[G_{t:t+n} - V(S_t)\right].$$

By using n-step bootstrapping, we can interpolate between TD and Monte Carlo methods, allowing us to balance bias and variance. When n approaches infinity, the n-step return converges to the full return as defined in (2.5). At this extreme, the n-step TD method becomes a Monte Carlo method, which is unbiased but has high variance. On the other hand, when $n = 1$, the n-step return is the target $R_{t+1} + \gamma V(S_{t+1})$ in (2.3), resulting in the n-step TD method degenerating into TD(0). As n decreases towards 1, the n-step TD method introduces bias in the target but substantially reduces variance. Note that the n-step TD method requires waiting for the receipt of R_{t+n} before updating $V(S_t)$ and may require additional memory to store intermediate rewards and states. As n increases, it becomes increasingly difficult to obtain R_{t+n}, making updates infeasible for very large values of n.

TD(λ): A Smooth Bias-Variance Trade-Off

Note that the above interpolation is discrete, but in fact, we can have a continuous interpolation. This smooth interpolation is based on a different type of returns, the λ-*return*, which is defined as the following:

$$G_t^\lambda \doteq (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} G_{t:t+n}, \tag{2.6}$$

where $G_{t:t+n}$ is the n-step return defined in (2.5). It's not hard to see that the λ-return is a weighted average of all the n-step returns starting from the current time step t, and the weights sum up to one. When $\lambda = 0$, the λ-return becomes the 1-step return, corresponding to TD(0); when λ approaches one, the λ-return will get closer to the full return, matching the Monte Carlo method; when $\lambda \in (0,1)$, the λ-return interpolates between the two extremes, achieving a smooth bias-variance trade-off. The resulting algorithm makes the following update to each S_t encountered:

$$V(S_t) \leftarrow V(S_t) + \alpha \left[G_t^\lambda - V(S_t) \right]. \tag{2.7}$$

While λ-return algorithms like the one above bring a nice benefit in controlling the bias-variance trade-off, they are not directly implementable in our infinite horizon setting. The reason is the same as the naive Monte Carlo method we mentioned - the λ-return G_t^λs are not computable by any finite time step for any t. Luckily, there are both effective approximate and exact implementations for λ-return algorithms. These implementations are based on the *backward view* of learning algorithms, in which the algorithm looks back in time to make changes that should have happened in the past. Relatively, updates like Update (2.7) are deemed as the *forward view*, which makes changes to the value of the current state by looking forward to future rewards and states. To keep track of the changes that should have been made in the past, backward-view algorithms maintain an auxiliary quantity, the *eligibility trace*, $Z : \mathcal{S} \to \mathbb{R}$. In our case, the eligibility trace is maintained by the following update:

$$\begin{aligned} Z(S_t) &\leftarrow \gamma\lambda Z(S_t) + 1, \\ Z(s) &\leftarrow \gamma\lambda Z(s), \quad \text{for all } s \neq S_t, \end{aligned}$$

where $Z(s)$ is initialized to zero for all $s \in \mathcal{S}$.

The resulting algorithm, *TD(λ)* (Sutton, 1988), makes the following update:

$$V(s) \leftarrow V(s) + \alpha Z(s)\delta_t, \quad \text{for all } s \in \mathcal{S} \tag{2.8}$$

where δ_t is the TD error defined in (2.3). To see the connection between the forward view and the backward view, we can rewrite the error of the λ-return algorithm:

$$
\begin{aligned}
G_t^\lambda - V(S_t) &= (1-\lambda)\sum_{n=1}^{\infty}\lambda^{n-1}G_{t:t+n} - V(S_t) \\
&= \sum_{n=1}^{\infty}(1-\lambda)\lambda^{n-1}\left[R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1}R_{t+n} + \gamma^n V(S_{t+n})\right] - V(S_t) \\
&= \sum_{n=1}^{\infty}(1-\lambda)\lambda^{n-1}\left[\sum_{k=0}^{n-1}\gamma^k R_{t+k+1} + \gamma^n V(S_{t+n})\right] - V(S_t) \\
&= \sum_{n=1}^{\infty}(1-\lambda)\lambda^{n-1}\sum_{k=0}^{n-1}\gamma^k R_{t+k+1} + \sum_{n=1}^{\infty}(1-\lambda)\lambda^{n-1}\gamma^n V(S_{t+n}) - V(S_t) \\
&= \sum_{k=0}^{\infty}\gamma^k R_{t+k+1}\sum_{n=k+1}^{\infty}(1-\lambda)\lambda^{n-1} + \sum_{n=1}^{\infty}(1-\lambda)\lambda^{n-1}\gamma^n V(S_{t+n}) - V(S_t) \qquad (2.9) \\
&= \sum_{k=0}^{\infty}(\gamma\lambda)^k R_{t+k+1} + \sum_{n=1}^{\infty}(\gamma\lambda)^{n-1}\left[\gamma V(S_{t+n}) - V(S_{t+n-1})\right] \\
&= \sum_{n=1}^{\infty}(\gamma\lambda)^{n-1}\left[R_{t+n} + \gamma V(S_{t+n}) - V(S_{t+n-1})\right] \\
&= \sum_{n=1}^{\infty}(\gamma\lambda)^{n-1}\delta_{t+n-1},
\end{aligned}
$$

where in (2.9), we swap the order of the summations[3]. From the last equation, we can see that the error of the λ-return algorithm can be represented by the TD errors used by TD(λ) starting from time step t. From this perspective, TD(λ) actually offers a mortgage on the updates of the λ-return algorithm: at every time step t, the TD error multiplied by the eligibility trace will account for parts of previously visited states' prediction errors. It also has to be mentioned that this equivalence only holds when V will not be changed, which is not the case in TD(λ) as V will be updated every time step. Nevertheless, TD(λ) is shown to converge and has the advantage to control a bias-variance trade-off (Watkins, 1989).

Due to its simplicity, biological consistency, and computational congeniality, we will focus on methods that built upon TD(λ). Also, for the ease of exposition, we will mostly focus on the one-step case when $\lambda = 0$ in our presentation and shift to the general multi-step scenario if necessary.

[3] It holds for any sequence $a_{n,k}$ indexed by n and k that $\sum_{n=1}^{\infty}\sum_{k=1}^{n} a_{n,k} = \sum_{k=1}^{\infty}\sum_{n=k}^{\infty} a_{n,k}$. Here, we shifted the index k by one.

2.4 Online OPPE with Linear Function Approximation

In this section, we describe the problem studied in this thesis, online off-policy policy evaluation (OPPE) with linear function approximation. On top of the prediction problem mentioned in Section 2.1, we consider the below three additional constraints: linear function approximation, online updates, and off-policy data.

Linear function approximation Instead of storing a table to represent the value estimates, the agent uses linear function approximation (LFA) to approximate them. There are two considerations behind this constraint. On the one hand, by using a feature vector, the agent can reduce the memory and computation required for storing and updating the value estimates. On the other hand, using state features allows for generalization across similar states, which is important for achieving good performance in environments with a high-dimensional state space. Specifically, the states are parameterized by a feature function $\boldsymbol{\phi} : \mathcal{S} \to \mathbb{R}^d$ or equivalently a feature matrix $\mathbf{\Phi} \in \mathbb{R}^{|\mathcal{S}| \times d}$, where d is the dimension of the features. With LFA, the agent approximates the value function with $\hat{v}(s; \boldsymbol{\theta}) = \boldsymbol{\phi}(s)^\top \boldsymbol{\theta}$ or in vector form, $\hat{\mathbf{v}}_{\boldsymbol{\theta}} = \mathbf{\Phi}\boldsymbol{\theta}$, where $\boldsymbol{\theta} \in \mathbb{R}^d$ is a parameter vector.

Online updates Further, we assume that the agent is learning while interacting with the environment simultaneously. We also limit the agent to perform strict incremental updates without storing any data samples, motivated by the same concerns for memory and computation usage that led to the LFA constraint. For one thing, this new constraint allows the agent to maintain its memory usage and per-sample computation as the number of samples increases (Mahmood, 2017). For another, storing no data samples allows us to leverage the analysis tools from the stochastic approximation literature (Borkar, 2008).

Off-policy data Finally, we consider scenarios in which the agent needs to estimate the value function of a policy different from the policy it is executing. We call the policy that the agent wants to evaluate *target policy*, denoted by π, and the policy that it is following *behavior policy*, denoted by μ. As discussed in the previous chapter, learning from off-policy data is of paramount importance for intelligent agents. Here, our setting is the simplest among problems of this form, but it can reveal generalizable and valuable insights.

Putting these constraints together, the agent's task in our problem is to estimate the value function $\mathbf{v}_\pi$ using $\hat{\mathbf{v}}_{\boldsymbol{\theta}}$ with data samples $\{(S_t, A_t, R_{t+1}, S_{t+1})\}_{t=0}^{\infty}$ generated from the behavior policy μ. In addition, we make a few common assumptions to make the problem more tractable: Firstly, Assumption 2.4.1 ensures the unique existence of the corresponding stationary distributions, $d_\mu \in \Delta(\mathcal{S})$ and $d_\pi \in \Delta(\mathcal{S})$. In addition, it holds that for any $s \in \mathcal{S}$, $d_\mu(s) > 0$ and $d_\pi(s) > 0$; secondly, Assumption 2.4.2 makes sure that the *importance sampling ratio* $\rho_t \doteq \frac{\pi(A_t|S_t)}{\mu(A_t|S_t)}$ at every time step is well-defined, which will be useful later; finally, Assumption 2.4.3 ensures that the features are well-behaved, avoiding singularity in the analysis.

Assumption 2.4.1 (Ergodicity)**.** The Markov chains induced by the behavior policy μ and the target policy π are ergodic.

Assumption 2.4.2 (Coverage)**.** For any $s \in \mathcal{S}$ and $a \in \mathcal{A}$, if $\pi(a|s) > 0$, then $\mu(a|s) > 0$.

Assumption 2.4.3 (Independent Features)**.** The feature matrix $\mathbf{\Phi}$ has independent columns.

Similar to identifying v_π as $\mathbf{v}_\pi$, we also identify d_μ as $\mathbf{d}_\mu$ and d_π as $\mathbf{d}_\pi$. Moreover, we define $\mathbf{D}_\mathbf{v} \doteq diag(\mathbf{v})$ for some vector $\mathbf{v}$. Specifically, we use $\mathbf{D}_\pi$ for $\mathbf{D}_{\mathbf{d}_\pi}$ and $\mathbf{D}_\mu$ for $\mathbf{D}_{\mathbf{d}_\mu}$. We use $\|\cdot\|_\mathbf{v}$ to denote the vector norm induced by $\mathbf{D}_\mathbf{v}$ for some vector $\mathbf{v}$, i.e., $\|\mathbf{x}\|_\mathbf{v} = \sqrt{\mathbf{x}^\top \mathbf{D}_\mathbf{v} \mathbf{x}}$.

2.5 The Deadly Triad: Instability of Off-Policy TD Algorithms

In this section, we discuss a notorious issue in solving the problem of online OPPE with linear function approximation (LFA). We will first approach the three constraints of online OPPE with LFA, landing our discussion on Off-policy TD(λ), the vanilla off-policy TD learning algorithm. Then we will discuss the instability of Off-policy TD(λ), which is a manifestation of *the deadly triad*, a well-known issue in reinforcement learning. Finally, we will formally define the stability of off-policy TD algorithms.

Among the three constraints of online OPPE with LFA, we can already address the constraint of online updates with TD learning presented in Section 2.1. TD methods are naturally suitable for performing strict incremental learning because their updates could depend on data only from the current transition without storing any samples. Based on TD(λ), we will develop approaches to other constraints.

When dealing with data collected from a different policy, TD(λ) can be easily extended to Off-policy TD(λ) to handle such off-policy data. For clarity, we will refer to TD(λ) with on-policy data as On-policy TD(λ) to be distinguished from Off-policy TD(λ) in the rest of the thesis. In the tabular case, Off-policy TD(λ) makes the following update with transitions $\{(S_t, A_t, R_{t+1}, S_{t+1})\}_{t=0}^{\infty}$ generated by the behavior policy:

$$\begin{aligned}
Z(S_t) &\leftarrow \rho_t(\gamma\lambda Z(S_t) + 1), \\
Z(s) &\leftarrow \rho_t\gamma\lambda Z(s), \quad \text{for all } s \neq S_t, \\
V(s) &\leftarrow V(s) + \alpha Z(s)\delta_t, \quad \text{for all } s \in \mathcal{S},
\end{aligned}$$

where $Z(s)$ is initialized to zero for all $s \in \mathcal{S}$, and $\rho_t = \frac{\pi(A_t|S_t)}{\mu(A_t|S_t)}$ is the *importance sampling ratio* at time step t. By multiplying the importance sampling ratio, Off-policy TD(λ) can treat A_t as an action selected under the target policy. Under some mild assumptions, Off-policy TD(λ) can be shown to converge to the value function of the target policy π with tabular representation (Precup, 2000).

Further, Off-policy TD(λ) can also be intuitively extended to accommodate the last constraint, linear function approximation. Under linear function approximation, Off-policy TD(λ) makes the following update:

$$\begin{aligned}\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\delta_t\mathbf{z}_t,\\ \delta_t &= R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top\boldsymbol{\theta}_t,\\ \mathbf{z}_t &= \rho_t(\gamma\lambda\mathbf{z}_{t-1} + \boldsymbol{\phi}_t), \text{with } \mathbf{z}_{-1} = \mathbf{0},\end{aligned} \tag{2.10}$$

where $\boldsymbol{\phi}_t \doteq \boldsymbol{\phi}(S_t)$ is the features of state S_t, and we overload the notation δ_t from (2.3) for the sake of consistency.

Yet, the accommodation may not be deemed successful because Off-policy TD(λ) is unstable in the linear function approximation case. By instability, we mean Off-policy TD(λ) cannot be guaranteed to converge in this case. Several counterexamples have been suggested to showcase the divergence of Off-policy TD(λ) (Baird, 1995; Tsitsiklis and Van Roy, 1996; Sutton and Barto, 2018; Manek and Kolter, 2022). In general, the problem of divergence can arise when bootstrapping, off-policy learning, and function approximation are combined. This problematic combination is often referred to as *the deadly triad* (Sutton and Barto, 2018).

In order to facilitate further discussion, we next formally define the stability of off-policy TD algorithms. Specifically, all the TD algorithms mentioned in this thesis can be analyzed using tools from stochastic approximation (SA; Borkar, 2008). In the SA literature, a *stochastic algorithm* is defined by an update of the following form:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha(\mathbf{b}_t - \mathbf{A}_t\boldsymbol{\theta}_t),$$

where $\alpha > 0$ is a scalar step-size parameter, $\{\boldsymbol{\theta}_t\}_{t=0}^\infty$ is the sequence of weight vectors generated by the algorithm, and $\{(\mathbf{A}_t, \mathbf{b}_t)\}_{t=0}^\infty$ is a sequence of random matrices and vectors that depend on the problem and the algorithm. We then define the following limits:

$$\mathbf{A} \doteq \lim_{t\to\infty} \mathbb{E}[\mathbf{A}_t] \quad \text{and} \tag{2.11}$$

$$\mathbf{b} \doteq \lim_{t\to\infty} \mathbb{E}[\mathbf{b}_t]. \tag{2.12}$$

Using $\mathbf{A}$ and $\mathbf{b}$, we can form a deterministic algorithm:

$$\bar{\boldsymbol{\theta}}_{t+1} = \bar{\boldsymbol{\theta}}_t + \alpha(\mathbf{b} - \mathbf{A}\bar{\boldsymbol{\theta}}_t), \tag{2.13}$$

which we call the *expected update* of the stochastic algorithm.

We use the definition of the stability of a stochastic algorithm from Sutton et al. (2016): A stochastic algorithm and its expected update are *stable* if the expected update converges to a unique fixed point under any initialization. It turns out that, the expected update is stable if and only if the eigenvalues of its $\mathbf{A}$ matrix all have positive real parts (Varga, 1999). Due to the key role of the $\mathbf{A}$ matrix in determining the stability of the expected update, we say the $\mathbf{A}$ is stable if it satisfies the aforementioned property. As discussed in Sutton et al. (2016), the stability of a stochastic algorithm is essential to its convergence: If a stochastic algorithm is stable, then its parameter vector may converge with probability one with a proper step-size scheduling. Besides, if the stochastic algorithm converges, it is to the fixed point of its expected update, $\bar{\boldsymbol{\theta}} = \mathbf{A}^{-1}\mathbf{b}$.

Using the above definition, we can analyze the stability of Off-policy TD(λ). Next, we first examine the simple one-step scenario ($\lambda = 0$). When $\lambda = 0$, the update of Off-policy TD(λ) (2.10) degenerates into the following update:

$$\begin{aligned}
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\rho_t\left(R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top\boldsymbol{\theta}_t\right)\boldsymbol{\phi}_t, \\
&= \boldsymbol{\theta}_t + \alpha\left(\underbrace{\left[\rho_t R_{t+1}\boldsymbol{\phi}_t\right]}_{\mathbf{b}_t} - \underbrace{\left[\rho_t\boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top\right]}_{\mathbf{A}_t}\boldsymbol{\theta}_t\right).
\end{aligned} \tag{2.14}$$

Then the $\mathbf{A}$ matrix of Off-policy TD(0) is

$$\begin{aligned}
\mathbf{A} = \lim_{t\to\infty}\mathbb{E}_\mu[\mathbf{A}_t] &= \lim_{t\to\infty}\mathbb{E}_\mu\left[\rho_t\boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top\right] \\
&= \sum_s d_\mu(s)\mathbb{E}_\mu\left[\rho_k\boldsymbol{\phi}_k(\boldsymbol{\phi}_k - \gamma\boldsymbol{\phi}_{k+1})^\top | S_k = s\right] \\
&= \sum_s d_\mu(s)\mathbb{E}_\pi\left[\boldsymbol{\phi}_k(\boldsymbol{\phi}_k - \gamma\boldsymbol{\phi}_{k+1})^\top | S_k = s\right] \\
&= \boldsymbol{\Phi}^\top\mathbf{D}_\mu(\mathbf{I} - \gamma\mathbf{P}_\pi)\boldsymbol{\Phi}.
\end{aligned}$$

As explained by Sutton et al. (2016), the $\mathbf{A}$ matrix is stable regardless of the feature matrix $\boldsymbol{\Phi}$ when $\mu = \pi$. The intuition behind the on-policy stability is that the stationary distribution d_π in $\mathbf{D}_\pi$ (remember that $\mathbf{D}_\pi = diag(\mathbf{d}_\pi)$) interacts well with $\mathbf{P}_\pi$ to ensure the positive definiteness of $\mathbf{D}_\pi(\mathbf{I} - \gamma\mathbf{P}_\pi)$, which in turn guarantees the positive definiteness of the $\mathbf{A}$ matrix. As a result, all the eigenvalues of the $\mathbf{A}$ matrix have positive real parts, making it stable. However, when $\mu \neq \pi$, these arguments break down, and the $\mathbf{A}$ matrix is no longer guaranteed to be stable. This may lead to divergence, even with a trivial feature matrix. These arguments could also be made for the general multi-step case. To fix the instability of Off-policy TD(λ), various techniques are proposed to modify the $\mathbf{A}$ matrix to ensure stability. These techniques will be discussed in later chapters.

2.6 Consistency of Off-Policy TD Algorithms

In this section, we introduce the concept of consistency of off-policy TD algorithms, a crucial element in this thesis. Consistency not only guarantees stability but also implies low approximation errors.

Not only do we want an off-policy TD algorithm to be stable, but we also want the solutions it finds to be accurate approximations of the true value function of the target policy. To measure the quality of the fixed point $\boldsymbol{\theta}$ that an algorithm converges to, we introduce a metric, namely the root-mean-square-value error, which is defined as follows:

$$\overline{\text{RMSVE}}(\boldsymbol{\theta}) \doteq \|\hat{\mathbf{v}}_{\boldsymbol{\theta}} - \mathbf{v}_\pi\|_{\mathbf{d}_\pi}.$$

By using $\overline{\text{RMSVE}}$, we can quantify the quality of the fixed point that an algorithm converges to through evaluating how well the fixed point approximates the value function of the target policy.

In the on-policy case, a positive upper bound on the $\overline{\text{RMSVE}}$ error exists for On-policy TD(λ) (Tsitsiklis and Van Roy, 1996, Lemma 6):

$$\|\hat{\mathbf{v}}_{\bar{\boldsymbol{\theta}}_{\text{On}}} - \mathbf{v}_\pi\|_{\mathbf{d}_\pi} \leq \frac{1-\lambda\gamma}{1-\gamma}\|\hat{\mathbf{v}}_{\boldsymbol{\theta}^*} - \mathbf{v}_\pi\|_{\mathbf{d}_\pi}, \tag{2.15}$$

where $\boldsymbol{\theta}^* \doteq \arg\min_{\boldsymbol{\theta}} \|\hat{\mathbf{v}}_{\boldsymbol{\theta}} - \mathbf{v}_\pi\|_{\mathbf{d}_\pi}$ represents the best approximation, and $\bar{\boldsymbol{\theta}}_{\text{On}}$ is the *on-policy fixed point*, to which On-policy TD(λ) converges to. Specifically, the on-policy fixed point is provided by $\bar{\boldsymbol{\theta}}_{\text{On}} = \mathbf{A}^{-1}\mathbf{b}$ where $\mathbf{A}$ and $\mathbf{b}$ are defined as follows:

$$\mathbf{A} = \boldsymbol{\Phi}^\top \mathbf{D}_\pi(\mathbf{I} - \lambda\gamma\mathbf{P}_\pi)^{-1}(\mathbf{I} - \gamma\mathbf{P}_\pi)\boldsymbol{\Phi} \quad \text{and} \tag{2.16}$$

$$\mathbf{b} = \boldsymbol{\Phi}^\top \mathbf{D}_\pi(\mathbf{I} - \lambda\gamma\mathbf{P}_\pi)^{-1}\mathbf{r}_\pi. \tag{2.17}$$

Similarly, the *off-policy fixed point*, which Off-policy TD(λ) converges to if it converges, can be provided by $\bar{\boldsymbol{\theta}}_{\text{Off}} = \mathbf{A}^{-1}\mathbf{b}$ where $\mathbf{A}$ and $\mathbf{b}$ are defined as follows:

$$\mathbf{A} = \boldsymbol{\Phi}^\top \mathbf{D}_\mu(\mathbf{I} - \lambda\gamma\mathbf{P}_\pi)^{-1}(\mathbf{I} - \gamma\mathbf{P}_\pi)\boldsymbol{\Phi} \quad \text{and} \tag{2.18}$$

$$\mathbf{b} = \boldsymbol{\Phi}^\top \mathbf{D}_\mu(\mathbf{I} - \lambda\gamma\mathbf{P}_\pi)^{-1}\mathbf{r}_\pi. \tag{2.19}$$

Apart from the instability issue, Off-policy TD(λ) also have no known guarantee on the boundedness of the off-policy fixed point's error. Instead, it has been shown that the off-policy fixed point could have an unbounded error in the one-step case (Kolter, 2011).

Since On-policy TD(λ) has a benign error bound, our goal is to ensure that our off-policy TD algorithm is consistent with it. We define an off-policy TD algorithm to be *consistent* if its expected update has the same fixed point as its on-policy counterpart. Specifically, for a one-step off-policy TD algorithm, consistency is achieved when its expected update shares the same fixed point as On-policy TD(0). For a multi-step off-policy TD algorithm with a hyperparameter λ, consistency is achieved when its expected update converges to the same fixed point as On-policy TD(λ). Note that consistency is a sufficient but not necessary condition for the stability of off-policy TD algorithms. For instance, ETD(λ) (Sutton et al., 2016) is a stable algorithm despite being biased and inconsistent. In contrast, a consistent off-policy TD algorithm will be stable since it has the same expected update as On-policy TD(λ), which has been proven stable (Tsitsiklis and Van Roy, 1996). Further, if a consistent off-policy TD algorithm converges, it will converge to the on-policy fixed point (Sutton et al., 2016; Hallak and Mannor, 2017).

Note that our definition of consistency is not unprecedented. Hallak and Mannor (2017) used the same notion of consistency for off-policy TD algorithms and consequently named their algorithm *Consistent Off-Policy TD* (COP-TD(λ, β)). It should also be pointed out that the consistency we defined differs from the standard definitions in statistics. Following the convention in statistics, one can define strong and weak consistencies for off-policy TD algorithms: An off-policy TD algorithm is considered *strongly consistent* if it can *converge almost surely* to the same fixed point as its on-policy counterpart; an off-policy TD algorithm is considered *weakly consistent* if it can *converge in probability* to the same fixed point as its on-policy counterpart. Here, weak consistency is weaker than and a necessary condition for strong consistency. Furthermore, the consistency we have defined is necessary for weak consistency and is even weaker than weak consistency. Thus, we also call the consistency we defined *loose consistency* to distinguish our definition from those that follow the statistics convention.

In this thesis, we will stick to our definition and *focus on developing consistent off-policy TD algorithms to address the online OPPE problem that involves linear function approximation.*

Chapter 3

Prior Efforts on Addressing the Deadly Triad

In this chapter, we summarize efforts made to address the instability and inconsistency issues of off-policy TD learning by adjusting different components of the $\mathbf{A}$ matrix in the OPPE literature. In Section 2.5, we discussed the connection between the $\mathbf{A}$ matrix of an off-policy TD algorithm and its stability. Specifically, the instability of Off-policy TD(λ) is due to its poorly conditioned $\mathbf{A}$ matrix. The literature suggests three approaches to address this issue. The first approach focuses on adapting the feature matrix $\mathbf{\Phi}$ to the given behavior and target policies, while the second and third approaches aim to adjust the update distribution $\mathbf{D}_\mu$ to ensure stability irrespective of the feature matrix. We will cover these approaches in order.

3.1 Ensuring Stability with Proper Representation

We begin with the representation learning approach, which learns a feature matrix that prevents TD from diverging. A key work in this category is by Ghosh and Bellemare (2020), which examines the approximation error, stability, and ease of estimation of various representation schemes. These schemes typically involve decomposing the state transition matrix and reward function, which are unknown to the agent. As a result, methods based on these representations are only approximate, and any analysis under ideal conditions may have limited applicability in practice. Additionally, such analyses often rely on assumptions that restrict either the reward function or the relation between the behavior and target policies.

3.2 Reweighting Updates with A Density Ratio Estimation

In this section, we will discuss methods that adjust the update distribution $\mathbf{D}_\mu$. These methods can be divided into two approaches. One approach is to calibrate the update distribution by reweighting updates with importance-sampling ratios, which we will cover in the next section. For now, we focus on the other approach that reweights updates with a learned estimation of the *density ratio*, $\frac{d_\pi(s)}{d_\mu(s)}$. By reweighting TD updates with the density ratio, the update distribution is brought back to the on-policy distribution d_π from the off-policy distribution d_μ. Subsequently, the update distribution matrix becomes $\mathbf{D}_\pi$, ensuring the stability and consistency of the resulting algorithms.

The seminal and most relevant work in this line is *Consistent Off-Policy TD* (COP-TD(λ, β)) by Hallak and Mannor (2017). In their paper, they showed that the density ratio satisfies an equation similar to the Bellman equation but in a time-reversed order. This equation is a recursive relation between the density ratios of states at different time steps. Inspired by the relationship between the Bellman equation and TD learning, COP-TD(λ, β) learns the approximation to the density ratio through a TD-like update based on the newly derived equation. However, COP-TD(λ, β) requires an extra projection step to ensure that the approximation remains a distribution. While it guarantees consistency in the tabular case, COP-TD(λ, β) is not consistent under general linear function approximation.

Following COP-TD(λ, β), several algorithms were proposed to estimate the density ratio by exploiting the time-reversed Bellman-like equation that it satisfies. These later methods rely on optimization approaches instead of the TD approach adopted by COP-TD(λ, β). They focus more on estimating a single value that summarizes the values of the policy, which motivates them to estimate a more generally defined density ratio. The general density ratio is not limited to the ratio of the stationary distributions but also includes the *discounted occupancy measure*, which is the expected discounted visitation of each state starting from the initial state distribution d_0. The Bellman-like equation is subsequently extended for the general density ratio.

Liu et al. (2018) first proposed to transform the Bellman-like equation into a min-max optimization problem. They further constrained the solution space of the inner maximization problem to allow for a closed-form solution, resulting in the reduction of the min-max problem to a minimization problem, which is then solved using gradient descent. Unfortunately, this method is restricted to the tabular case and lacks theoretical guarantees under general linear function approximation.

Recent advancements in this area, however, have been made by the GradientDICE method proposed by Zhang et al. (2020b). GradientDICE is part of the stationary DIstribution Correction Estimation (DICE) family, along with DualDICE (Nachum et al., 2019) and GenDICE (Zhang et al., 2020a). One notable feature of these methods is that they can handle datasets that are not necessarily collected by a single behavior policy. Although their application differs somewhat from our specific case, these techniques have the potential to be adapted to suit our needs. Similar to Liu et al. (2018), each of these DICE methods constructs an objective that is transformed into a min-max optimization problem. However, unlike Liu et al. (2018), they utilize primal-dual algorithms to solve the problem without incurring a high computational complexity, drawing on techniques from the optimization literature.

Despite their popularity, most DICE methods, including DualDICE and GenDICE, assume a tabular representation. On the other hand, GradientDICE stands out as it has convergence and consistency guarantees under general linear function approximation, but only in the case of discounted occupancy measures. While the method is shown to converge for estimating the density ratio of the stationary distributions, it does introduce bias due to regularization.

3.3 Full IS TD: Reweighting Updates with the Full IS-Ratio Product

To date, *Full Importance-Sampling TD* (Full IS TD(λ); Precup et al., 2001) is the only consistent off-policy TD algorithm that does not put restrictive assumptions on the feature representation. Prior to density-ratio approaches introduced in Section 3.2, Full IS TD(λ) first introduced the idea of correcting the update distribution $\mathbf{D}_\mu$ by reweighting the update to ensure stability and consistency. Instead of reweighting the TD update with an estimated density ratio, Full IS TD(λ) uses an unbiased estimate of the density ratio, which avoids the restrictions on the feature representation. This unbiased estimate called the *full IS-ratio product* is the product of all the IS ratios up to the current time step:

$$F_t = \rho_{t-1}\rho_{t-2}\cdots\rho_0,$$

or, iteratively,

$$F_t = \rho_{t-1}F_{t-1}, \text{with } F_0 = 1. \tag{3.1}$$

Then it can be proven that F_t is an unbiased estimate of the *per-step density ratio*, $\frac{\mathbb{P}_\pi(S_t=s)}{\mathbb{P}_\mu(S_t=s)}$:

$$\mathbb{E}_\mu[F_t|S_t = s] = \frac{\mathbb{P}_\pi(S_t = s)}{\mathbb{P}_\mu(S_t = s)} \text{ for any } t > 0. \tag{3.2}$$

To provide some intuition, we next show this by induction. For $k = 0$, it is obvious that

$$\mathbb{E}_\mu[F_0|S_0 = s] = \mathbb{E}_\mu[1|S_0 = s] = 1 = \frac{d_0(s)}{d_0(s)} = \frac{\mathbb{P}_\pi(S_0 = s)}{\mathbb{P}_\mu(S_0 = s)}.$$

Assume it holds for $k < t$ that $\mathbb{E}_\mu[F_k|S_k = s] = \frac{\mathbb{P}_\pi(S_k=s)}{\mathbb{P}_\mu(S_k=s)}$, then for $k = t$, we have

$$\begin{aligned}
\mathbb{E}_\mu[F_t|S_t = s] &= \mathbb{E}_\mu[\rho_{t-1}F_{t-1}|S_t = s] \\
&= \sum_{\bar{s},\bar{a}} \mathbb{P}_\mu(S_{t-1} = \bar{s}, A_{t-1} = \bar{a}|S_t = s)\frac{\pi(\bar{a}|\bar{s})}{\mu(\bar{a}|\bar{s})}\mathbb{E}_\mu[F_{t-1}|S_{t-1} = \bar{s}] \\
&= \sum_{\bar{s},\bar{a}} \frac{\mathbb{P}_\mu(S_{t-1} = \bar{s}, A_{t-1} = \bar{a}, S_t = s)}{\mathbb{P}_\mu(S_t = s)}\frac{\pi(\bar{a}|\bar{s})}{\mu(\bar{a}|\bar{s})}\frac{\mathbb{P}_\pi(S_{t-1} = \bar{s})}{\mathbb{P}_\mu(S_{t-1} = \bar{s})} \\
&= \sum_{\bar{s},\bar{a}} \frac{\mathbb{P}_\mu(S_{t-1} = \bar{s})\mu(\bar{a}|\bar{s})p(s|\bar{s},\bar{a})}{\mathbb{P}_\mu(S_t = s)}\frac{\pi(\bar{a}|\bar{s})}{\mu(\bar{a}|\bar{s})}\frac{\mathbb{P}_\pi(S_{t-1} = \bar{s})}{\mathbb{P}_\mu(S_{t-1} = \bar{s})} \\
&= \frac{\sum_{\bar{s},\bar{a}} \mathbb{P}_\pi(S_{t-1} = \bar{s})\pi(\bar{a}|\bar{s})p(s|\bar{s},\bar{a})}{\mathbb{P}_\mu(S_t = s)} \\
&= \frac{\mathbb{P}_\pi(S_t = s)}{\mathbb{P}_\mu(S_t = s)}.
\end{aligned}$$

Utilizing the full IS-ratio product to correct the update distribution, Full IS TD(λ) makes the following update:

$$\begin{aligned}
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\delta_t\mathbf{z}_t, \\
\delta_t &= R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top\boldsymbol{\theta}_t, \\
\mathbf{z}_t &= \rho_t(\gamma\lambda\mathbf{z}_{t-1} + F_t\boldsymbol{\phi}_t), \text{with } \mathbf{z}_{-1} = \mathbf{0}, \\
F_t &= \rho_{t-1}F_{t-1}, \text{with } F_0 = 1.
\end{aligned} \tag{3.3}$$

Using the unbiasedness of F_t (3.2), we can obtain the unbiasedness of Full IS TD(λ)'s $\mathbf{A}_t$ matrix and $\mathbf{b}_t$ vector. We next show this in the one-step case to give some intuition.

In the one-step case ($\lambda = 0$), the update of Full IS TD(λ) (3.3) becomes much simpler:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha\rho_tF_t\left(R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top\boldsymbol{\theta}_t\right)\boldsymbol{\phi}_t, \tag{3.4}$$

$$= \boldsymbol{\theta}_t + \alpha\left(\underbrace{\left[\rho_tF_tR_{t+1}\boldsymbol{\phi}_t\right]}_{\mathbf{b}_t} - \underbrace{\left[\rho_tF_t\boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top\right]}_{\mathbf{A}_t}\boldsymbol{\theta}_t\right). \tag{3.5}$$

We will first show that $\mathbb{E}_\mu[\mathbf{A}_t] = \mathbb{E}_\pi\left[\boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top\right]$:

$$\begin{aligned}
\mathbb{E}_\mu[\mathbf{A}_t] &= \mathbb{E}_\mu\left[\rho_t F_t \boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top\right] \\
&= \sum_s \mathbb{P}_\mu(S_t = s)\mathbb{E}_\mu\left[\rho_t F_t \boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top | S_t = s\right] \\
&= \sum_s \mathbb{P}_\mu(S_t = s)\mathbb{E}_\mu\left[F_t | S_t = s\right]\mathbb{E}_\mu\left[\rho_t \boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top | S_t = s\right] \qquad (3.6) \\
&= \sum_s \mathbb{P}_\mu(S_t = s)\mathbb{E}_\mu\left[F_t | S_t = s\right]\mathbb{E}_\pi\left[\boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top | S_t = s\right] \\
&= \sum_s \mathbb{P}_\mu(S_t = s)\frac{\mathbb{P}_\pi(S_t = s)}{\mathbb{P}_\mu(S_t = s)}\mathbb{E}_\pi\left[\boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top | S_t = s\right] \qquad \text{(using (3.2))} \\
&= \sum_s \mathbb{P}_\pi(S_t = s)\mathbb{E}_\pi\left[\boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top | S_t = s\right] \\
&= \mathbb{E}_\pi\left[\boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top\right],
\end{aligned}$$

where (3.6) is due to the independence between F_t and $\rho_t\boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top$ given S_t. Likewise, it can be shown that $\mathbb{E}_\mu[\mathbf{b}_t] = \mathbb{E}_\pi[R_{t+1}\boldsymbol{\phi}_t]$. Replacing μ with π and ρ_t with 1 in (2.14), we can see that $\boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top$ and $R_{t+1}\boldsymbol{\phi}_t$ are On-policy TD(0)'s $\mathbf{A}_t$ matrix and $\mathbf{b}_t$ vector. Since the data comes from policy π in On-policy TD(0), the expectations of its $\mathbf{A}_t$ matrix and $\mathbf{b}_t$ vector are also $\mathbb{E}_\pi[\boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top]$ and $\mathbb{E}_\pi[R_{t+1}\boldsymbol{\phi}_t]$. Thus, Full IS TD(0)'s $\mathbf{A}_t$ matrix and $\mathbf{b}_t$ vector are unbiased.

It should be noted that the unbiasedness of an off-policy TD algorithm's $\mathbf{A}_t$ matrix and $\mathbf{b}_t$ vector implies its consistency since its $\mathbf{A}$ matrix and $\mathbf{b}$ vector are the limits of the expectations of its $\mathbf{A}_t$ matrix and $\mathbf{b}_t$ vector, respectively. Therefore, Full IS TD(0) is consistent.

Similarly, it can be shown for the general multi-step case that Full IS TD(λ) is consistent. The fixed point of Full IS TD(λ)'s expected update is thus the same as the on-policy fixed point: $\bar{\boldsymbol{\theta}}_{\text{On}} = \mathbf{A}^{-1}\mathbf{b}$ where $\mathbf{A}$ and $\mathbf{b}$ are defined as follows:

$$\mathbf{A} = \boldsymbol{\Phi}^\top\mathbf{D}_\pi(\mathbf{I} - \lambda\gamma\mathbf{P}_\pi)^{-1}(\mathbf{I} - \gamma\mathbf{P}_\pi)\boldsymbol{\Phi} \quad \text{and} \qquad (3.7)$$

$$\mathbf{b} = \boldsymbol{\Phi}^\top\mathbf{D}_\pi(\mathbf{I} - \lambda\gamma\mathbf{P}_\pi)^{-1}\mathbf{r}_\pi. \qquad (3.8)$$

Although the full IS-ratio product has the nice property of unbiasedness, its variance grows unbearably fast as t increases. As a result, Full IS TD(λ) is scarcely practical due to high variance (Sutton et al., 2016). Our experiment results in Chapter 7 also show that Full IS TD(λ) can barely learn.

3.4 Emphatic TD: Reweigthing Updates with the Emphasis

In order to address the high variance of Full IS TD(λ), Sutton et al. (2016) proposed to use an alternative weighting called the *emphasis* to reweight the TD update. The core of the emphasis is the *followon trace*, which is defined as follows:

$$F_t = \gamma\rho_{t-1}F_{t-1} + 1, \text{with } F_0 = 1. \tag{3.9}$$

If we expand the followon trace, we can see that it is a geometrically weighted sum of IS-ratio products accumulated from different time steps:

$$\begin{aligned}
F_t &= \gamma\rho_{t-1}F_{t-1} + 1 \\
&= \gamma\rho_{t-1}(\gamma\rho_{t-2}F_{t-2} + 1) + 1 \\
&= \gamma^2\rho_{t-1}\rho_{t-2}F_{t-2} + \gamma\rho_{t-1} + 1 \\
&= \gamma^2\rho_{t-1}\rho_{t-2}(\gamma\rho_{t-3}F_{t-3} + 1) + \gamma\rho_{t-1} + 1 \\
&= \gamma^3\rho_{t-1}\rho_{t-2}\rho_{t-3}F_{t-3} + \gamma^2\rho_{t-1}\rho_{t-2} + 1 \\
&\ \ \dots \\
&= \gamma^t\Pi_{k=1}^t\rho_{k-1}F_0 + \gamma^{t-1}\Pi_{k=2}^t\rho_{k-1} + \dots + \gamma^2\rho_{t-1}\rho_{t-2} + \gamma\rho_{t-1} + 1 \\
&= \gamma^t\Pi_{k=1}^t\rho_{k-1} + \gamma^{t-1}\Pi_{k=2}^t\rho_{k-1} + \dots + \gamma^2\rho_{t-1}\rho_{t-2} + \gamma\rho_{t-1} + 1.
\end{aligned}$$

By utilizing *incomplete IS-ratio products* ($\Pi_{k=n}^t\rho_{k-1}$ for $1 < n \leq t+1$), the followon trace F_t has a much lower variance than the full IS-ratio product (3.1) but also introduces bias. Thus, the algorithm that uses this trace may not be stable in general. To ensure the stability of the algorithm in the multi-step case, an extra trick is applied on top of the followon trace to form the emphasis:

$$M_t = (1-\lambda)F_t + \lambda. \tag{3.10}$$

The resulting algorithm called *Emphatic TD* (ETD(λ)) makes the following update[1]:

$$\begin{aligned}
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\delta_t\mathbf{z}_t, \\
\delta_t &= R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top\boldsymbol{\theta}_t, \\
\mathbf{z}_t &= \rho_t(\gamma\lambda\mathbf{z}_{t-1} + M_t\boldsymbol{\phi}_t), \text{with } \mathbf{z}_{-1} = \mathbf{0}, \\
M_t &= (1-\lambda)F_t + \lambda, \\
F_t &= \gamma\rho_{t-1}F_{t-1} + 1, \text{with } F_0 = 1.
\end{aligned} \tag{3.11}$$

[1] For simplicity, we have not included general state-dependent interest, discounting, and bootstrapping functions of ETD(λ) (Sutton et al., 2016). However, algorithms developed in this thesis could also be extended to those cases.

As mentioned above, ETD(λ) is biased and not consistent due to the introduction of incomplete IS-ratio products. The fixed point it converges is $\bar{\boldsymbol{\theta}}_{\text{ETD}} = \mathbf{A}^{-1}\mathbf{b}$, where $\mathbf{A}$ and $\mathbf{b}$ are defined as follows (Sutton et al., 2016):

$$\mathbf{A} = \boldsymbol{\Phi}^\top \mathbf{D}_\mathbf{m}(\mathbf{I} - \lambda\gamma\mathbf{P}_\pi)^{-1}(\mathbf{I} - \gamma\mathbf{P}_\pi)\boldsymbol{\Phi} \quad \text{and} \tag{3.12}$$

$$\mathbf{b} = \boldsymbol{\Phi}^\top \mathbf{D}_\mathbf{m}(\mathbf{I} - \lambda\gamma\mathbf{P}_\pi)^{-1}\mathbf{r}_\pi. \tag{3.13}$$

Here, $\mathbf{D}_\mathbf{m}$ is defined as $diag(\mathbf{m})$, where $\mathbf{m} = [m(s_1), \cdots, m(s_{|\mathcal{S}|})]^\top \in \mathbb{R}^{|\mathcal{S}|}$, and $m(s) \in \mathbb{R}$ is defined as follows:

$$m(s) \doteq d_\mu(s) \lim_{t\to\infty} \mathbb{E}_\mu[M_t|S_t = s], \text{ for any } s \in \mathcal{S}.$$

As it is shown in Sutton et al. (2016), $\mathbf{m}$ is not equal to $\mathbf{d}_\pi$ in general. Specifically, it can be shown that

$$m(s) = (1-\lambda)f(s) + \lambda d_\mu(s), \tag{3.14}$$

where $f(s)$ is defined as follows:

$$f(s) = d_\mu(s) \lim_{t\to\infty} \mathbb{E}_\mu[F_t|S_t = s], \text{ for any } s \in \mathcal{S}.$$

Further, in vector form, this f can be represented by the following:

$$\mathbf{f} = \left(\mathbf{I} - \gamma\mathbf{P}_\pi^\top\right)^{-1}\mathbf{d}_\mu. \tag{3.15}$$

From (3.14) and (3.15), we can obtain the intuition that ETD(λ) changes the update distribution from $\mathbf{d}_\mu$ to a distribution that is related to both the behavior policy μ and the target policy π. Though it can be shown that this distribution decided by $\mathbf{m}$ can ensure stability, it is biased from the on-policy distribution, resulting in ETD(λ)'s inconsistency.

Chapter 4

Average Emphatic TD

In this chapter, we introduce the *average followon trace* and the resulting one-step TD algorithm, *one-step Average Emphatic TD* (AETD(0)). AETD(0) is a new consistent off-policy TD learning algorithm that utilizes incomplete IS-ratio products to achieve a fading bias and avoid high variance. Firstly, in Section 4.1, we talk about the derivation and properties of the average followon trace. Then, in Section 4.2, we provide the stability and consistency guarantees of AETD(0).

4.1 Average Followon Trace

As discussed in the previous chapter, Full IS TD(λ) is the only consistent method, but its high variance makes it impractical. In contrast, ETD(λ) effectively remedies the variance issue but is biased and deviates from our objective of finding the on-policy fixed point. Therefore, the question arises: *Can we fill the gap and strike a balance between Full IS TD(λ) and ETD(λ)?* Specifically, can we find a method that is consistent and has a lower variance than Full IS TD(λ)? Fortunately, the answer is yes. Inspired by the idea of reducing variance using incomplete IS-ratio products, we propose to use the average emphatic weighting below:

$$F_t = \frac{t}{t+1}\rho_{t-1}F_{t-1} + \frac{1}{t+1}, \text{with } F_0 = 1, \tag{4.1}$$

which we term the *average followon trace.* If we expand the average followon trace, we can see that it is a uniform average of the IS-ratio products. The motivation for using an average weighting instead of the geometrically decayed weighting is to gradually reduce bias by de-emphasizing the new IS-ratio product at each time step. In fact, the bias will vanish in the limit. We next present the properties of the average followon trace F_t and then discuss the resulting one-step TD algorithm in the next section.

Lemma 4.1.1. *Under Assumption 2.4.1 and 2.4.2, if* $\lim_{t\to\infty}\mathbb{E}_\mu[F_t|S_t=s]$ *exists for all* $s\in\mathcal{S}$, *then*

$$\lim_{t\to\infty}\mathbb{E}_\mu[F_t|S_t=s]=\frac{d_\pi(s)}{d_\mu(s)}$$

holds for any $s\in\mathcal{S}$.

Proof. Let $\mathbf{f}=[f(s_1),\cdots,f(s_{|\mathcal{S}|})]^\top\in\mathbb{R}^{|\mathcal{S}|}$, and $f(s)\in\mathbb{R}$ is defined as follows:

$$f(s)\doteq d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu[F_t|S_t=s],\text{ for any }s\in\mathcal{S},\tag{4.2}$$

which exists under our assumptions. Then we have

$$\begin{aligned}
f(s)&=d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu[F_t|S_t=s]\\
&=d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu\left[\frac{t}{t+1}\rho_{t-1}F_{t-1}+\frac{1}{t+1}|S_t=s\right]\\
&=d_\mu(s)\left(\lim_{t\to\infty}\frac{t}{t+1}\mathbb{E}_\mu[\rho_{t-1}F_{t-1}|S_t=s]+\lim_{t\to\infty}\frac{1}{t+1}\right)\qquad(4.3)\\
&=d_\mu(s)\lim_{t\to\infty}\frac{t}{t+1}\lim_{t\to\infty}\mathbb{E}_\mu[\rho_{t-1}F_{t-1}|S_t=s]\qquad(4.4)\\
&=d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu[\rho_{t-1}F_{t-1}|S_t=s]\\
&=d_\mu(s)\lim_{t\to\infty}\sum_{\bar{s},\bar{a}}\mathbb{P}_\mu(S_{t-1}=\bar{s},A_{t-1}=\bar{a}|S_t=s)\frac{\pi(\bar{a}|\bar{s})}{\mu(\bar{a}|\bar{s})}\mathbb{E}_\mu[F_{t-1}|S_{t-1}=\bar{s}]\\
&=d_\mu(s)\lim_{t\to\infty}\sum_{\bar{s},\bar{a}}\frac{\mathbb{P}_\mu(S_{t-1}=\bar{s},A_{t-1}=\bar{a},S_t=s)}{\mathbb{P}_\mu(S_t=s)}\frac{\pi(\bar{a}|\bar{s})}{\mu(\bar{a}|\bar{s})}\mathbb{E}_\mu[F_{t-1}|S_{t-1}=\bar{s}]\\
&=d_\mu(s)\sum_{\bar{s},\bar{a}}\frac{d_\mu(\bar{s})\mu(\bar{a}|\bar{s})p(s|\bar{s},\bar{a})}{d_\mu(s)}\frac{\pi(\bar{a}|\bar{s})}{\mu(\bar{a}|\bar{s})}\lim_{t\to\infty}\mathbb{E}_\mu[F_{t-1}|S_{t-1}=\bar{s}]\\
&=\sum_{\bar{s},\bar{a}}\pi(\bar{a}|\bar{s})p(s|\bar{s},\bar{a})d_\mu(\bar{s})\lim_{t\to\infty}\mathbb{E}_\mu[F_{t-1}|S_{t-1}=\bar{s}]\\
&=\sum_{\bar{s}}[\mathbf{P}_\pi]_{\bar{s}s}f(\bar{s}),
\end{aligned}$$

where in (4.3) and (4.4), we use the assumption that $\lim_{t\to\infty}\mathbb{E}_\mu[F_t|S_t=s]$ exists for any $s\in\mathcal{S}$ and the facts that $\lim_{t\to\infty}\frac{t}{t+1}=1$ and $\lim_{t\to\infty}\frac{1}{t+1}=0$. From the last equation, we have $\mathbf{f}^\top=\mathbf{f}^\top\mathbf{P}_\pi$ in vector form. Since the expectations of importance sampling ratios are one and $F_0=1$, by induction, the expectation of F_t will remain one for any $t\in\mathbb{N}$. Then we have:

$$\begin{aligned}
\mathbf{1}^\top\mathbf{f}=\sum_s f(s)&=\sum_{s\in\mathcal{S}}d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu[F_t|S_t=s]\\
&=\lim_{t\to\infty}\mathbb{E}_\mu[F_t|S_t]=1.
\end{aligned}$$

By Assumption 2.4.1, the existence of the target policy's stationary distribution is unique. From $\mathbf{f}^\top = \mathbf{f}^\top \mathbf{P}_\pi$ and $\mathbf{1}^\top \mathbf{f} = 1$, we can infer that $\mathbf{f} = \mathbf{d}_\pi$, that is,

$$d_\mu(s) \lim_{t\to\infty} \mathbb{E}_\mu[F_t|S_t = s] = d_\pi(s). \tag{4.5}$$

Since it holds that $d_\mu(s) > 0$ for any $s \in \mathcal{S}$ by Assumption 2.4.1, we can divide both sides of (4.5) by $d_\mu(s)$ and conclude the proof. □

From Lemma 4.1.1, we can see that the limit of the average followon trace's expectation is the density ratio. Different from the trace F_t of Full IS TD(λ), the expectation of the average followon trace at time step t, $\mathbb{E}_\mu[F_t|S_t = s]$, will not be equal to $\frac{\mathbb{P}_\pi(S_t=s)}{\mathbb{P}_\mu(S_t=s)}$ in general. However, the distance between the two quantities will diminish to zero as t goes to infinity. Thus, we say that the average followon trace has a fading bias or a transient bias. Next, we present a simple TD algorithm that uses this trace and its property in the next section.

4.2 One-Step Average Emphatic TD: A Novel Consistent Off-Policy Algorithm

We refer to the one-step TD algorithm that utilizes the average followon trace as *one-step Average Emphatic TD* (Average Emphatic TD(0) or AETD(0)), which makes the following update:

$$\begin{aligned} \boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\rho_t F_t \delta_t \boldsymbol{\phi}_t, \\ \delta_t &= R_{t+1} + \gamma \boldsymbol{\phi}_{t+1}^\top \boldsymbol{\theta}_t - \phi_t^\top \theta_t, \\ F_t &= \frac{t}{t+1}\rho_{t-1}F_{t-1} + \frac{1}{t+1}, \text{with } F_0 = 1. \end{aligned} \tag{4.6}$$

By virtue of the relationship between the average followon trace and the density ratio (see Lemma 4.1.1), AETD(0) is a consistent off-policy TD algorithm that we desire. We present its stability and consistency in Theorem 4.2.1.

Theorem 4.2.1 (Stability and Consistency of AETD(0))**.** *Under Assumption 2.4.3 and the assumptions of Lemma 4.1.1, the expected update of AETD(0) is the same as On-policy TD(0). Accordingly, AETD(0) is stable and consistent.*

Proof. We start from the update of AETD(0). Specifically, we can rewrite its update (4.6) as follows:

$$\begin{aligned}\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\rho_t F_t \delta_t \boldsymbol{\phi}_t \\ &= \boldsymbol{\theta}_t + \alpha\rho_t F_t \left(R_{t+1} + \gamma\phi_{t+1}^\top \theta_t - \phi_t^\top \theta_t\right)\boldsymbol{\phi}_t \\ &= \boldsymbol{\theta}_t + \alpha\left(\underbrace{\left[\rho_t F_t R_{t+1}\boldsymbol{\phi}_t\right]}_{\mathbf{b}_t} - \underbrace{\left[\rho_t F_t \boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top\right]}_{\mathbf{A}_t}\boldsymbol{\theta}_t\right).\end{aligned} \tag{4.7}$$

Defining $\mathbf{A} \doteq \lim_{t\to\infty}\mathbb{E}_\mu[\mathbf{A}_t]$ and $\mathbf{b} \doteq \lim_{t\to\infty}\mathbb{E}_\mu[\mathbf{b}_t]$, we analyze the expected update of AETD(0):

$$\bar{\boldsymbol{\theta}}_{t+1} = \bar{\boldsymbol{\theta}}_t + \alpha(\mathbf{b} - \mathbf{A}\bar{\boldsymbol{\theta}}_t). \tag{4.8}$$

We first analyze the $\mathbf{A}$ matrix. Let $\mathbf{P}_\pi \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}$ denote the on-policy transition matrix with $[\mathbf{P}_\pi]_{ij} \doteq \sum_{a\in\mathcal{A}}\pi(a|i)p(j|i,a)$. Similar to obtain ETD(0)'s $\mathbf{A}$ matrix in Section 4 of Sutton et al. (2016), we have

$$\begin{aligned}\mathbf{A} = \lim_{t\to\infty}\mathbb{E}_\mu[\mathbf{A}_t] &= \lim_{t\to\infty}\mathbb{E}_\mu\left[\rho_t F_t \boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top | S_t = s\right] \\ &= \sum_s d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu\left[F_t\rho_t\boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top | S_t = s\right] \\ &= \sum_s d_\mu(s)\lim_{t\to\infty}\mathbb{E}_\mu\left[F_t|S_t = s\right]\mathbb{E}_\mu\left[\rho_k\boldsymbol{\phi}_k(\boldsymbol{\phi}_k - \gamma\boldsymbol{\phi}_{k+1})^\top | S_k = s\right] \qquad (4.9) \\ &= \sum_s f(s)\mathbb{E}_\mu\left[\rho_k\boldsymbol{\phi}_k(\boldsymbol{\phi}_k - \gamma\boldsymbol{\phi}_{k+1})^\top | S_k = s\right] \qquad \text{(Definition of } f(s) \text{ in (4.2))} \\ &= \sum_s f(s)\mathbb{E}_\pi\left[\boldsymbol{\phi}_k(\boldsymbol{\phi}_k - \gamma\boldsymbol{\phi}_{k+1})^\top | S_k = s\right] \\ &= \sum_s f(s)\boldsymbol{\phi}(s)\left(\boldsymbol{\phi}(s) - \gamma\sum_{s'}[\mathbf{P}_\pi]_{ss'}\boldsymbol{\phi}(s')\right)^\top \\ &= \mathbf{\Phi}^\top\mathbf{D_f}(\mathbf{I} - \gamma\mathbf{P}_\pi)\mathbf{\Phi},\end{aligned}$$

where in (4.9), we use the independence between F_t and $\rho_t\boldsymbol{\phi}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top$ given S_t. From the proof of Lemma 4.1.1, we know that $\mathbf{f}$ equals $\mathbf{d}_\pi$. Plugging $\mathbf{f} = \mathbf{d}_\pi$ back to the $\mathbf{A}$ matrix, we have

$$\mathbf{A} = \mathbf{\Phi}^\top\mathbf{D}_\pi(\mathbf{I} - \gamma\mathbf{P}_\pi)\mathbf{\Phi},$$

which is exactly the $\mathbf{A}$ matrix of On-policy TD(0) and known to be stable (Tsitsiklis and Van Roy, 1996). Thus, AETD(0) and its expected update are also stable by our definition.

Similarly, we can infer that

$$\mathbf{b} = \lim_{t\to\infty} \mathbb{E}_\mu[\mathbf{b}_t] = \mathbf{\Phi}^\top \mathbf{D}_\pi \mathbf{r}_\pi,$$

where $\mathbf{r}_\pi = [r_\pi(s_1), \cdots, r_\pi(s_{|\mathcal{S}|})]^\top \in \mathbb{R}^S$, and $r_\pi(s) \in \mathbb{R}$ is defined as $r_\pi(s) = \sum_{a\in\mathcal{A}} \pi(a|s)r(s,a)$. Note that this $\mathbf{b}$ vector is also the same as On-policy TD(0). Since AETD(0) shares the same fixed point as On-policy TD(0), namely $\bar{\boldsymbol{\theta}} = \mathbf{A}^{-1}\mathbf{b}$, it satisfies the definition of consistency. □

In fact, the idea of using a uniformly weighted sum of IS-ratio products to reweight the TD update is not entirely new. Hallak et al. (2016) generalized ETD(λ) by introducing a tunable decay parameter, β, to control the followon trace's (3.9) decay rate. The resulting algorithm, ETD(λ, β), uses the below followon trace:

$$F_t = \beta\rho_{t-1}F_{t-1} + 1, \text{with } F_0 = 1. \tag{4.10}$$

When $\beta < 1$, ETD(λ, β) is inconsistent and even unstable: It may diverge when $\beta < \beta_0$, where $0 \leq \beta_0 \leq \gamma$ is a condition number that depends on λ and the behavior and target policies of the problem instance being considered. When $\beta = 1$, ETD(λ, β)'s followon trace will equally weight each IS-ratio product with weight 1. However, in their case, equally weighting the products will be problematic because the expectation of the followon trace (4.10) will diverge to infinity in the limit. Therefore, our approach is the first to effectively implement this idea in a stable manner.

4.3 Summary

In this chapter, we introduced the average followon trace and the resulting one-step TD algorithm, one-step Average Emphatic TD (AETD(0)). The idea of the average followon trace comes from the intention to combine the strength of the full IS-ratio product and the followon trace. The former is an unbiased but impractical estimator of the per-step density ratio $\frac{\mathbb{P}_\pi(S_t=s)}{\mathbb{P}_\mu(S_t=s)}$, which converges to the density ratio $\frac{d_\pi(s)}{d_\mu(s)}$. The latter is more feasible but introduces persistent bias. Combining the strength of both, we showed that the average followon trace's expectation also converges to the density ratio while having a much lower variance than the full IS-ratio product, which will be demonstrated in Chapter 7. Finally, we proved that AETD(0) is stable and consistent thanks to the use of the average followon trace.

Chapter 5

General Emphatic TD

In this chapter, we introduce a generalization of the average followon trace, which we call the *general followon trace*, and the resulting algorithm, *General Emphatic TD* (GETD(λ, β, ν)). GETD(λ, β, ν) is a general off-policy TD learning algorithm that is obtained by unifying AETD(0) and other algorithms and extending to the multi-step scenario. We first talk about the derivation of the general followon trace and how the resulting algorithm unifies some existing approaches in Section 5.1. Then we extend these results to the multi-step bootstrapping case in Section 5.2.

5.1 One-Step General Emphatic TD: A Smooth Bias-Variance Trade-Off

In this section, our goal is to achieve a smooth bias-variance trade-off by unifying AETD(0) with existing methods. This approach of combining different methods to achieve a better trade-off is a recurring theme in reinforcement learning. For instance, the multi-step method TD(λ) is obtained by unifying Monte Carlo and TD methods. In Section 2.1, we discussed how varying the value of λ in TD(λ) achieves a smooth bias-variance trade-off. When $\lambda = 1$, TD(λ) is similar to Monte Carlo, with low bias and high variance; when $\lambda = 0$, it becomes TD(0), with low variance and high bias. Another example is ETD(λ, β), which achieves a smooth bias-variance trade-off by unifying existing methods and introducing the controllable parameter β. When $\beta = 0$, ETD(λ, β) becomes Off-policy TD(λ), with low variance but large bias; when $\beta = \gamma$, it degenerates into ETD(λ), with low bias but large variance; when $\beta = 1$, it is related to Full IS TD(λ).

We now apply this strategy to AETD(0) and unify it with existing algorithms. Figure 5.1 shows a conceptual relationship between AETD(0) and other one-step off-policy TD methods. To interpolate between AETD(0) and existing algorithms, we aim to unify it with Off-policy TD(0)

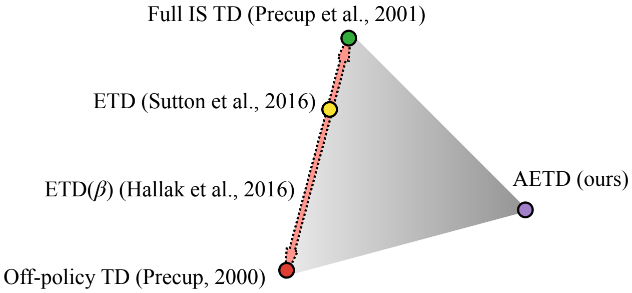

Full IS TD (Precup et al., 2001)
ETD (Sutton et al., 2016)
ETD(β) (Hallak et al., 2016)
AETD (ours)
Off-policy TD (Precup, 2000)

Figure 5.1: The relationship between various one-step off-policy TD algorithms, with the λ argument of these methods hidden. The grey area indicates potential interpolation between AETD(0) and existing algorithms. Note that the connection between ETD(0, β) (the peachy pink line) and Full IS TD(0) (the green dot) is conceptual and not exact.

and Full IS TD(0), which respectively have the least variance but the greatest bias and the least bias but the greatest variance. This unification will allow us to achieve a smooth bias-variance trade-off similar to that achieved by TD(λ) and ETD(λ, β).

We start by unifying AETD(0) and Full IS TD(0). The traces that the two methods use are $F_t = \frac{t}{t+1}\rho_{t-1}F_{t-1} + \frac{1}{t+1}$ and $F_t = \rho_{t-1}F_{t-1}$. To unify these two traces, we introduce a tunable parameter $\beta' \in [0, 1]$ to the average followon trace:

$$F_t^{(1)} = \left(1 - \frac{\beta'}{t+1}\right)\rho_{t-1}F_{t-1}^{(1)} + \frac{\beta'}{t+1}, \text{with } F_0^{(1)} = 1.$$

Then, when $\beta' = 0$, $F_t^{(1)}$ becomes $F_t^{(1)} = \rho_{t-1}F_{t-1}^{(1)}$, which corresponds to the trace of Full IS TD(0); when $\beta' = 1$, $F_t^{(1)}$ is reduced to the average followon trace.

Similarly, we can unify AETD(0) and Off-policy TD(0) with another tunable parameter ν and a new trace:

$$F_t^{(2)} = \left(1 - \frac{1}{(t+1)^\nu}\right)\rho_{t-1}F_{t-1}^{(2)} + \frac{1}{(t+1)^\nu}, \text{with } F_0^{(2)} = 1.$$

When $\nu = 0$, $F_t^{(2)}$ becomes constant 1, which corresponds to the trace of Off-policy TD(0); when $\nu = 1$, $F_t^{(2)}$ degenerates into the average followon trace.

Then we further unify the two traces, $F_t^{(1)}$ and $F_t^{(2)}$, leading us to a third trace with two parameters β' and ν:

$$F_t^{(3)} = \left(1 - \frac{\beta'}{(t+1)^\nu}\right)\rho_{t-1}F_{t-1}^{(3)} + \frac{\beta'}{(t+1)^\nu}, \text{with } F_0^{(3)} = 1.$$

It seems that we have finished the work of unification. However, we found that when $\nu = 0$, the trace becomes $F_t = (1 - \beta')\rho_{t-1}F_{t-1} + \beta'$, which is also a geometrically weighted sum of IS-ratio

Table 5.1: The coefficients of different IS-ratio products in F_t for $t > 1$.

IS-ratio Product	Off-policy TD(λ)	Scaled ETD(λ, β)	Full IS TD(λ)	AETD(λ)	ETD(λ, β)
1	1	$1-\beta$	0	$1/(t+1)$	1
ρ_{t-1}	0	$\beta(1-\beta)$	0	$1/(t+1)$	β
$\rho_{t-1}\rho_{t-2}$	0	$\beta^2(1-\beta)$	0	$1/(t+1)$	β^2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\Pi_{k=2}^t \rho_{k-1}$	0	$\beta^{t-1}(1-\beta)$	0	$1/(t+1)$	β^{t-1}
$\Pi_{k=1}^t \rho_{k-1}$	0	β^t	1	$1/(t+1)$	β^t

products as in ETD(λ, β). To ensure the resulting trace has the same decay rate as the followon trace (4.10), we replace β' with $1-\beta$ in $F_t^{(3)}$, and we refer to the resulting trace as the *general followon trace*:

$$F_t = (1-g(t))\rho_{t-1}F_{t-1} + g(t), \text{with } F_0 = 1, \tag{5.1}$$

where $g(t)$ is defined as follows:

$$g(t) \doteq (1-\beta)(t+1)^{-\nu} \tag{5.2}$$

for $\beta \in [0,1]$ and $\nu \in [0,1]$. Note that when we choose $\nu = 0$, the resulting trace will be $F_t = \beta\rho_{t-1}F_{t-1} + (1-\beta)$ with $F_0 = 1$, which we call the scaled followon trace. The resulting one-step algorithm is subsequently called *Scaled ETD(*0*,* β*)*. Although the scaled followon trace has the same decay rate as the original followon trace (3.9), it is downscaled by $1-\beta$ (see Table 5.1). This discrepancy, however, is not a qualitative difference because the constant factor $1-\beta$ can be absorbed in the step size hyperparameter. In addition, if we take the downscaling view, we will notice that the full IS-ratio product term is not downscaled to $\beta^t(1-\beta)$. However, this minor difference does not prevent Scaled ETD(0, β) from sharing the same theory and empirical performance as ETD(0, β) (with the step size scaled). Thus, Scaled ETD(0, β) can be viewed as a slight variant of ETD(0, β).

Having settled the relationship between Scaled ETD(0, β) and ETD(0, β), we are now ready to name the algorithm that unifies AETD(0), Off-policy TD(0), Full IS TD(0), and Scaled ETD(0, β). We call the resulting one-step algorithm *one-step General Emphatic TD* (GETD(0, β, ν)), which is defined as follows:

$$\begin{aligned} \boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\rho_t F_t \boldsymbol{\phi}_t, \\ \delta_t &= R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \phi_t^\top\boldsymbol{\theta}_t, \\ F_t &= (1-g(t))\rho_{t-1}F_{t-1} + g(t), \text{with } F_0 = 1, \end{aligned} \tag{5.3}$$

where $g(t)$ is defined as follows:

$$g(t) \doteq (1-\beta)(t+1)^{-\nu}$$

with $\beta \in [0,1]$ and $\nu \in [0,1]$.

5.2 Multi-Step Average Emphatic TD and General Emphatic TD

In this section, we extend the one-step TD algorithm GETD(0, β, ν) to the multi-step bootstrapping case. The objective, here, is to extend GETD(0, β, ν) in a way that leads to a unification of all the multi-step versions of algorithms that it subsumes in the one-step case. In the one step case, GETD(0, β, ν) unifies AETD(0), Off-policy TD(0), Full IS TD(0), and Scaled ETD(0, β) (including Scaled ETD(0)). Before we review the multi-step versions of these algorithms, we introduce a general description of multi-step TD algorithms with an unspecified trace M_t to simplify the presentation:

$$\begin{aligned}\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\delta_t\mathbf{z}_t,\\ \delta_t &= R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top\boldsymbol{\theta}_t,\\ \mathbf{z}_t &= \rho_t(\gamma\lambda\mathbf{z}_{t-1} + M_t\boldsymbol{\phi}_t), \text{with } \mathbf{z}_{-1} = \mathbf{0}.\end{aligned} \tag{5.4}$$

Following Sutton et al. (2016), we will refer to M_t as the *emphasis* as it emphasizes the importance of feature $\boldsymbol{\phi}_t$. For Off-policy TD(0) and Full IS TD(0), their multi-step versions have been provided in the papers that propose them (Precup, 2000; Precup et al., 2001). While these two methods were not originally designed to include the notion of emphasis, we have retroactively extended the notion to them by introducing trivial emphases. Specifically, Off-policy TD(λ) uses a constant emphasis:

$$M_t = 1,$$

while Full IS TD(λ) uses the full IS-ratio product as the emphasis:

$$M_t = F_t, \tag{5.5}$$

$$F_t = \rho_{t-1}F_{t-1}, \text{with } F_0 = 1. \tag{5.6}$$

For Scaled ETD(0, β), its multi-step version can be derived by rescaling ETD(λ, β) (Sutton et al., 2016; Hallak et al., 2016). As a result, the update of Scaled ETD(λ, β) is composed of Update

(5.4) and the following M_t:

$$M_t = (1-\lambda)F_t + \lambda(1-\beta), \tag{5.7}$$

$$F_t = \beta\rho_{t-1}F_{t-1} + (1-\beta), \text{with } F_0 = 1, \tag{5.8}$$

where F_t is the followon trace of Scaled ETD(0, β). As for AETD(0), while there are multiple potential forms for extending it to the multi-step bootstrapping case, we have not done so in the previous chapter. We have delayed the discussion of its extension until now to ensure that it can be naturally unified with other algorithms.

Naive AETD(λ)

In order to retain the consistency of AETD(0), an intuitive way to extend it to the multi-step scenario is to define the emphasis as the same as the average followon trace, that is, using (5.5) with F_t defined in (4.1). We will call this extension *Naive AETD(λ)*. The motivation of Naive AETD(λ) is that by directly using F_t as M_t as it is done in Full IS TD(λ), the consistency of AETD(0) can be retained in the multi-step case. However, it is unnatural to unify Naive AETD(λ) with other methods, as we will illustrate. Consider the structure of different algorithms' M_t in (5.5) and (5.7), we can assume the unified M_t has the following form:

$$M_t = (1 - h_1(t))F_t + h_2(t), \tag{5.9}$$

where $h_1(t)$ and $h_2(t)$ should be continuous functions and satisfy that

$$(h_1(t), h_2(t)) = \begin{cases} (\lambda, \lambda(1-\beta)) & \text{for Scaled ETD}(\lambda,\ \beta), \\ (0, 0) & \text{for Naive AETD}(\lambda) \text{ and Full IS TD}(\lambda). \end{cases} \tag{5.10}$$

Note that we have hidden the potential dependence of $h_1(t)$ and $h_2(t)$ on parameters λ, β, and ν for simplicity, but we should keep in mind the existence of the potential dependence when choosing the two functions. We first consider continuous $h_1(t)$ that works for both Scaled ETD(λ, β) and Naive AETD(λ). From Figure 5.2, we can see the values of β and ν for the two algorithms. Specifically, since $h_1(t)$ for different β in Scaled ETD(λ, β) is constant, we can ignore the dependence on β and define $h_1(t)$ as a function of only λ and ν. In this case, $h_1(t) \doteq \lambda(1-\nu)$ satisfies our purpose of unifying Scaled ETD(λ, β) and Naive AETD(λ): For Scaled ETD(λ, β) with $\nu = 0$, $h_1(t)$ becomes λ; For Naive AETD(λ) with $\nu = 1$, $h_1(t)$ degenerates into 0. This simple $h_1(t)$ seems to be an elegant solution, but unfortunately, it is incompatible with Full IS TD(λ). When $\lambda > 0$ and $\nu < 1$, $h_1(t)$ takes on a non-zero value, which conflicts with the desired zero value.

To make sure $h_1(t) = 0$ when $\beta = 1$, $\lambda > 0$, and $\nu < 1$ for Full IS TD(λ), we can introduce

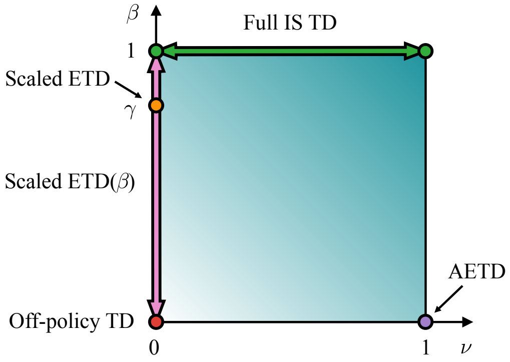


Figure 5.2: The landscape of GETD(0, β, ν). The darkness of the color at each point inside the square represents the magnitude of F_t's variance.

$(1-\beta)^\nu$ into $h_1(t)$: $h_1(t) \doteq \lambda(1-\nu)(1-\beta)^\nu$. In this way, when $\beta = 1$, $\lambda > 0$, and $0 < \nu < 1$, $h_1(t)$ will be zero, while not affecting the value of $h_1(t)$ for Scaled ETD(λ, β) and Naive AETD(λ). Yet, it still has not solved all the problems: when $\beta = 1$ and $\nu = 0$ (top-left corner of Figure 5.2), $(1-\beta)^\nu = 0^0$ is undefined. Even if we define it as 1, which is the most sensible value (Knuth, 1992), the value of $h_1(t)$ equals λ instead of 0. In fact, this is not a problem of the $h(t)$ we found, it is an unremovable discrepancy between Scaled ETD(λ, β) with $\beta = 1$ and Full IS TD(λ) when $\lambda > 0$. For the former, the emphasis is $M_t = (1-\lambda)F_t$, while for the latter, the emphasis is $M_t = F_t$, where in both cases, F_t is the full IS-ratio product. Thus, in the multi-step scenario, neither ETD(λ, β) nor Scaled ETD(λ, β) can recover Full IS TD(λ) exactly.

We can perform similar procedure to obtain $h_2(t) \doteq \lambda(1-\nu)(1-\beta)$. Finally, we can settle a smooth connection between Scaled ETD(λ, β), Naive AETD(λ), and Full IS TD(λ) in the multi-step case using the following emphasis:

$$M_t = \left(1 - \lambda(1-\nu)(1-\beta)^\nu\right) F_t + \lambda(1-\nu)(1-\beta). \tag{5.11}$$

AETD(λ) and GETD(λ)

So far, we have only examined one variant of AETD(0)'s multi-step extension. Next, we will explore a more natural form that promotes a more organic unification. Instead of using an emphasis as Full IS TD(λ) (see (5.5)), alternatively, we can create one that shares the same spirit of Scaled ETD(λ, β)'s emphasis. Specifically, if we take a closer look at the emphasis in (5.7) and the followon trace in (5.8), we can see a repeated pattern: M_t is a decaying accumulation of F_ts, while F_t is a decaying accumulation of $\rho_{t-1}F_{t-1}$s. In fact, we can also find the pattern parallel in the emphasis and followon trace of Full IS TD(λ) in (5.5) and (5.6): M_t and F_t are degenerated accumulations of F_ts and $\rho_{t-1}F_{t-1}$s both with a decaying rate of 1. Subsequently, enforcing the same pattern to AETD(0)'s multi-step extension is natural. A possible alternative emphasis in this case would be $M_t = \left(1 - \frac{1}{t+1}\right) F_t + \frac{1}{t+1}$. When $\lambda = 0$, the emphasis should reduce to the followon trace F_t. However, it does not. Thus, we introduce λ into this emphasis and obtain a new emphasis that satisfies this property:

$$M_t = \left(1 - \frac{\lambda}{t+1}\right) F_t + \frac{\lambda}{t+1}.$$

We call the resulting multi-step AETD algorithm that uses this emphasis *AETD(λ)*, which can control an additional bias-variance trade-off through λ in M_t compared to Naive AETD(λ).

Next, we unify AETD(λ) with Scaled ETD(λ, β) and Full IS TD(λ). We will use the general emphasis defined in (5.9) but with $h_1(t)$ and $h_2(t)$ that satisfy the following conditions:

$$(h_1(t), h_2(t)) = \begin{cases} (\lambda, \lambda(1-\beta)) & \text{for Scaled ETD}(\lambda,\ \beta), \\ (\frac{\lambda}{t+1}, \frac{\lambda}{t+1}) & \text{for Vanilla AETD}(\lambda), \\ (0,0) & \text{for Full IS TD}(\lambda). \end{cases} \tag{5.12}$$

Once again, we begin by considering a continuous function $h_1(t)$ that works for both AETD(λ) and Scaled ETD(λ, β). Similar to what we have done to the Naive AETD(λ) case, we can use ν to remove the appearance of $\frac{1}{t+1}$ when $\nu = 0$, resulting $h_1(t) \doteq \lambda(t+1)^{-\nu}$.

To incorporate Full IS TD(λ), we need the same ingredient from Naive AETD(λ)'s case for a similar cause. To make sure $h_1(t) = 0$ when $\beta = 1$ and $\lambda > 0$, we introduce $(1-\beta)^\nu$ into $h_1(t)$: $h_1(t) \doteq \lambda(t+1)^{-\nu}(1-\beta)^\nu$. Note that, when $\beta = 1$, $\lambda > 0$, and $\nu = 0$ (the top-left corner of Figure 5.2), we will still have $h_1(t) = \lambda \neq 0$ as in the case of Naive AETD(λ).

Similar, we can obtain $h_2(t) \doteq \lambda(t+1)^{-\nu}(1-\beta)$ that satisfies the conditions in (5.12). Thus, we can obtain a unified emphasis:

$$M_t = \left(1 - \lambda\left(\frac{1-\beta}{t+1}\right)^\nu\right) F_t + \lambda\frac{1-\beta}{(t+1)^\nu}.$$

We refer to the resulting unified multi-step algorithm as *General Emphatic TD* (GETD(λ, β, ν)). With a simplified presentation of M_t, GETD(λ, β, ν) makes the following update:

$$\begin{aligned} \boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\delta_t\mathbf{z}_t, \\ \delta_t &= R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top\boldsymbol{\theta}_t, \\ \mathbf{z}_t &= \rho_t(\gamma\lambda\mathbf{z}_{t-1} + M_t\boldsymbol{\phi}_t), \text{with } \mathbf{z}_{-1} = \mathbf{0}, \\ M_t &= (1 - \lambda h(t))\, F_t + \lambda g(t), \\ F_t &= (1 - g(t))\, \rho_{t-1}F_{t-1} + g(t), \text{with } F_0 = 1, \end{aligned} \tag{5.13}$$

where $h(t)$ and $g(t)$ are defined as follows:

$$h(t) \doteq \left(\frac{1-\beta}{t+1}\right)^{\nu} \text{ and } g(t) \doteq \frac{1-\beta}{(t+1)^{\nu}}, \tag{5.14}$$

with $\beta \in [0,1]$ and $\nu \in [0,1]$.

Similar to the one-step bootstrapping case, GETD(λ, β, ν) subsumes AETD(λ) , Off-policy TD(λ), Full IS TD(λ), Scaled ETD(λ), and Scaled ETD(λ, β). A list of the updates of all these algorithms is included in the appendix.

5.3 Summary

In this chapter, we unified AETD(0) with existing algorithms and extended the unification to the multi-step bootstrapping case. The unification is motivated by ETD(λ, β), which can control a smooth bias-variance trade-off through a hyperparameter β, the production of unification of existing methods. By introducing the general followon trace, we unified AETD(0) with Full IS TD(0) and Off-policy TD(0). We additionally found that the general followon trace also subsumes a variant of ETD(λ, β)'s followon trace. Then we extended the resulting one-step algorithm to General Emphatic TD (GETD(λ, β, ν)), the most general multi-step TD algorithm that subsumes all the aforementioned algorithms[1]. The extension was done in three steps: deriving a proper multi-step version of AETD(0), unifying the derived multi-step AETD with existing algorithms, and finally, improving the unification with a simple modification.

[1] Precisely, GETD(λ, β, ν) subsumes Scaled ETD(λ, β) instead of ETD(λ, β), but the former is a close variant of the latter with similar theoretical and empirical properties.

Chapter 6

Consistent Emphatic TD

In this chapter, we investigate a promising subclass of GETD(λ, β, ν) called *Consistent Emphatic TD* (CETD(λ, β, ν)), which offers strong theoretical guarantees. More specifically, each instance of this subclass is a consistent algorithm. In Section 6.1, we prove the stability and consistency of CETD(λ, β, ν). Then, in Section 6.2, we present three specific instances of this algorithm that are useful for practical applications and empirical evaluation.

6.1 Consistent Emphatic TD: A Practical, Consistent Off-Policy Algorithm

In this section, we focus on studying a specific part of GETD(λ, β, ν) where $\beta \in [0, 1)$ and $\nu \in (0, 1]$, as the properties of other parts are already covered by known results. When $\beta = 1$ and $\nu \in (0, 1]$, GETD(λ, β, ν) becomes Full IS TD(λ), which is consistent but impractical. On the other hand, when $\beta \in [0, 1]$ and $\nu = 0$, GETD(λ, β, ν) degenerates into Scaled ETD(λ, β). This method shares similar theoretical properties with ETD(λ, β) (Sutton et al., 2016; Hallak et al., 2016): it is biased when $\beta \in [0, 1)$ and behaves similarly to Full IS TD(λ) when $\beta = 1$.

Interestingly, the remaining subclass of GETD(λ, β, ν) with $\beta \in [0, 1)$ and $\nu \in (0, 1]$ exhibits the same stability and consistency properties as its bottom-right corner point, AETD(0) (see Figure 5.2). Therefore, we refer to this subclass as *Consistent Emphatic TD* (CETD(λ, β, ν)). CETD(λ, β, ν) employs the same update rule as GETD(λ, β, ν), but with $\beta \in [0, 1)$ and $\nu \in (0, 1]$, or $\beta = 1$

and $\nu \in [0,1]$[1]:

$$\begin{aligned}
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha \delta_t \mathbf{z}_t, \\
\delta_t &= R_{t+1} + \gamma \boldsymbol{\phi}_{t+1}^\top \boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top \boldsymbol{\theta}_t, \\
\mathbf{z}_t &= \rho_t(\gamma\lambda \mathbf{z}_{t-1} + M_t \boldsymbol{\phi}_t), \text{with } \mathbf{z}_{-1} = \mathbf{0}, \\
M_t &= (1 - \lambda h(t))\, F_t + \lambda g(t), \\
F_t &= (1 - g(t))\, \rho_{t-1} F_{t-1} + g(t), \text{with } F_0 = 1,
\end{aligned} \tag{6.1}$$

where $h(t)$ and $g(t)$ are defined as follows:

$$h(t) \doteq \left(\frac{1-\beta}{t+1}\right)^\nu \text{ and } g(t) \doteq \frac{1-\beta}{(t+1)^\nu}. \tag{6.2}$$

We next present the stability and consistency analysis of CETD(λ, β, ν). We begin by analyzing the properties of CETD(λ, β, ν)'s followon trace and emphasis. We refer to its F_t as the *consistent followon trace* and its M_t as the *consistent emphasis* in (6.1) when $\beta \in [0,1)$ and $\nu \in (0,1]$, or $\beta = 1$ and $\nu \in [0,1]$, as their properties presented in Lemma 6.1.1 and Lemma 6.1.2 guarantee the consistency of CETD(λ, β, ν) (see Theorem 6.1.3).

Lemma 6.1.1. *Under Assumption 2.4.1 and 2.4.2, for any $\beta \in [0,1)$ and $\nu \in (0,1]$, or $\beta = 1$ and $\nu \in [0,1]$, if $\lim_{t\to\infty} \mathbb{E}_\mu[F_t | S_t = s]$ exists for all $s \in \mathcal{S}$, where F_t is defined in (6.1), then*

$$\lim_{t\to\infty} \mathbb{E}_\mu[F_t | S_t = s] = \frac{d_\pi(s)}{d_\mu(s)}$$

holds for any $s \in \mathcal{S}$.

Proof. The proof is the same as that of Lemma 4.1.1, with the only difference being the substitution of $\frac{t}{t+1}$ and $\frac{1}{t+1}$ by $1 - g(t)$ and $g(t)$, respectively, in the function $f(s)$. □

Lemma 6.1.2. *Under the assumptions of Lemma 6.1.1, for any $\beta \in [0,1)$ and $\nu \in (0,1]$, or $\beta = 1$ and $\nu \in [0,1]$, it holds for any $s \in \mathcal{S}$ that*

$$\lim_{t\to\infty} \mathbb{E}_\mu[M_t | S_t = s] = \frac{d_\pi(s)}{d_\mu(s)},$$

where M_t is defined in (6.1).

[1] We also include Full IS TD(λ) in CETD(λ, β, ν), since it is also a consistent algorithm. CETD(λ, β, ν) becomes Full IS TD(λ) when $\beta = 1$ and $\nu \in [0,1]$.

Proof. We can expand M_t and use the result from Lemma 6.1.1:

$$\begin{aligned}
\lim_{t\to\infty} \mathbb{E}_\mu[M_t|S_t = s] &= \lim_{t\to\infty} \mathbb{E}_\mu[(1-\lambda h(t))F_t + \lambda g(t)|S_t = s] \\
&= \lim_{t\to\infty}(1-\lambda h(t))\mathbb{E}_\mu[F_t|S_t = s] + \lim_{t\to\infty} \lambda g(t) \\
&= \lim_{t\to\infty}(1-\lambda h(t)) \lim_{t\to\infty} \mathbb{E}_\mu[F_t|S_t = s] && (6.3) \\
&= \lim_{t\to\infty} \mathbb{E}_\mu[F_t|S_t = s] && (6.4) \\
&= \frac{d_\pi(s)}{d_\mu(s)}, && \text{(Lemma 6.1.1)}
\end{aligned}$$

where, in Equation (6.3) and (6.4), we make use of $\lim_{t\to\infty} g(t) = \lim_{t\to\infty}(1-\beta)(t+1)^{-\nu} = 0$ and $\lim_{t\to\infty} h(t) = \lim_{t\to\infty}(1-\beta)^\nu(t+1)^{-\nu} = 0$ for any $\beta \in [0, 1)$ and $\nu \in (0, 1]$, or $\beta = 1$ and $\nu \in [0, 1]$. □

Next, we prove the stability and consistency of CETD(λ, β, ν) in Theorem 6.1.3.

Theorem 6.1.3 (Stability and Consistency of CETD(λ, β, ν))**.** *Let Assumptions 2.4.1-2.4.3 hold. For any $\beta \in [0, 1)$ and $\nu \in (0, 1]$, or $\beta = 1$ and $\nu \in [0, 1]$, if $\lim_{t\to\infty} \mathbb{E}_\mu[F_t|S_t = s]$ and $\lim_{t\to\infty} \mathbb{E}_\mu[\mathbf{z}_t|S_t = s]$ exist for all $s \in \mathcal{S}$, then the expected update of CETD(λ, β, ν) is the same as On-policy TD(λ). Accordingly, CETD(λ, β, ν) is stable and consistent.*

Proof. The proof is similar in structure to the proof of Theorem 1 in the work of Sutton et al. (2016). We start from the update of CETD(λ, β, ν). Specifically, we can rewrite its update (6.1) as follows:

$$\begin{aligned}
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\delta_t\mathbf{z}_t \\
&= \boldsymbol{\theta}_t + \alpha\left(R_{t+1} + \gamma\phi_{t+1}^\top\theta_t - \phi_t^\top\theta_t\right)\mathbf{z}_t \\
&= \boldsymbol{\theta}_t + \alpha\left(\underbrace{\left[\mathbf{z}_t R_{t+1}\right]}_{\mathbf{b}_t} - \underbrace{\left[\mathbf{z}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top\right]}_{\mathbf{A}_t}\boldsymbol{\theta}_t\right).
\end{aligned} \tag{6.5}$$

Defining $\mathbf{A} \doteq \lim_{t\to\infty} \mathbb{E}_\mu[\mathbf{A}_t]$ and $\mathbf{b} \doteq \lim_{t\to\infty} \mathbb{E}_\mu[\mathbf{b}_t]$, we analyze the expected update of CETD(λ, β, ν):

$$\bar{\boldsymbol{\theta}}_{t+1} = \bar{\boldsymbol{\theta}}_t + \alpha(\mathbf{b} - \mathbf{A}\bar{\boldsymbol{\theta}}_t). \tag{6.6}$$

We first analyze the $\mathbf{A}$ matrix. Similar to obtain ETD(λ)'s $\mathbf{A}$ matrix (Sutton et al., 2016), we have

$$\begin{aligned}
\mathbf{A} = \lim_{t\to\infty} \mathbb{E}_\mu[\mathbf{A}_t] &= \lim_{t\to\infty} \mathbb{E}_\mu \left[\mathbf{z}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top | S_t = s\right] \\
&= \sum_s d_\mu(s) \lim_{t\to\infty} \mathbb{E}_\mu \left[\mathbf{z}_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top | S_t = s\right] \\
&= \sum_s d_\mu(s) \lim_{t\to\infty} \mathbb{E}_\mu \left[\rho_t(\gamma\lambda\mathbf{z}_{t-1} + M_t\boldsymbol{\phi}_t)(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top | S_t = s\right] \\
&= \sum_s d_\mu(s) \lim_{t\to\infty} \mathbb{E}_\mu \left[(\gamma\lambda\mathbf{z}_{t-1} + M_t\boldsymbol{\phi}_t) | S_t = s\right] \mathbb{E}_\mu \left[\rho_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top | S_t = s\right] \\
&\quad \text{(because, } \gamma\lambda\mathbf{z}_{t-1} + M_t\boldsymbol{\phi}_t \text{ is independent of } \rho_t(\boldsymbol{\phi}_t - \gamma\boldsymbol{\phi}_{t+1})^\top \text{ if } S_t \text{ is given)} \\
&= \sum_s \underbrace{d_\mu(s) \lim_{t\to\infty} \mathbb{E}_\mu \left[\gamma\lambda\mathbf{z}_{t-1} + M_t\boldsymbol{\phi}_t | S_t = s\right]}_{\mathbf{z}(s)\in\mathbb{R}^d} \mathbb{E}_\mu \left[\rho_k(\boldsymbol{\phi}_k - \gamma\boldsymbol{\phi}_{k+1})^\top | S_k = s\right] \\
&= \sum_s \mathbf{z}(s) \mathbb{E}_\mu \left[\rho_k(\boldsymbol{\phi}_k - \gamma\boldsymbol{\phi}_{k+1})^\top | S_k = s\right] \\
&= \sum_s \mathbf{z}(s) \mathbb{E}_\pi \left[\boldsymbol{\phi}_k - \gamma\boldsymbol{\phi}_{k+1} | S_k = s\right]^\top \\
&= \sum_s \mathbf{z}(s) \left(\boldsymbol{\phi}(s) - \gamma \sum_{s'} [\mathbf{P}_\pi]_{ss'} \boldsymbol{\phi}(s')\right)^\top \\
&= \mathbf{Z}^\top(\mathbf{I} - \gamma\mathbf{P}_\pi)\boldsymbol{\Phi},
\end{aligned}$$

where $\mathbf{Z} \doteq [\mathbf{z}(s_1), \cdots, \mathbf{z}(s_{|\mathcal{S}|})]^\top \in \mathbb{R}^{|\mathcal{S}|\times d}$, and $\mathbf{z}(s) \in \mathbb{R}^d$ is defined by

$$\begin{aligned}
\mathbf{z}(s) &\doteq d_\mu(s) \lim_{t\to\infty} \mathbb{E}_\mu \left[\gamma\lambda\mathbf{z}_{t-1} + M_t\boldsymbol{\phi}_t | S_t = s\right] \\
&= \underbrace{d_\mu(s) \lim_{t\to\infty} \mathbb{E}_\mu \left[M_t | S_t = s\right]}_{m(s)} \boldsymbol{\phi}(s) + \gamma\lambda d_\mu(s) \lim_{t\to\infty} \mathbb{E}_\mu \left[\mathbf{z}_{t-1} | S_t = s\right] \\
&= m(s)\boldsymbol{\phi}(s) + \gamma\lambda d_\mu(s) \sum_{\bar{s},\bar{a}} \lim_{t\to\infty} \mathbb{P}_\mu(S_{t-1} = \bar{s}, A_{t-1} = \bar{a} | S_t = s) \mathbb{E}_\mu \left[\mathbf{z}_{t-1} | S_{t-1} = \bar{s}, A_{t-1} = \bar{a}\right] \\
&= m(s)\boldsymbol{\phi}(s) + \gamma\lambda d_\mu(s) \sum_{\bar{s},\bar{a}} \frac{d_\mu(\bar{s})\mu(\bar{a}|\bar{s})p(s|\bar{s},\bar{a})}{d_\mu(s)} \lim_{t\to\infty} \mathbb{E}_\mu \left[\mathbf{z}_{t-1} | S_{t-1} = \bar{s}, A_{t-1} = \bar{a}\right] \\
&= m(s)\boldsymbol{\phi}(s) + \gamma\lambda \sum_{\bar{s},\bar{a}} d_\mu(\bar{s})\mu(\bar{a}|\bar{s})p(s|\bar{s},\bar{a}) \frac{\pi(\bar{a}|\bar{s})}{\mu(\bar{a}|\bar{s})} \lim_{t\to\infty} \mathbb{E}_\mu \left[\gamma\lambda\mathbf{z}_{t-2} + M_{t-1}\boldsymbol{\phi}_{t-1} | S_{t-1} = \bar{s}\right] \\
&= m(s)\boldsymbol{\phi}(s) + \gamma\lambda \sum_{\bar{s}} \left(\sum_{\bar{a}} \pi(\bar{a}|\bar{s})p(s|\bar{s},\bar{a})\right) \mathbf{z}(\bar{s}) \\
&= m(s)\boldsymbol{\phi}(s) + \gamma\lambda \sum_{\bar{s}} [\mathbf{P}_\pi]_{\bar{s}s} \mathbf{z}(\bar{s}).
\end{aligned}$$

In matrix form, we have

$$\begin{aligned}\mathbf{Z}^\top &= \mathbf{\Phi}^\top \mathbf{D_m} + \mathbf{Z}^\top(\gamma\lambda \mathbf{P}_\pi)\\ &= \mathbf{\Phi}^\top \mathbf{D_m} + \mathbf{\Phi}^\top \mathbf{D_m}(\gamma\lambda \mathbf{P}_\pi) + \mathbf{Z}^\top(\gamma\lambda \mathbf{P}_\pi)^2\\ &= \mathbf{\Phi}^\top \mathbf{D_m} + \mathbf{\Phi}^\top \mathbf{D_m}(\gamma\lambda \mathbf{P}_\pi) + \mathbf{\Phi}^\top \mathbf{D_m}(\gamma\lambda \mathbf{P}_\pi)^2 + \cdots\\ &= \mathbf{\Phi}^\top \mathbf{D_m}(\mathbf{I} - \gamma\lambda \mathbf{P}_\pi)^{-1},\end{aligned}$$

where $\mathbf{D_m} \doteq diag(\mathbf{m}) \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}$, $\mathbf{m} = [m(s_1), \cdots, m(s_{|\mathcal{S}|})]^\top \in \mathbb{R}^{|\mathcal{S}|}$, and $m(s) \in \mathbb{R}$ is defined as follows:

$$m(s) \doteq d_\mu(s) \lim_{t\to\infty} \mathbb{E}_\mu[M_t | S_t = s], \text{ for any } s \in \mathcal{S},$$

which exists due to Lemma 6.1.2. Further, from Lemma 6.1.2, we have that

$$\begin{aligned}m(s) &= d_\mu(s) \lim_{t\to\infty} \mathbb{E}_\mu[M_t | S_t = s]\\ &= d_\mu(s)\frac{d_\pi(s)}{d_\mu(s)}\\ &= d_\pi(s). \qquad (6.7)\end{aligned}$$

In vector form, we have $\mathbf{m} = \mathbf{d}_\pi$.

Plugging $\mathbf{m} = \mathbf{d}_\pi$ and $\mathbf{Z}^\top = \mathbf{\Phi}^\top \mathbf{D_m}(\mathbf{I} - \gamma\lambda \mathbf{P}_\pi)^{-1}$ back to the $\mathbf{A}$ matrix, we have

$$\mathbf{A} = \mathbf{\Phi}^\top \mathbf{D}_\pi(\mathbf{I} - \lambda\gamma \mathbf{P}_\pi)^{-1}(\mathbf{I} - \gamma \mathbf{P}_\pi)\mathbf{\Phi},$$

which is exactly the $\mathbf{A}$ matrix of On-policy TD(λ) and known to be stable (Tsitsiklis and Van Roy, 1996). Thus, CETD(λ, β, ν) and its expected update are also stable by our definition.

Similarly, we can infer that

$$\mathbf{b} = \lim_{t\to\infty} \mathbb{E}_\mu[\mathbf{b}_t] = \mathbf{\Phi}^\top \mathbf{D}_\pi(\mathbf{I} - \lambda\gamma \mathbf{P}_\pi)^{-1}\mathbf{r}_\pi,$$

where $\mathbf{r}_\pi = [r_\pi(s_1), \cdots, r_\pi(s_{|\mathcal{S}|})]^\top \in \mathbb{R}^{|\mathcal{S}|}$, and $r_\pi(s) \in \mathbb{R}$ is defined as $r_\pi(s) = \sum_{a\in\mathcal{A}} \pi(a|s) r(s,a)$. Note that this $\mathbf{b}$ is also the same as On-policy TD(λ). Thus, CETD(λ, β, ν) has the same fixed point, $\bar{\boldsymbol{\theta}} = \mathbf{A}^{-1}\mathbf{b}$, as On-policy TD($\lambda$). □

Remark 6.1.4. CETD(λ, β, ν) is stable for any $\beta \in [0, 1)$ and $\nu \in (0, 1]$, or $\beta = 1$ and $\nu \in [0, 1]$. This is significantly stronger than ETD(λ, β) (Hallak et al., 2016). In their case, ETD(λ, β) is stable only with $\beta > \beta_0$, where $0 \le \beta_0 \le \gamma$ is a condition number that depends on λ and behavior and target policies of the problem instance.

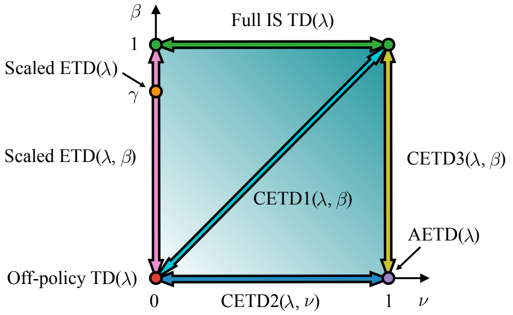


Figure 6.1: The landscape of GETD(λ, β, ν). The square excluding the left edge and its bottom endpoint represents CETD(λ, β, ν). The darkness of the color at each point inside the square represents the magnitude of F_t's variance.

Remark 6.1.5. CETD(λ, β, ν) is consistent for any $\beta \in [0, 1)$ and $\nu \in (0, 1]$, or $\beta = 1$ and $\nu \in [0, 1]$. This is, again, significantly stronger than ETD(λ, β). For any $\beta \in [0, 1)$, ETD(λ, β) has a persistent bias. In particular, the bias will increase as the value of β decrease. At the extreme end when $\beta = 0$, ETD(λ, β) becomes Off-policy TD(λ), which could have unbounded bias when $\lambda = 0$ (Kolter, 2011). On the other hand, CETD(λ, β, ν)'s bias is transient for any $\beta \in [0, 1)$ and $\nu \in (0, 1]$, or $\beta = 1$ and $\nu \in [0, 1]$.

Remark 6.1.6. AETD(λ) introduced in Section 5.2 is stable and consistent, as it is a special case of CETD(λ, β, ν) with $\beta = 0$ and $\nu = 1$.

6.2 Instances of Consistent Emphatic TD

Having settled the consistency of CETD(λ, β, ν), we now discuss the bias-variance trade-off we obtained. Figure 6.1 plots the landscape of GETD(λ, β, ν), which clearly illustrates the relationship between CETD(λ, β, ν) and other algorithms. Starting from AETD(λ), intuitively, as the value of ν decreases, the algorithm will get closer to Off-policy TD(λ) with the variance decreased but the bias increased; Meanwhile, as the value of β increases, the algorithm will approach Full IS TD(λ) with the bias decreased but the variance increased. More generally, it holds for CETD(λ, β, ν) that increasing β or ν will reduce the bias and increase the variance, and vice versa.

To better analyze the bias-variance trade-off that β and ν control, we study CETD(λ, β, ν)'s three instances, which cover a diagonal line and two edges of CETD(λ, β, ν) (see Figure 6.1).

The first instance is CETD1(λ, β), which corresponds to a diagonal line of CETD(λ, β, ν). In this diagonal line, the value of ν is always the same as the value of β. This line has the special property that it connects Off-policy TD(λ) and Full IS TD(λ). The update of CETD1(λ, β) is the same as Update (6.1) but with $h(t)$ and $g(t)$ specified as the following:

$$h(t) \doteq \left(\frac{1-\beta}{t+1}\right)^{\beta} \text{ and } g(t) \doteq \frac{1-\beta}{(t+1)^{\beta}}. \tag{6.8}$$

The second instance is CETD2(λ, ν), which corresponds to the bottom edge of CETD(λ, β, ν). In this line, β is always 0. Towards the left endpoint, CETD2(λ, ν) approaches Off-policy TD(λ); at the right endpoint, CETD2(λ, ν) becomes AETD(λ). The update of CETD2(λ, ν) is the same as Update (6.1) but with $h(t)$ and $g(t)$ specified as the following:

$$h(t) \doteq (t+1)^{-\nu} \text{ and } g(t) \doteq (t+1)^{-\nu}. \tag{6.9}$$

The third instance is CETD3(λ, β), which corresponds to the right edge of CETD(λ, β, ν). In this edge, ν is always 1. At the bottom endpoint, CETD3(λ, β) becomes AETD(λ); at the top endpoint, CETD3(λ, β) degenerates into Full IS TD(λ). The update of CETD3(λ, β) is the same as Update (6.1) but with $h(t)$ and $g(t)$ specified as the following:

$$h(t) \doteq \frac{(1-\beta)}{t+1} \text{ and } g(t) \doteq \frac{(1-\beta)}{t+1}. \tag{6.10}$$

6.3 Summary

In this chapter, we investigated Consistent Emphatic TD (CETD(λ, β, ν)), an attractive subclass of GETD(λ, β, ν) when $\beta \in [0, 1)$ and $\nu \in (0, 1]$, or $\beta = 1$ and $\nu \in [0, 1]$. We settled the stability and consistency of CETD(λ, β, ν). We also discussed the effect of β and ν in controlling the bias-variance trade-off as well as introducing its three instances for convenient empirical evaluation and practical use.

Chapter 7

Empirical Properties of Consistent Emphatic TD

In this chapter, we examine the empirical properties of CETD(λ, β, ν). First, in Section 7.1, we illustrate the benefit of CETD(λ, β, ν)'s consistency in the one-step case on a simple didactic task. We then demonstrate the practicality of CETD(λ, β, ν) in the one-step case on a task with more complex feature representation and higher variance in Section 7.2. In Section 7.3, we present the results of similar experiments in the multi-step scenario. Furthermore, we provide the step-size sensitivity analysis to the experiments mentioned above in Section 7.4. Lastly, in Section 7.5, we perform a bias-variance analysis that illustrates how the decay parameters (β and ν) control the trade-off between bias and variance.

We use constant step sizes $\alpha = 2^x$ where $x \in \{-18, -17, \cdots, -1, 0\}$ for all algorithms in all the experiments. For *tunable algorithms* with an adjustable decay parameter (ETD(λ, β), CETD1(λ, β), CETD2(λ, ν), and CETD3(λ, β)), the decay parameter (β or ν) is chosen from $\{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$. Notice that ETD($\lambda$, β), CETD1(λ, β), and CETD2(λ, ν) with $\beta = 0.0$ or $\nu = 0.0$ are the same as Off-policy TD(λ); CETD1(λ, β) and CETD3(λ, β) with $\beta = 1.0$ degenerates into Full IS TD(λ); ETD(λ, β) with $\beta = 1.0$ is an unsound method with a followon trace whose expectation will blow up to infinity in the limit. Unless otherwise specified, results are reported with the best-performing step size, with which the final error is the smallest. The final error is calculated by averaging the errors in the last 1% of the training steps. Compared to the area under the learning curve (AUC), the final error is favored because it is a better reflection of how the algorithm performs asymptotically. To evaluate the quality of the learned $\boldsymbol{\theta}$, we use the root-mean-square-value error as our metric:

$$\overline{\text{RMSVE}}(\boldsymbol{\theta}) \doteq \|\hat{\mathbf{v}}_{\boldsymbol{\theta}} - \mathbf{v}_\pi\|_{\mathbf{d}_\pi}.$$

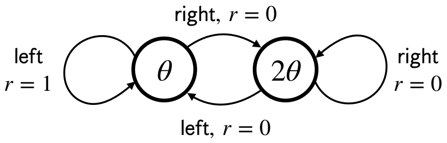


Figure 7.1: The Two-state task. The values of the two states are approximated by θ and 2θ, respectively.

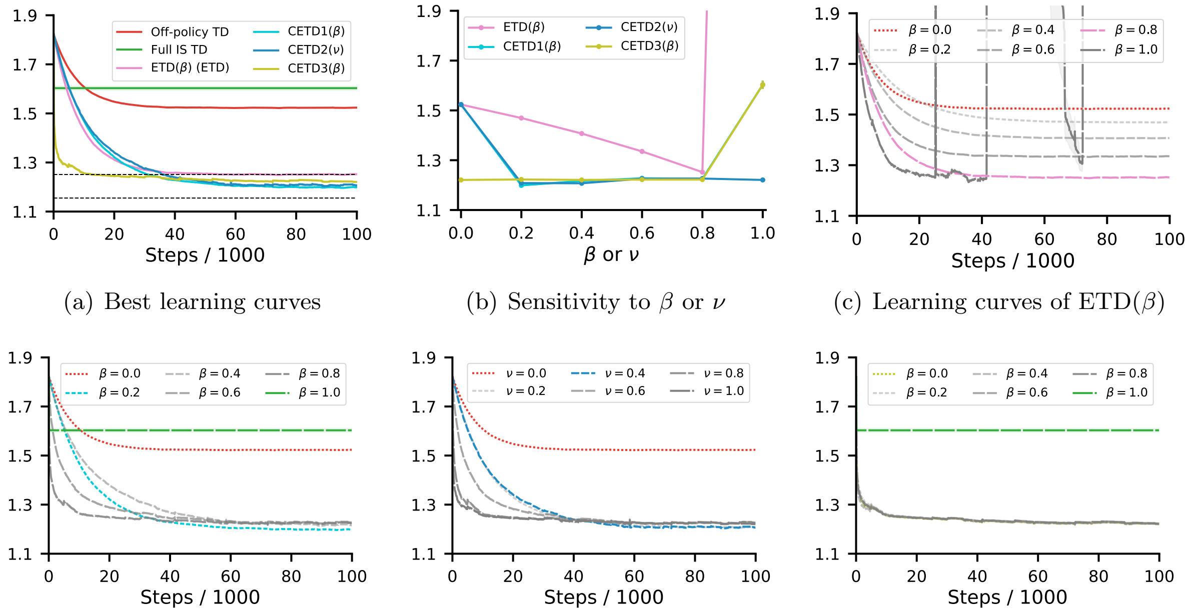
The shaded region near each presented learning curve represents the standard error over multiple runs (see the corresponding section for the number of runs). Likewise, the standard error is shown as an error bar for each point in the presented sensitivity plot. For simplicity, we drop the λ argument in all the algorithms in experiments in the one-step ($\lambda = 0$) case when there is no confusion. For example, we will use CETD(β, ν) for CETD(0, β, ν).

7.1 Consistency of Consistent Emphatic TD

Two-State Didactic Task To illustrate the benefit of CETD(β, ν)'s consistency in the one-step case, we designed a didactic task with two states (see Figure 7.1). In this task, the target policy π will go to the left state from any state with a probability of 0.6, while the probability for the behavior policy is 0.4. The discount factor γ is 0.8. The on-policy fixed point for this task is $\bar{\theta}_{\text{On}} \approx 0.8257$, which induces an error of $\overline{\text{RMSVE}}(\bar{\theta}_{\text{On}}) \approx 1.155$. The off-policy fixed point is $\bar{\theta}_{\text{Off}} \approx 0.3061$, which induces an error of $\overline{\text{RMSVE}}(\bar{\theta}_{\text{Off}}) \approx 1.523$. The fixed point of ETD (ETD(β) with $\beta = 0.8$) is $\bar{\theta}_{\text{ETD}} \approx 0.7392$, which induces an error of $\overline{\text{RMSVE}}(\bar{\theta}_{\text{ETD}}) \approx 1.251$. Thus, consistent algorithms have a theoretical advantage in this task because their fixed point is the on-policy fixed point, which has the lowest $\overline{\text{RMSVE}}$. In this task, we run each algorithm for $100,000$ steps and present the experiment results in Figure 7.2, which are averaged over 100 independent runs.

From Figure 7.2(a), we can see that all CETD instances achieve the lowest error between $\overline{\text{RMSVE}}(\bar{\theta}_{\text{ETD}})$ and $\overline{\text{RMSVE}}(\bar{\theta}_{\text{On}})$ (see the dash lines). Since the transient bias has not completely faded away within the given time steps, the error is still slightly larger than the theoretical optimal error of 1.155. Nevertheless, this is the best performance among existing algorithms and a significant improvement over the only existing consistent algorithm, Full IS TD, which does not learn due to the high variance issue. On the other hand, ETD (ETD(β) with $\beta = \gamma = 0.8$) is the second-tier algorithm in this task, achieving its theoretical optimal error of 1.251. For Off-policy TD, it also converges to its fixed point, which induces a significantly larger error of $\overline{\text{RMSVE}}(\bar{\theta}_{\text{Off}}) \approx 1.523$.

Figures 7.2(c)-7.2(f) plot the learning curves of tunable algorithms with fixed values of the decay parameter. Note that the red dotted and green dashed lines correspond to Off-policy TD and Full IS TD, respectively. From Figure 7.2(c), it is evident that ETD(β) converges to solutions with large biases for most values of β. For $\beta = 1$, its error suddenly explodes after some steps,



(a) Best learning curves

(b) Sensitivity to β or ν

(c) Learning curves of ETD(β)

(d) Learning curves of CETD1(β)

(e) Learning curves of CETD2(ν)

(f) Learning curves of CETD3(β)

Figure 7.2: Performance of different algorithms on the Two-state task. The y-axis shows $\overline{\text{RMSVE}}$. The dash lines from top to bottom in Figure (a) show $\overline{\text{RMSVE}}(\bar{\theta}_{\text{ETD}}) \approx 1.251$ and $\overline{\text{RMSVE}}(\bar{\theta}_{\text{On}}) \approx 1.155$, respectively.

demonstrating the unsoundness of ETD(β) with $\beta = 1$ in the continuing case. From Figures 7.2(d)-7.2(f), with the decay parameter in interval $[0.2, 0.8]$, CETD instances all converge smoothly to errors close to $\overline{\text{RMSVE}}(\bar{\theta}_{\text{On}})$, providing support to CETD(β, ν)'s consistency under a wide range of its hyperparameters values.

Figure 7.2(b) summarizes these results. We can conclude that all CETD instances enjoy lower errors compared to existing algorithms across a wide range of the decay hyperparameter, illustrating the benefit of CETD(β, ν)'s consistency.

7.2 Practicality of Consistent Emphatic TD

Rooms Task To further test the performance of CETD(β, ν) in more complex tasks with higher variances, we extended the episodic Rooms task proposed by Ghiassian and Sutton (2021) to the continuing setting. The Rooms task is based on the Four Rooms environment (Sutton et al., 1999), which can be partitioned into four parts that are connected by hallways (see Figure 7.3). The Four Rooms environment has 104 states, including four hallway states. The four actions in this environment will move the agent by 1 state towards the corresponding direction. If an action causes the agent to get out of the boundary, the agent will stay in the current state.

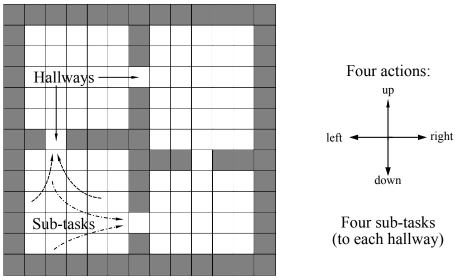


Figure 7.3: The Rooms task. Modified from Sutton et al. (1999).

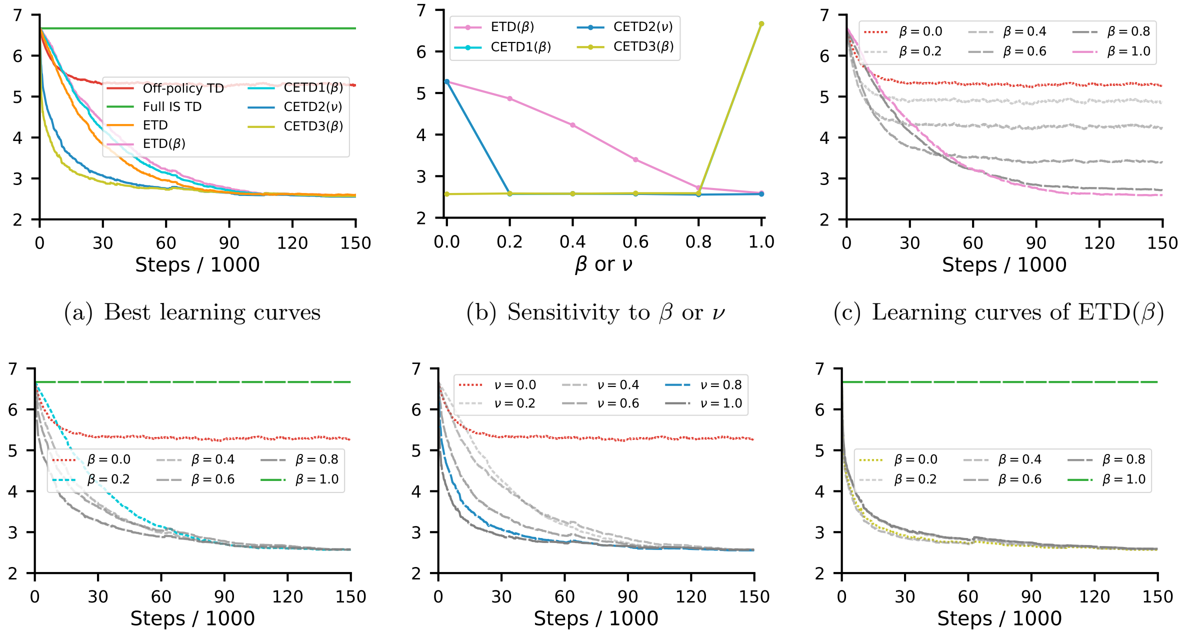
The task consists of four sub-tasks. Each sub-task will assign a reward of 1 to the agent if it arrives or stays at the corresponding hallway state. However, the agent cannot stay in a hallway state permanently as there is noise in the interactions. At each time step, there is a probability of 50% that the action of the agent will be treated as one of the other three actions with equal probability. The agent needs to learn the value functions for the four target policies while following a uniform random behavior policy. The four target policies will each try to go to a hallway state. Specifically, each target policy will choose the optimal action to the corresponding hallway state with probability $1 - \epsilon$ and choose a random action with probability ϵ, which is set to 0.1 in our experiments. The discount factor γ is 0.9.

Compared to the Two-state task, the Rooms task has a larger difference between the behavior and target policies, inducing a much higher variance. Moreover, the Rooms task has more states and complex feature representation. The coordinates of each state (x, y) are tile coded. Each coordinate ranges from 0 to 10, with the origin located at the bottom left corner. Four tilings are applied, and each of them consists of two by two tiles.

Note that it is hard to calculate the fixed points analytically in this task. Thus, we applied On-policy TD with tabular features on a trajectory following the target policy for $2,000,000$ steps for each target policy and used the final value function as the ground truth v_π. Similarly, the on-policy distributions are calculated by following each target policy for $2,000,000$ steps. We run each algorithm for $150,000$ steps and 30 runs. The error, $\overline{\text{RMSVE}}$, is averaged over four sub-tasks. To showcase the advantage of CETD(β, ν), we focus on the *Interquartile Mean*[1] (IQM) results in this task if not otherwise specified. The results are presented in Figure 7.4.

From Figure 7.4(a), we can see that ETD, ETD(β), and all CETD instances achieve similar final errors. Among them, CETD2(ν) and CETD3(β) learn the fastest. Same as in the Two-state task, Off-policy TD converges very fast to a solution with significant bias, while Full IS TD does not learn at all.

[1] The *Interquartile Mean* (IQM) is a statistic that measures the average value of the middle 50% of the data, which makes it more robust and statistically efficient than the mean or median (Agarwal et al., 2021).



(a) Best learning curves

(b) Sensitivity to β or ν

(c) Learning curves of ETD(β)

(d) Learning curves of CETD1(β)

(e) Learning curves of CETD2(ν)

(f) Learning curves of CETD3(β)

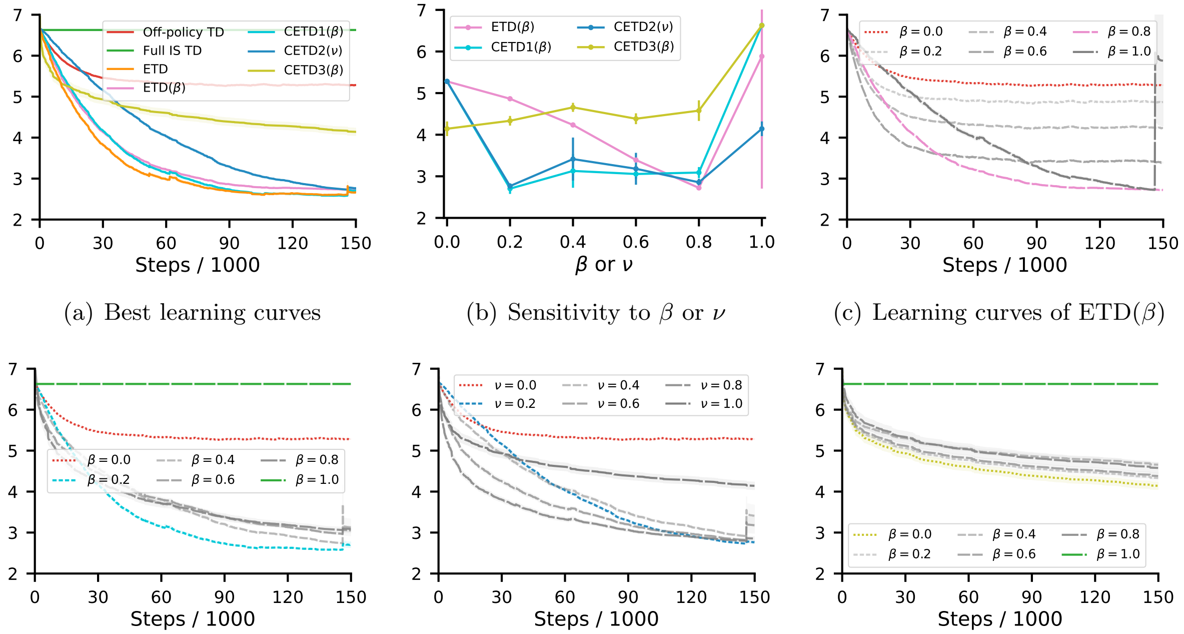
Figure 7.4: Performance of different algorithms on the Rooms task. The y-axis shows $\overline{\text{RMSVE}}$.

Figures 7.4(c)-7.4(f) plot the learning curves of tunable algorithms with fixed values of the decay parameter. From Figure 7.4(c), we can see that as the value of β increases, the bias of the solution ETD(β) found becomes smaller, and the learning also becomes slower. For CETD1(β) and CETD2(ν) (see Figures 7.4(d) and 7.4(e)), they learn faster with larger values of the decay parameter. On the other hand, CETD3(β) is not sensitive to the value of β (see Figure 7.4(f)).

Figure 7.4(b) summarizes the above results. We can see that even in the high variance setting, CETD instances are still better: They converge faster to the lowest error and are less sensitive to the decay parameter compared to ETD(β).

To provide a comprehensive performance profile of CETD algorithms, we present the mean results that averaged over all 30 runs in Figure 7.5. Compared to the more stable IQM results in Figure 7.4, the mean results are more susceptible to the influence of outlier scores.

From Figure 7.5(a), we can see that ETD, ETD(β), CETD1(β), and CETD2(ν) are the top-tier algorithms in this case. Among them, ETD, CETD1(β), and CETD2(ν) perform less stable due to the high variance of this task. More notably, CETD3(β) suffers more from the variance issue and cannot learn efficiently, but still, it is much better than Off-policy TD. For Full IS TD and Off-policy TD, their performances are not much different than the IQM results presented in Figure 7.4(a): The former cannot learn at all despite being the only existing consistent algorithm, while the latter converges to a solution with significant bias. Finally, from Figure 7.5(b), we can see that



(a) Best learning curves (b) Sensitivity to β or ν (c) Learning curves of ETD(β)

(d) Learning curves of CETD1(β) (e) Learning curves of CETD2(ν) (f) Learning curves of CETD3(β)

Figure 7.5: Mean results averaged over all 30 runs on the Rooms task. The y-axis shows $\overline{\text{RMSVE}}$.

CETD1(β) and CETD2(ν) are still less sensitive to the decaying parameter compared to ETD(β).

7.3 Results for Multi-Step Bootstrapping

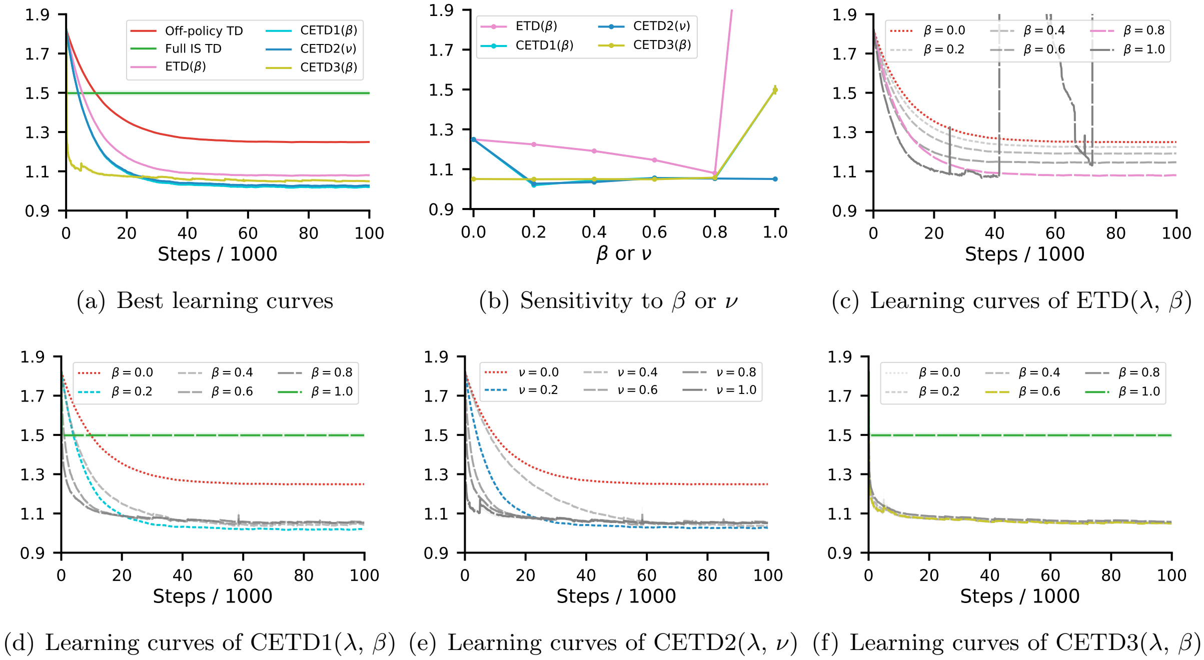
In this section, we present the results for different algorithms with multi-step bootstrapping. Specifically, we studied two values of λ: $\{0.5, 0.9\}$, which correspond to different levels of bootstrapping.

Results on the Two-State Task

Figures 7.6 and 7.7 show the results of different algorithms with multi-step bootstrapping on the Two-state task. The conclusion is similar to the one-step case presented in Section 7.1 except that the biases of Off-policy TD(λ) and ETD(λ, β) reduce significantly as λ increases.

Results on the Rooms Task

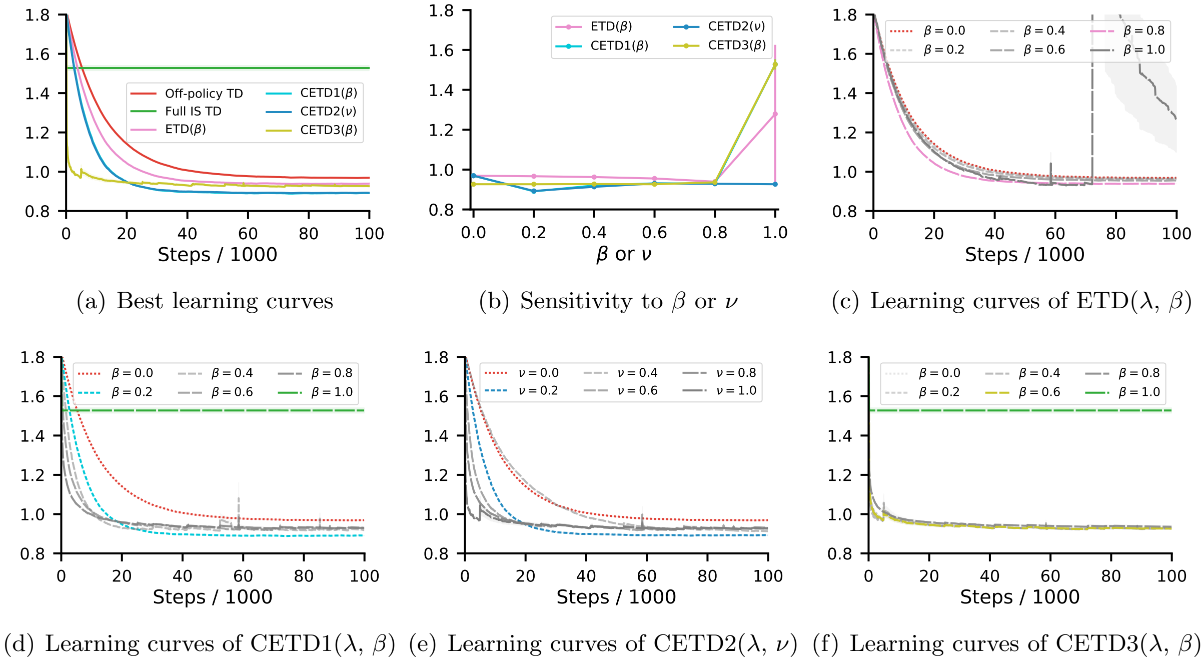
Figures 7.8 and 7.9 show the results of different algorithms with multi-step bootstrapping on the Rooms task. The conclusion is similar to the one-step case presented in Section 7.2 except that the biases of Off-policy TD(λ) and ETD(λ, β) reduce significantly as λ increases.



(a) Best learning curves (b) Sensitivity to β or ν (c) Learning curves of ETD(λ, β)

(d) Learning curves of CETD1(λ, β) (e) Learning curves of CETD2(λ, ν) (f) Learning curves of CETD3(λ, β)

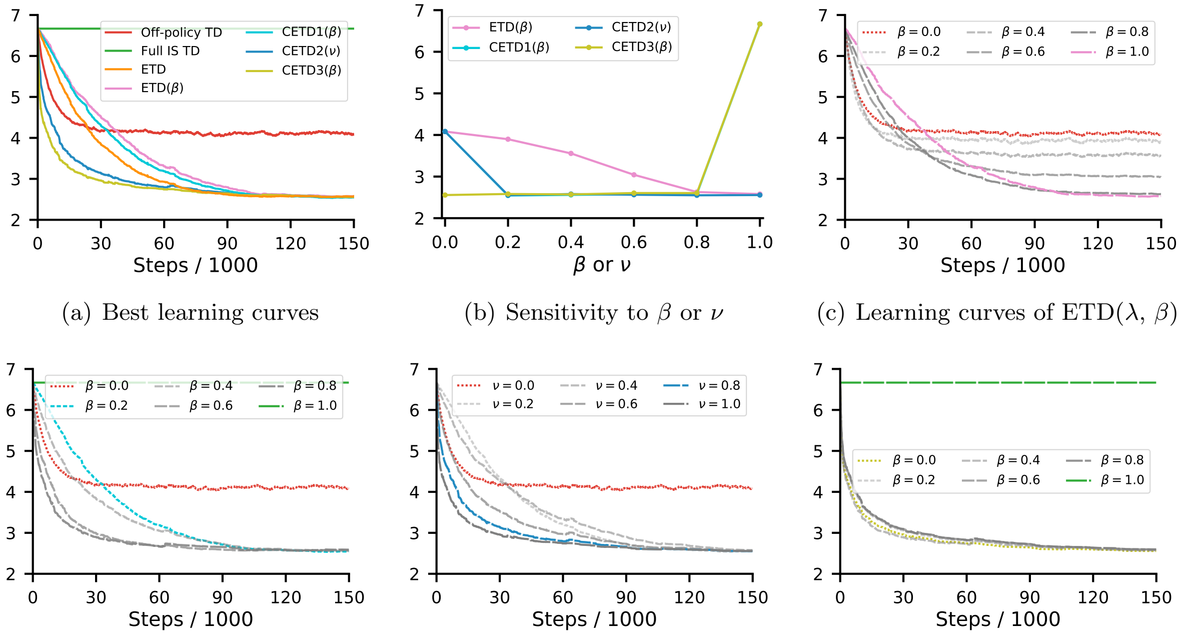
Figure 7.6: Results on the Two-state task ($\lambda = 0.5$). The y-axis shows $\overline{\text{RMSVE}}$.



(a) Best learning curves (b) Sensitivity to β or ν (c) Learning curves of ETD(λ, β)

(d) Learning curves of CETD1(λ, β) (e) Learning curves of CETD2(λ, ν) (f) Learning curves of CETD3(λ, β)

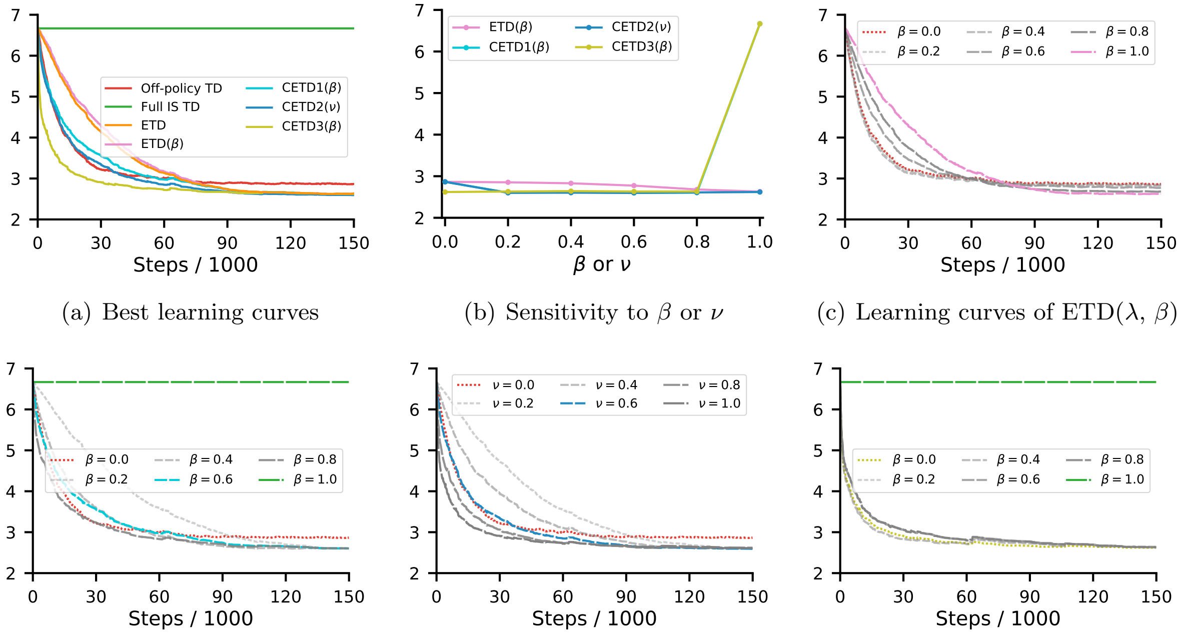
Figure 7.7: Results on the Two-state task ($\lambda = 0.9$). The y-axis shows $\overline{\text{RMSVE}}$.



(a) Best learning curves (b) Sensitivity to β or ν (c) Learning curves of ETD(λ, β)

(d) Learning curves of CETD1(λ, β) (e) Learning curves of CETD2(λ, ν) (f) Learning curves of CETD3(λ, β)

Figure 7.8: Results on the Rooms task ($\lambda = 0.5$). The y-axis shows $\overline{\text{RMSVE}}$.



(a) Best learning curves (b) Sensitivity to β or ν (c) Learning curves of ETD(λ, β)

(d) Learning curves of CETD1(λ, β) (e) Learning curves of CETD2(λ, ν) (f) Learning curves of CETD3(λ, β)

Figure 7.9: Results on the Rooms task ($\lambda = 0.9$). The y-axis shows $\overline{\text{RMSVE}}$.

7.4 Step Size Sensitivity

In this section, we provide the sensitivity analysis of all the experiments presented in previous sections. We aggregate the results in Figure 7.10 for convenient comparisons across different dimensions. We will discuss in order the following aspects:

- The effect of an algorithm's decay parameter (β or ν) on its step size sensitivity.
- The comparison of the step size sensitivity of different algorithms on a single task.
- The effect of an algorithm's bootstrapping parameter (λ) on its step size sensitivity.
- The comparison of the step size sensitivity of different algorithms across different tasks.

Firstly, from the top-left corner of Figure 7.10, we can see how different values of β affect the step size sensitivity of CETD1(0, β). Specifically, on the left extreme ($\beta = 0$), CETD1(0, β) becomes Off-policy TD(0), which is the least sensitive algorithm but converges to solutions with high errors. On the right extreme ($\beta = 1$), CETD1(0, β) degenerates into Full IS TD(0), which is the most sensitive and learns extremely slowly. While CETD1(0, β) with all intermediate values of β achieves significantly lower errors, it also has an intermediate sensitivity to the step size. In summary, the sensitivity to the step size will increase as the decay parameter increase. This pattern can also be validated in other plots in the figure except for those completely flat curves that represent no sign of learning of Full IS TD(λ).

Next, we compare different one-step ($\lambda = 0$) algorithms' step size sensitivity on the Two-state task from the top row of Figure 7.10. It's quite obvious that ETD(0, β) is the least sensitive across different values of the decay parameter, while CETD3(0, β) is at the other extreme. In addition, their best-performing step sizes for different values of the decay parameter are quite similar, which is not the case for CETD1(0, β) and CETD2(0, ν). Nevertheless, the latter two algorithms with a decay parameter with a value of 0.2 present low sensitivity while achieving the lowest error. These observations remain valid for other rows in the figure.

Further, the leftmost plots of the top three lines provide insights into how different values of λ impact the step size sensitivity of CETD1(λ, β). Notably, as λ increases, we observe four significant findings. Firstly, CETD1(λ, β) yields lower errors across different values of the decay parameter. Secondly, the method becomes increasingly sensitive to step size due to higher variance. Thirdly, the difference in error between Off-policy TD(λ) ($\beta = 0$) and CETD1(λ, β) ($0 < \beta < 1$) diminishes. Finally, the sensitivity curve shifts toward smaller step sizes. These observations are consistent with those found in CETD2(λ, ν) and ETD(λ, β).

Finally, we compare the sensitivity of different one-step ($\lambda = 0$) algorithms to step size on two different tasks presented in the first and fourth rows of Figure 7.10. Our observations reveal that

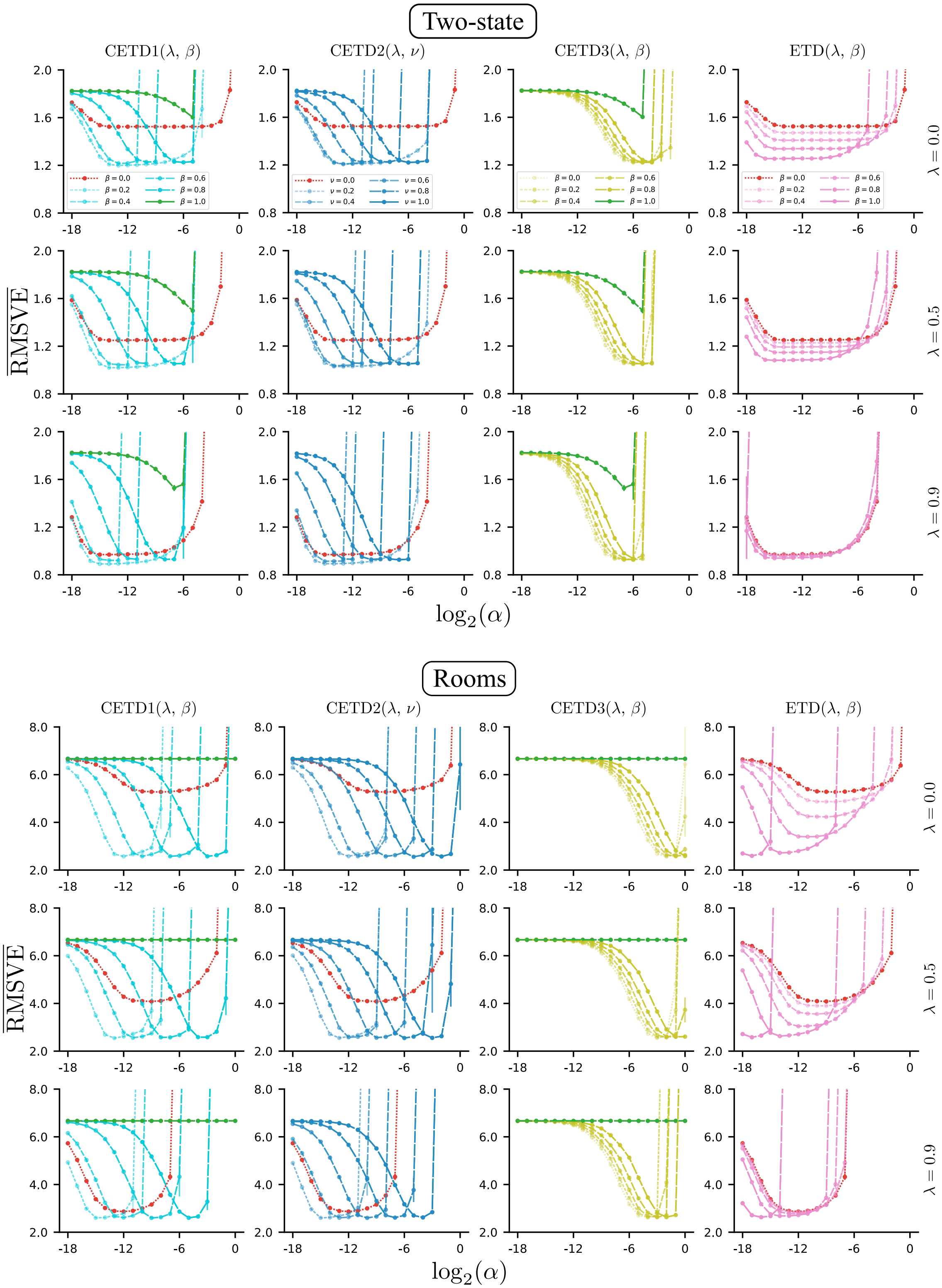


Figure 7.10: Step size sensitivity.

algorithms exhibit greater sensitivity in the Rooms task, which has a higher variance, compared to the Two-state task. This is especially notable for algorithms that were previously found to be less sensitive in the Two-state task. We propose two possible explanations for this phenomenon. Firstly, the shrinkage of the suitable step size range may become smaller as the variance of the task increases. Alternatively, the difference could be due to the ways in which the results are summarized. We remind the reader that the results for the Two-state task were averaged over all 100 runs, while the results for the Rooms task were averaged over the middle 15 runs.

In summary, higher variance can lead to greater sensitivity to the step size. In the case of CETD algorithms, reducing variance through a small decay parameter can improve usability. This is supported by the above analysis, which showed that a small decay parameter resulted in the lowest error while also reducing sensitivity to changes in the step-size parameter. Therefore, using a small decay parameter may be an effective way to optimize the performance of CETD algorithms.

7.5 The Bias-Variance Trade-Off

Having presented the advantages of the CETD instances in a didactic example and a more complex task with high variance, we now focus on analyzing the bias-variance trade-off that β and ν control to provide more insights on the properties of CETD(β, ν) in the one-step case. Specifically, we look at the bias and variance of the general followon trace F_t in (5.1) for different algorithms. Ideally, the random variable F_t given $S_t = s$ should have an expectation of $\frac{\mathbb{P}_\pi(S_t=s)}{\mathbb{P}_\mu(S_t=s)}$, which converges to the density ratio in the limit and corrects the distribution of the updates back to the on-policy distribution. Full IS TD is such an algorithm with zero bias. However, the variance of Full IS TD's F_t is unbearably high. For the CETD instances, their F_ts have a non-zero bias within finite time steps, but the bias converges to zero asymptotically. The benefit of having some transient bias in their F_ts is that the CETD instances can enjoy a lower variance. On the other hand, Scaled ETD(β)'s F_t has a much lower variance but a persistent bias. We performed experiments on the Two-state task. In order to obtain an accurate estimation, we sample $100,000$ trajectories of length 30 to calculate the bias and variance of F_t. The results are presented in Figure 7.11.

From Figure 7.11, we can see that, for all algorithms, as the decay parameter (β or ν) becomes larger, the bias will decrease, and the variance will increase. However, the magnitude and the speed of the changes differ. For CETD1(β), the bias and variance curves are quite symmetric. On the leftmost point with $\beta = 0$, CETD1(β) becomes Off-policy TD, whose F_t has the lowest variance but the largest bias; on the right-most point with $\beta = 1$, CETD1(β) becomes Full IS TD, whose F_t has the lowest bias but the largest variance; the points in the middle achieve a good bias-variance trade-off. Note that Scaled ETD(β) also connects Off-policy TD and Full IS TD. However, it is not a consistent method and can only become less biased as β approaches one. The result for

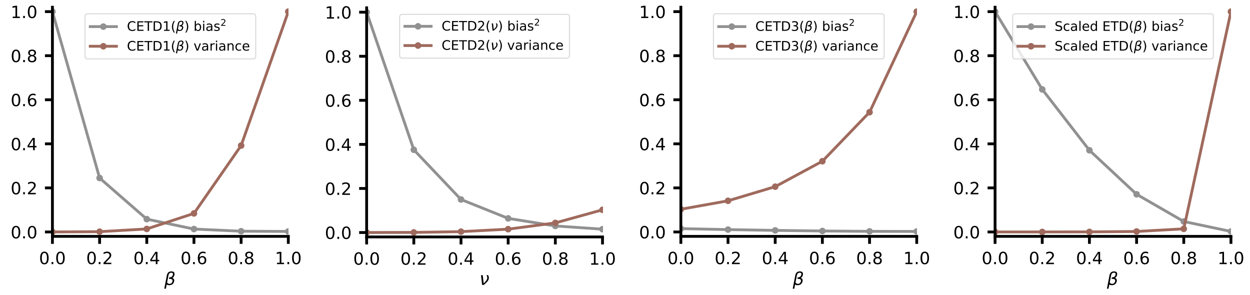


Figure 7.11: Bias-variance trade-off of different algorithms. The y-axis shows the normalized bias and variance of F_t.

Scaled ETD(β) also indicates the bias will only decrease to a relatively low level when β is quite large (in this case, at least 0.8), not to mention that its bias is persistent. In contrast, the bias of the CETD instances' F_ts will fade away as more time steps are given. Besides, CETD2(ν) and CETD3(β) together form a polygonal line connecting Off-policy TD and Full IS TD. The bias and the variance curves of the two algorithms combined form a similar shape to that of CETD1(β) but much wider. As a result, these two algorithms are less sensitive to the decay parameter but also put themselves at risk of not achieving the best trade-off. Generally, CETD2(ν) would hold the best trade-off point in tasks with high variance.

In summary, the analysis illustrates how the decay parameters β and ν affect the bias and variance of F_t, providing insights into the property of the corresponding algorithm.

7.6 Summary

In this chapter, we investigated the empirical properties of our main algorithm, CETD(λ, β, ν). Specifically, we illustrated the benefit of CETD(λ, β, ν)'s consistency by showing its low prediction errors in a didactic task compared to several existing algorithms. Further, we demonstrated that CETD(λ, β, ν) is competitive among existing algorithms and much more practical than Full IS TD(λ), the only consistent algorithm under general linear function approximation in the literature. These results suggested that CETD(λ, β, ν) is the first practical, consistent algorithm under general function approximation. Furthermore, we provided step size sensitivity results for all the experiments as well as a bias-variance analysis to understand the bias-variance landscape of GETD(λ, β, ν).

Chapter 8

Conclusions

In this thesis, we introduced Consistent Emphatic TD (CETD(λ, β, ν)), a novel off-policy TD learning algorithm that ensures consistency and outperforms existing methods. Our work encompassed the development, theoretical analysis, and empirical evaluation of CETD(λ, β, ν), providing a comprehensive understanding of its properties and potential applications.

The development of CETD(λ, β, ν) originated from one-step Average Emphatic TD (AETD(0)), a novel consistent algorithm that strikes a better balance between bias and variance, inspired by Full IS TD(λ) (Precup et al., 2001) and ETD(λ) (Sutton et al., 2016). AETD(0) utilizes incomplete IS-ratio products to avoid the high variance of the full IS-ratio product in Full IS TD(λ), similar to ETD(λ). However, unlike ETD(λ), the incomplete IS-ratio products in AETD(0) are weighted uniformly instead of geometrically, resulting in a consistent algorithm with fading bias. Next, we unified AETD(0) with Off-policy TD(0) (Precup, 2000) and Full IS TD(0) by introducing extra hyperparameters to achieve a smooth bias-variance trade-off, inspired by ETD(λ, β) (Hallak et al., 2016). Surprisingly, the unified algorithm, one-step General Emphatic TD (GETD((0, β, ν)), also subsumes Scaled ETD(0, β), a new variant of ETD(0, β). We then extended AETD(0) and GETD(0, β, ν) to the multi-step bootstrapping case. The resulting multi-step unified algorithm, GETD(λ, β, ν), is the most general off-policy TD algorithm to date, providing a connection between existing algorithms, including Off-policy TD(λ), Full IS TD(λ), ETD(λ, β), and AETD(λ), and giving rise to our main algorithm, CETD(λ, β, ν).

Consistent Emphatic TD (CETD(λ, β, ν)) is a highly desirable off-policy TD learning algorithm with several theoretical properties. Firstly, unlike ETD(λ, β), CETD(λ, β, ν) is guaranteed to be stable, regardless of the values of its hyperparameters. Secondly, while ETD(λ, β) has a biased fixed point, CETD(λ, β, ν) shares the same fixed point as On-policy TD(λ). Thirdly, CETD(λ, β, ν) offers a flexible and robust bias-variance trade-off that can effectively combat high variance, making it a highly effective remedy to the high variance of Full IS TD(λ), the only consistent method previously. To our knowledge, CETD(λ, β, ν) is the first practical, consistent algorithm capable of learning effectively for off-policy TD learning with general linear function approximation. We can obtain three instances of CETD(λ, β, ν) by constraining the value of its hyperparameters, resulting in the same number of hyperparameters as ETD(λ, β), which makes it more convenient to empirically evaluate and use.

To evaluate the empirical properties of CETD(λ, β, ν), we conducted experiments on both a didactic Two-state task and a more complex Rooms task with high variance. The results of the Two-state task experiments showed that CETD(λ, β, ν) had lower prediction errors, thereby validating the benefit of its consistency. Furthermore, experiments on the Rooms task demonstrated the effectiveness of CETD(λ, β, ν) on tasks with high variance, highlighting its superior practicality over Full IS TD(λ), the only existing consistent algorithm. We also provided step size sensitivity results for these experiments, as well as a bias-variance analysis to gain insight into the bias-variance trade-off of CETD(λ, β, ν).

In summary, CETD(λ, β, ν) is a novel off-policy TD learning algorithm that offers a robust bias-variance trade-off and ensures stability and consistency. Its development is based on the unification of existing algorithms and a new consistent algorithm called AETD(λ), which draws inspiration from these existing algorithms. This approach provides a highly effective solution to the high variance of the only consistent algorithm available before, filling a long-standing gap in the literature. Empirical evaluation on both a didactic task and a high variance task demonstrates the benefit of its consistency and practicality of CETD(λ, β, ν). In conclusion, this thesis presents a significant contribution to the field of off-policy reinforcement learning by introducing a novel and effective algorithm.

Limitations and Future Work

Our work has limitations in several aspects. Firstly, the theoretical analysis presented in this thesis is limited. Similar to Sutton et al. (2016), we provided a stability guarantee for CETD(λ, β, ν), which is a necessary condition for its convergence. However, proving the convergence of CETD(λ, β, ν) poses significant technical challenges, similar to the proof of ETD(λ)'s convergence (Yu, 2015, 2016). Thus, this remains an avenue for future work. Nonetheless, we have also provided a consistency guarantee for CETD(λ, β, ν).

In addition to the convergence analysis, it is crucial to understand the sample complexity of our method and how it compares to alternative approaches. Our algorithm is closely related to ETD(λ), for which no sample complexity analysis has been conducted to date. However, Guan et al. (2022) reported that as the variance of ETD(λ) grows exponentially with the number of iterations, it requires an exponentially large number of samples to converge. To address this issue, they proposed PER-ETD(λ), which periodically restarts the followon trace to control variance. This method was shown to converge to the same fixed point as ETD(λ) with a polynomial sample complexity. It is worth noting that the idea of cutting off the followon trace was also explored by Zhang and Whiteson (2022), who used a fixed truncation length to control variance. However, their analysis leads to convergence to a biased fixed point that depends on the truncation length but provides much better variance control. It would be interesting to apply these techniques and analyses to determine the sample complexity of CETD(λ, β, ν).

Another limitation of the theoretical aspect of our work is the lack of comprehensive analysis of the optimality of off-policy TD learning methods. Although CETD(λ, β, ν) has the same fixed point as the on-policy fixed point, we did not thoroughly explore its optimality. While we were motivated by the on-policy error bound provided by Tsitsiklis and Van Roy (1996), ETD(λ, β) was also equipped with error bounds under different measures (Hallak et al., 2016). Generally, the on-policy measure and the corresponding on-policy fixed point are preferred, but Hallak et al. (2016) showed that the fixed point of ETD(λ, β) could achieve lower approximation error to the true value function than the on-policy fixed point in certain scenarios. Thus, understanding the conditions under which different fixed points are optimal remains an interesting future direction for research.

In terms of empirical evaluation, although we provided empirical evidence of the effectiveness of CETD(λ, β, ν) in two examples with linear function approximation, its performance in other scenarios remains to be evaluated. Future work should investigate how the method performs in cases with extremely high variance, and it may be worthwhile to explore the techniques proposed by Zhang and Whiteson (2022) and Guan et al. (2022) in such cases. Additionally, extending the method to more complex problems that require nonlinear function approximation is another important direction to pursue. Previous research has already extended ETD(λ) to parameterized emphasis and deep reinforcement learning (Zhang et al., 2020c; Jiang et al., 2021, 2022). However, it is important to note that these methods also rely on assumptions about feature representation. Therefore, to obtain a comprehensive understanding, it's necessary to compare the performance of CETD(λ, β, ν) in combination with these methods to the density-ratio-based techniques discussed in Section 3.2.

Finally, while we focus on off-policy prediction problems and TD methods, the consistent followon trace could also be adapted to off-policy control problems and policy gradient methods. For off-policy control problems with TD learning, Zhang and Whiteson (2022) managed to apply the truncated followon trace to the control setting. For off-policy policy gradient, the original followon trace has been extended to estimate the gradient of the corresponding off-policy objective for general policy parametrization (Imani et al., 2018; Graves et al., 2021). Studying the potential benefits of the consistent followon trace in these scenarios would also be an interesting research direction.

Bibliography

Agarwal, R., Schwarzer, M., Castro, P. S., Courville, A. C., and Bellemare, M. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems*, volume 34, pp. 29304–29320, 2021.

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

Bellman, R. E. *Dynamic Programming.* Dover Publications, Inc., 2003.

Borkar, V. *Stochastic Approximation: A Dynamical Systems Viewpoint.* Cambridge University Press, 2008.

Dayan, P. The convergence of TD (λ) for general λ. *Machine learning*, 8:341–362, 1992.

Ghiassian, S. and Sutton, R. S. An empirical comparison of off-policy prediction learning algorithms in the four rooms environment. *arXiv preprint arXiv:2109.05110*, 2021.

Ghosh, D. and Bellemare, M. G. Representations for stable off-policy reinforcement learning. In *International Conference on Machine Learning*, pp. 3556–3565. PMLR, 2020.

Graves, E., Imani, E., Kumaraswamy, R., and White, M. Off-policy actor-critic with emphatic weightings. *arXiv preprint arXiv:2111.08172*, 2021.

Guan, Z., Xu, T., and Liang, Y. PER-ETD: A polynomially efficient emphatic temporal difference learning method. In *International Conference on Learning Representations*, 2022.

Hallak, A. and Mannor, S. Consistent on-line off-policy evaluation. In *International Conference on Machine Learning*, pp. 1372–1383. PMLR, 2017.

Hallak, A., Tamar, A., Munos, R., and Mannor, S. Generalized emphatic temporal difference learning: Bias-variance analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016.

Imani, E., Graves, E., and White, M. An off-policy policy gradient theorem using emphatic weightings. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Jiang, R., Zahavy, T., Xu, Z., White, A., Hessel, M., Blundell, C., and Van Hasselt, H. Emphatic algorithms for deep reinforcement learning. In *International Conference on Machine Learning*, pp. 5023–5033. PMLR, 2021.

Jiang, R., Zhang, S., Chelu, V., White, A., and van Hasselt, H. Learning expected emphatic traces for deep RL. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

Knuth, D. E. Two notes on notation. *American Mathematical Monthly*, 99(5):403–422, 1992.

Kolter, J. The fixed points of off-policy TD. In *Advances in Neural Information Processing Systems*, volume 24, 2011.

Lee, D., Seo, H., and Jung, M. W. Neural basis of reinforcement learning and decision making. *Annual Review of Neuroscience*, 35:287–308, 2012.

Liu, Q., Li, L., Tang, Z., and Zhou, D. Breaking the curse of horizon: Infinite-horizon off-policy estimation. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Mahmood, A. R. *Incremental Off-Policy Reinforcement Learning Algorithms.* PhD thesis, University of Alberta, 2017.

Manek, G. and Kolter, J. Z. The pitfalls of regularization in off-policy TD learning. In *Advances in Neural Information Processing Systems*, 2022.

Montague, P. R., Dayan, P., and Sejnowski, T. J. A framework for mesencephalic dopamine systems based on predictive hebbian learning. *Journal of Neuroscience*, 16(5):1936–1947, 1996.

Nachum, O., Chow, Y., Dai, B., and Li, L. DualDICE: Behavior-agnostic estimation of discounted stationary distribution corrections. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Precup, D. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, pp. 80, 2000.

Precup, D., Sutton, R. S., and Dasgupta, S. Off-policy temporal-difference learning with function approximation. In *International Conference on Machine Learning*, pp. 417–424, 2001.

Schultz, W., Dayan, P., and Montague, P. R. A neural substrate of prediction and reward. *Science*, 275(5306):1593–1599, 1997.

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

Sutton, R. S. and Barto, A. G. *Reinforcement Learning: An Introduction.* MIT press, 2018.

Sutton, R. S., Precup, D., and Singh, S. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1-2):181–211, 1999.

Sutton, R. S., Mahmood, A. R., and White, M. An emphatic approach to the problem of off-policy temporal-difference learning. *Journal of Machine Learning Research*, 17(1):2603–2631, 2016.

Sutton, R. S., Bowling, M. H., and Pilarski, P. M. The Alberta Plan for AI research. *arXiv preprint arXiv:2208.11173*, 2022.

Tsitsiklis, J. and Van Roy, B. Analysis of temporal-diffference learning with function approximation. In *Advances in Neural Information Processing Systems*, volume 9, 1996.

Varga, R. *Matrix Iterative Analysis.* Springer Berlin Heidelberg, 1999.

Watkins, C. J. C. H. *Learning from Delayed Rewards.* PhD thesis, King's College, 1989.

White, A., Modayil, J., and Sutton, R. S. Scaling life-long off-policy learning. In *2012 IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL)*, pp. 1–6. IEEE, 2012.

Yu, H. On convergence of emphatic temporal-difference learning. In *Conference on Learning Theory*, pp. 1724–1751. PMLR, 2015.

Yu, H. Weak convergence properties of constrained emphatic temporal-difference learning with constant and slowly diminishing stepsize. *Journal of Machine Learning Research*, 17(1):7745–7802, 2016.

Zhang, R., Dai, B., Li, L., and Schuurmans, D. GenDICE: Generalized offline estimation of stationary values. In *International Conference on Learning Representations*, 2020a.

Zhang, S. and Whiteson, S. Truncated emphatic temporal difference methods for prediction and control. *Journal of Machine Learning Research*, 23(153):1–59, 2022.

Zhang, S., Liu, B., and Whiteson, S. GradientDICE: Rethinking generalized offline estimation of stationary values. In *International Conference on Machine Learning*, pp. 11194–11203. PMLR, 2020b.

Zhang, S., Liu, B., Yao, H., and Whiteson, S. Provably convergent two-timescale off-policy actor-critic with function approximation. In *International Conference on Machine Learning*, pp. 11204–11213. PMLR, 2020c.

Appendix A

Update Rules for Relevant Algorithms

This chapter includes the update rules for the algorithms mentioned in the thesis.

Off-policy TD(λ):

$$\begin{aligned}
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\delta_t\mathbf{z}_t,\\
\delta_t &= R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top\boldsymbol{\theta}_t,\\
\mathbf{z}_t &= \rho_t(\gamma\lambda\mathbf{z}_{t-1} + \boldsymbol{\phi}_t), \text{with } \mathbf{z}_{-1} = \mathbf{0}.
\end{aligned}$$

Full IS TD(λ):

$$\begin{aligned}
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\delta_t\mathbf{z}_t,\\
\delta_t &= R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top\boldsymbol{\theta}_t,\\
\mathbf{z}_t &= \rho_t(\gamma\lambda\mathbf{z}_{t-1} + F_t\boldsymbol{\phi}_t), \text{with } \mathbf{z}_{-1} = \mathbf{0},\\
F_t &= \rho_{t-1}F_{t-1}, \text{with } F_0 = 1.
\end{aligned}$$

ETD(λ):

$$\begin{aligned}
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\delta_t\mathbf{z}_t,\\
\delta_t &= R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top\boldsymbol{\theta}_t,\\
\mathbf{z}_t &= \rho_t(\gamma\lambda\mathbf{z}_{t-1} + M_t\boldsymbol{\phi}_t), \text{with } \mathbf{z}_{-1} = \mathbf{0},\\
M_t &= (1-\lambda)F_t + \lambda,\\
F_t &= \gamma\rho_{t-1}F_{t-1} + 1, \text{with } F_0 = 1.
\end{aligned}$$

ETD(λ, β):

$$\begin{aligned}
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\delta_t\mathbf{z}_t,\\
\delta_t &= R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top\boldsymbol{\theta}_t,\\
\mathbf{z}_t &= \rho_t(\gamma\lambda\mathbf{z}_{t-1} + M_t\boldsymbol{\phi}_t), \text{with } \mathbf{z}_{-1} = \mathbf{0},\\
M_t &= (1-\lambda)F_t + \lambda,\\
F_t &= \beta\rho_{t-1}F_{t-1} + 1, \text{with } F_0 = 1.
\end{aligned}$$

Scaled ETD(λ):

$$\begin{aligned}
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\delta_t\mathbf{z}_t,\\
\delta_t &= R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top\boldsymbol{\theta}_t,\\
\mathbf{z}_t &= \rho_t(\gamma\lambda\mathbf{z}_{t-1} + M_t\boldsymbol{\phi}_t), \text{with } \mathbf{z}_{-1} = \mathbf{0},\\
M_t &= (1-\lambda)F_t + \lambda(1-\gamma),\\
F_t &= \gamma\rho_{t-1}F_{t-1} + (1-\gamma), \text{with } F_0 = 1.
\end{aligned}$$

Scaled ETD(λ, β):

$$\begin{aligned}
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\delta_t\mathbf{z}_t,\\
\delta_t &= R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top\boldsymbol{\theta}_t,\\
\mathbf{z}_t &= \rho_t(\gamma\lambda\mathbf{z}_{t-1} + M_t\boldsymbol{\phi}_t), \text{with } \mathbf{z}_{-1} = \mathbf{0},\\
M_t &= (1-\lambda)F_t + \lambda(1-\beta),\\
F_t &= \beta\rho_{t-1}F_{t-1} + (1-\beta), \text{with } F_0 = 1.
\end{aligned}$$

AETD(λ):

$$\begin{aligned}
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\delta_t\mathbf{z}_t,\\
\delta_t &= R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top\boldsymbol{\theta}_t,\\
\mathbf{z}_t &= \rho_t(\gamma\lambda\mathbf{z}_{t-1} + M_t\boldsymbol{\phi}_t), \text{with } \mathbf{z}_{-1} = \mathbf{0},\\
M_t &= (1-\lambda g(t))F_t + \lambda g(t),\\
F_t &= (1-g(t))\rho_{t-1}F_{t-1} + g(t), \text{with } F_0 = 1,\\
g(t) &= (t+1)^{-1}.
\end{aligned}$$

CETD(λ, β, ν):

$$\begin{aligned}
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\delta_t\mathbf{z}_t,\\
\delta_t &= R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top\boldsymbol{\theta}_t,\\
\mathbf{z}_t &= \rho_t(\gamma\lambda\mathbf{z}_{t-1} + M_t\boldsymbol{\phi}_t), \text{with } \mathbf{z}_{-1} = \mathbf{0},\\
M_t &= (1-\lambda h(t))F_t + \lambda g(t),\\
F_t &= (1-g(t))\rho_{t-1}F_{t-1} + g(t), \text{with } F_0 = 1,\\
h(t) &= (1-\beta)^\nu(t+1)^{-\nu},\\
g(t) &= (1-\beta)(t+1)^{-\nu}.
\end{aligned}$$

CETD1(λ, β):

$$\begin{aligned}
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\delta_t\mathbf{z}_t,\\
\delta_t &= R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top\boldsymbol{\theta}_t,\\
\mathbf{z}_t &= \rho_t(\gamma\lambda\mathbf{z}_{t-1} + M_t\boldsymbol{\phi}_t), \text{with } \mathbf{z}_{-1} = \mathbf{0},\\
M_t &= (1-\lambda h(t))F_t + \lambda g(t),\\
F_t &= (1-g(t))\rho_{t-1}F_{t-1} + g(t), \text{with } F_0 = 1,\\
h(t) &= (1-\beta)^\beta(t+1)^{-\beta},\\
g(t) &= (1-\beta)(t+1)^{-\beta}.
\end{aligned}$$

CETD2(λ, ν):

$$\begin{aligned}
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\delta_t\mathbf{z}_t,\\
\delta_t &= R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top\boldsymbol{\theta}_t,\\
\mathbf{z}_t &= \rho_t(\gamma\lambda\mathbf{z}_{t-1} + M_t\boldsymbol{\phi}_t), \text{with } \mathbf{z}_{-1} = \mathbf{0},\\
M_t &= (1-\lambda g(t))F_t + \lambda g(t),\\
F_t &= (1-g(t))\rho_{t-1}F_{t-1} + g(t), \text{with } F_0 = 1,\\
g(t) &= (t+1)^{-\nu}.
\end{aligned}$$

CETD3(λ, β):

$$\begin{aligned}
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \alpha\delta_t\mathbf{z}_t,\\
\delta_t &= R_{t+1} + \gamma\boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t - \boldsymbol{\phi}_t^\top\boldsymbol{\theta}_t,\\
\mathbf{z}_t &= \rho_t(\gamma\lambda\mathbf{z}_{t-1} + M_t\boldsymbol{\phi}_t), \text{with } \mathbf{z}_{-1} = \mathbf{0},\\
M_t &= (1 - \lambda g(t))F_t + \lambda g(t),\\
F_t &= (1 - g(t))\rho_{t-1}F_{t-1} + g(t), \text{with } F_0 = 1,\\
g(t) &= (1 - \beta)(t + 1)^{-1}.
\end{aligned}$$