PID Accelerated Temporal Difference Algorithms

Mark Bedaywi* **Amin Rakhsha*** **Amir-massoud Farahmand**
Department of Computer Science, University of Toronto
Vector Institute

Abstract

Long-horizon tasks, which have a large discount factor, pose a challenge for most conventional reinforcement learning (RL) algorithms. Algorithms such as Value Iteration and Temporal Difference (TD) learning have a slow convergence rate and become inefficient in these tasks. When the transition distributions are given, PID VI was recently introduced to accelerate the convergence of Value Iteration using ideas from control theory. Inspired by this, we introduce PID TD Learning and PID Q-Learning algorithms for the RL setting, in which only samples from the environment are available. We give a theoretical analysis of the convergence of PID TD Learning and its acceleration compared to the conventional TD Learning. We also introduce a method for adapting PID gains in the presence of noise and empirically verify its effectiveness.

1 Introduction

The Value Iteration (VI) algorithm is one of the primary dynamic programming methods for solving (discounted) Markov Decision Processes (MDP). It is the foundation of many Reinforcement Learning (RL) algorithms such as the Temporal Difference (TD) Learning (Sutton, 1988; Tsitsiklis and Van Roy, 1997), Q-Learning (Watkins, 1989), Approximate/Fitted Value Iteration (Gordon, 1995; Ernst et al., 2005; Munos and Szepesvári, 2008; Tosatto et al., 2017), and DQN (Mnih et al., 2015; Van Hasselt et al., 2016), which can all be seen as sample-based variants of VI. A weakness of the VI algorithm and the RL algorithms built on top of it is their slow convergence in problems with discount factor γ close to 1, which corresponds to the long-horizon problems where the agent aims to maximize its cumulative rewards far in the future. One can show that the error of the value function calculated by VI at iteration k goes to zero with the slow rate of $\mathcal{O}(\gamma^k)$. The slow convergence rate when $\gamma \approx 1$ also appears in the error analysis of the downstream temporal difference (Szepesvári, 1997; Even-Dar and Mansour, 2003; Wainwright, 2019) and fitted value iteration algorithms (Munos and Szepesvári, 2008; Farahmand et al., 2010; Chen and Jiang, 2019; Fan et al., 2019). If $\gamma \approx 1$, these algorithms become very slow and inefficient. This work introduces accelerated temporal difference learning algorithms that can mitigate this issue.

Farahmand and Ghavamzadeh (2021) recently suggested that one may view the iterates of VI as a dynamical system. This opens up the possibility of using tools from control theory to modify, and perhaps accelerate, the VI's dynamics. They specifically used the simple class of Proportional-Integral-Derivative (PID) controllers to modify VI, resulting in a new procedure called the PID VI algorithm. They showed that with a careful choice of the controller gains, PID VI can converge significantly faster than the conventional VI. They also introduced a gain adaptation mechanism, a meta-learning procedure, to automatically choose these gains.

PID VI, similar to VI, is a dynamic programming algorithm and requires access to the full transition dynamics of the environment. In the RL setting, however, the transition dynamics is not directly accessible to the agent; the agent can only acquire samples from the transition dynamics by interacting with the environment.

*These authors contributed equally to this work.

In this work, we show how the ideas of the PID VI algorithm can be used in the RL setting. Our contributions are:

- Introduce PID TD Learning and PID Q-Learning algorithms (Section 3) for the RL setting that show accelerated convergence compared to their conventional counterparts.
- Theoretically show the convergence and acceleration of the PID TD Learning (Section 4).
- A sample-based gain adaptation mechanism to automatically tune the controller gains, reducing the hyperparameter-tuning required for the algorithms (Section 5).

The new algorithms are a step towards RL algorithms that can tackle long-horizon tasks more efficiently.

2 Background

Given a set Ω, let $\mathcal{M}(\Omega)$ be the set of probability distributions over Ω, and $\mathcal{B}(\Omega)$ be the set of bounded functions over Ω. We consider a discounted MDP (Bertsekas and Tsitsiklis, 1996; Szepesvári, 2010; Sutton and Barto, 2018) defined as $(\mathcal{X}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$ where $\mathcal{X}$ is the finite set of n states, $\mathcal{A}$ is the finite set of m actions, $\mathcal{P}\colon \mathcal{X} \times \mathcal{A} \to \mathcal{M}(\mathcal{X})$ is the transition kernel, $\mathcal{R} : \mathcal{X} \times \mathcal{A} \to \mathcal{M}([0,1])$ is the reward function, and $\gamma \in [0,1)$ is the discount factor.

A policy π is a function $\pi\colon \mathcal{X} \to \mathcal{M}(\mathcal{A})$ representing the distribution over the actions an agent would take from each state. Given a policy π, the functions $V^\pi : \mathcal{X} \to \mathbb{R}$ and $Q^\pi\colon \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ are the corresponding (state-)value and action-value functions defined as the expected discounted return when following π starting at a certain state or state-action pair. We also let $\mathcal{P}^\pi\colon \mathcal{X} \to \mathcal{M}(\mathcal{X})$ and $\mathcal{R}^\pi\colon \mathcal{X} \to \mathcal{M}([0,1])$ be the associated transition and reward kernels of policy π, and $r^\pi : \mathcal{X} \to [0,1]$ be the expected reward of following π at any state.

The Policy Evaluation (PE) problem is the problem of finding the value function V^π corresponding to a given policy π and the Control problem is the problem of finding the policy π^* that maximizes the corresponding value function $Q^*(x,a) \triangleq Q^{\pi^*}(x,a) = \max_\pi Q^\pi(x,a)$, for each state x and action a. We shall use V whenever we talk about the PE problem and Q for the Control problem, for the brevity of the presentation.

The Bellman operator, T^π, and the Bellman optimality operator, $T^\star$, are defined as follows:

$$(T^\pi V)(x) \triangleq r^\pi(x) + \gamma \int \mathcal{P}^\pi(\mathrm{d}y \mid x) V(y), \qquad (\forall x \in \mathcal{X}),$$

$$(T^\star Q)(x,a) \triangleq r(x,a) + \gamma \int \mathcal{P}(\mathrm{d}y \mid x,a) \max_{a' \in \mathcal{A}} Q(y,a') \qquad (\forall x \in \mathcal{X}, a \in \mathcal{A}).$$

The Bellman residual operators are defined as $\mathrm{BR}^\pi V \triangleq T^\pi V - V$ (for PE) and $\mathrm{BR}^* Q \triangleq T^\star Q - Q$ (for Control). The value function V^π is the unique function with $\mathrm{BR}^\pi V^\pi = 0$ and Q^* is the unique function with $\mathrm{BR}^* Q^* = 0$.

The iteration $V_{k+1} \leftarrow T^\pi V_k$ converges to V^π, and the iteration $Q_{k+1} \leftarrow T^\star Q_k$ converges to Q^*. This is known as the VI algorithm. The convergence is due to the γ-contraction of the Bellman operators with respect to the supremum norm, and can be proven using the Banach fixed-point theorem. The result also shows that the convergence rate of VI is $\mathcal{O}(\gamma^k)$. This can be extremely slow for long horizon tasks with γ very close to 1.

2.1 PID Value Iteration

The PID VI algorithm (Farahmand and Ghavamzadeh, 2021) is designed to address the slow convergence of VI. The key observation is that the VI algorithm can be interpreted as a feedback control

system with the Bellman residual as the error signal. The conventional VI corresponds to a Proportional controller, perhaps the simplest form of controller. The PID VI algorithm uses a more general PID controller (Dorf and Bishop, 2008; Ogata, 2010) in the feedback loop instead.

A PID controller consists of three components (terms), which together determine the update of the value function from V_k to V_{k+1}, or from Q_k to Q_{k+1}. The *P component* is a rescaling of the Bellman residual itself, that is, $\mathrm{BR}^\pi V_k$ or $\mathrm{BR}^* Q_k$. The *D component* is the discrete derivative of the value updates, that is, $V_k - V_{k-1}$ or $Q_k - Q_{k-1}$. The *I component* is a running average of the Bellman residuals. The contribution of each of these terms to the value update is determined by controller gains $\kappa_p, \kappa_I, \kappa_d \in \mathbb{R}$.

To find the I component, we maintain a running average (hence, the name integration) of Bellman residual error by $z_k \colon \mathcal{X} \to \mathbb{R}$ for PE and $z_k \colon \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ in the Control case,

$$z_{k+1} = \beta z_k + \alpha \mathrm{BR}^\pi V_k \quad \text{(PE)} \qquad , \qquad z_{k+1} = \beta z_k + \alpha \mathrm{BR}^* Q_k \quad \text{(Control)}, \tag{1}$$

with $\alpha, \beta \in \mathbb{R}$ and z_1 initialized to a vector of all zeroes. PID VI updates the value function by

$$\begin{aligned} V_{k+1} &= V_k + \kappa_p \mathrm{BR}^\pi V_k + \kappa_I(\beta z_k + \alpha \mathrm{BR}^\pi V_k) + \kappa_d(V_k - V_{k-1}) \quad \text{(PE)}, \\ Q_{k+1} &= Q_k + \kappa_p \mathrm{BR}^* Q_k + \kappa_I(\beta z_k + \alpha \mathrm{BR}^* Q_k) + \kappa_d(Q_k - Q_{k-1}) \quad \text{(Control)}. \end{aligned} \tag{2}$$

This is a generalization of the conventional VI algorithm: VI corresponds to the choice of $(\kappa_p, \kappa_I, \kappa_d) = (1, 0, 0)$. PID VI has the same fixed point as the conventional VI, for both PE and Control. The dynamics of the sequence (V_k), however, depends on the controller gains $(\kappa_p, \kappa_I, \kappa_d)$ and (α, β) of the integrator. For some choices of the gains, the dynamics converges to the fixed point, the true value function, at an accelerated rate. We also note that the dynamics is not necessarily stable, and for some gains, it might diverge.

The choice of the gains that (maximally) accelerates convergence depends on the MDP and the policy being evaluated. One approach is to place assumptions on the structure of the MDP, and analytically derive the gains that optimize the convergence rate. Farahmand and Ghavamzadeh (2021) provide such a result for PE in the class of reversible Markov chains. Assuming structure on the MDP is not desirable though, so the same work also proposes a *gain adaptation* algorithm that automatically tunes the controller gains during the fixed point iteration.

The proposed gain adaptation algorithm performs gradient descent at each iteration in the direction that minimizes the squared Bellman residual. For added efficiency, it is normalized by the previous Bellman residual. Formally, we pick a meta-learning rate $\eta \in \mathbb{R}$ and for each gain $\kappa_\cdot \in \{\kappa_p, \kappa_I, \kappa_d\}$, after each iteration of PID VI, we perform

$$\kappa_\cdot \leftarrow \kappa_\cdot - \eta \frac{2}{\|\mathrm{BR}^\pi V_k\|_2^2} \frac{\partial \frac{1}{2} \|\mathrm{BR}^\pi V_{k+1}\|_2^2}{\partial \kappa_\cdot} = \kappa_\cdot - \eta \frac{1}{\|\mathrm{BR}^\pi V_k\|_2^2} \cdot \left\langle \mathrm{BR}^\pi V_{k+1}, \frac{\partial \mathrm{BR}^\pi V_{k+1}}{\partial \kappa_\cdot} \right\rangle. \tag{3}$$

The Control case is described similarly by substituting BR^π with BR^*. The PID VI algorithm (1)–(2) and its gain adaptation procedure (3) depend on the computation of the Bellman residual $\mathrm{BR}^\pi V$ or its gradient $\partial \mathrm{BR}^\pi V / \partial \kappa_\cdot$, both of which require accessing the transition dynamics $\mathcal{P}$. PID VI, like VI, is a dynamic programming/planning algorithm after all. They are not directly applicable to the RL setting, where the agent has access only to samples from the environment that are obtained online. The goal of the next few sections is to develop RL variants of these algorithms.

3 PID TD Learning and PID Q-Learning

We introduce the PID TD Learning as well as the PID Q-Learning algorithms. These are stochastic approximation versions of the PID VI algorithm and use samples in the form of (X_t, A_t, R_t, X'_t) with $A_t \sim \pi(\cdot|X_t)$ (for PE), $X'_t \sim \mathcal{P}(\cdot|X_t, A_t)$ and $R_t \sim \mathcal{R}(\cdot|X_t, A_t)$, instead of directly accessing $\mathcal{P}$ and $\mathcal{R}$. In a typical RL setting, they form a sequence with $X_{t+1} = X'_t$.

To generalize the PID VI procedure to the sample-based setting, we first describe each iteration of PID VI with an operator. This viewpoint allows translation of PID VI to a sample-based algorithm through stochastic approximation. The same translation applied to the Bellman operator, which is the update rule for VI, yields the conventional TD Learning and Q-Learning. PID VI for PE updates three functions $V, V', z : \mathcal{X} \to \mathbb{R}$ at each iteration. Here, V stores the value function, V' stores the previous value function, and z stores the running average of the Bellman errors. For the PID Q-Learning, the domain of these functions would be $\mathcal{X} \times \mathcal{A}$, that is, we have $Q, Q', z : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$. We shall use $\tilde{V} \colon \mathcal{X} \times \{\texttt{v, v', z}\} \to \mathbb{R}$ and $\tilde{Q} \colon \mathcal{X} \times \mathcal{A} \times \{\texttt{v, v', z}\} \to \mathbb{R}$ as a compact representation of these three functions. Here, $\{\texttt{v, v', z}\}$ is set of size 3 that indexes the three functions contained in $\tilde{V}$. Note that since we focus on finite MDPs, these functions can be represented by finite-dimensional vectors $\tilde{V} \in \mathbb{R}^{3n}$ and $\tilde{Q} \in \mathbb{R}^{3nm}$:

$$\tilde{V} = \begin{bmatrix} V \\ z \\ V' \end{bmatrix} \qquad , \qquad \tilde{Q} = \begin{bmatrix} Q \\ z \\ Q' \end{bmatrix}.$$

Also define $\tilde{V}^\pi \triangleq [V^\pi \; 0 \; V^\pi]^\top$ and $\tilde{Q}^* \triangleq [Q^* \; 0 \; Q^*]^\top$. Define the space of all possible choices of gains $(\kappa_p, \kappa_I, \kappa_d, \alpha, \beta)$ to be $\mathcal{G} \triangleq \mathbb{R}^5$. For a policy π and the controller gains $g \in \mathcal{G}$, we denote the PID VI operator on the space of $\mathcal{B}(\mathcal{X} \times \{\texttt{v, v', z}\})$ by L_g^π defined as

$$L_g^\pi \tilde{V} \triangleq \tilde{V} \mapsto \begin{bmatrix} V + \kappa_p \cdot \mathrm{BR}^\pi V + \kappa_I(\beta z + \alpha \cdot \mathrm{BR}^\pi V) + \kappa_d(V - V') \\ \beta z + \alpha \cdot \mathrm{BR}^\pi V \\ V \end{bmatrix}.$$

The operator L_g^* on $\mathcal{B}(\mathcal{X} \times \mathcal{A} \times \{\texttt{v, v', z}\})$ is defined analogously, replacing BR^π with BR^*. With these notations, the PID VI algorithm can be written as

$$\tilde{V}_{k+1} \leftarrow L_g^\pi \tilde{V}_k \quad \text{(PE)} \qquad , \qquad \tilde{Q}_{k+1} \leftarrow L_g^* \tilde{Q}_k \qquad \text{(Control)}.$$

Now we use stochastic approximation and this operator to derive our sample-based algorithms. At each iteration, the agent receives a sample (X_t, A_t, R_t, X_t') from the environment. Focus on PE and let $\tilde{V}_t$ be the compact form of functions V_t, z_t, V_t' at iteration t. To perform the stochastic approximation update on the value of $\tilde{V}_t(X_t, \texttt{f})$ for some $\texttt{f} \in \{\texttt{v, v', z}\}$, we need an unbiased estimator $\hat{L}_{t,\texttt{f}}$ of $(L_g^\pi \tilde{V}_t)(X_t, \texttt{f})$, which is a scalar random variable. Let $N_t(x)$ and $N_t(x, a)$ be the number of times state x and state-action x, a are visited by time t. We consider the learning rate schedule $\mu \colon \mathbb{Z} \to \mathbb{R}^+$ that maps the state count to the current learning rate. With the estimator $\hat{L}_{t,\texttt{f}}$, and state-count dependent learning rate $\mu(N_t(X_t))$, the update given by stochastic approximation is of the form

$$\tilde{V}_{t+1}(X_t, \texttt{f}) \leftarrow \tilde{V}_t(X_t, \texttt{f}) + \mu(N_t(X_t))(\hat{L}_{t,\texttt{f}} - \tilde{V}_t(X_t, \texttt{f})). \tag{4}$$

Note that all values of $\tilde{V}_{t+1}$ that are not assigned an updated value will remain the same.

The only term in $(L_g^\pi \tilde{V}_t)(X_t, \texttt{f})$ that requires estimation is $(\mathrm{BR}^\pi V_t)(X_t)$, which depends on the transition distributions of the MDP that is not available. We can form an unbiased estimate $\widehat{\mathrm{BR}}_t$ of $(\mathrm{BR}^\pi V_t)(X_t) = (T^\pi V)(X_t) - V(X_t)$ or $(\mathrm{BR}^* Q_t)(X_t, A_t) = (T^\star Q_t)(X_t, A_t) - Q_t(X_t, A_t)$ by

$$\widehat{\mathrm{BR}}_t = \begin{cases} R_t + \gamma V_t(X_t') - V_t(X_t) & \text{(PE)}, \\ R_t + \gamma \max_{a' \in \mathcal{A}} Q_t(X_t', a') - Q_t(X_t, A_t) & \text{(Control)}. \end{cases} \tag{5}$$

A stochastic approximation procedure can then be used to update the values $V_t(X_t)$, $z_t(X_t)$, $V_t'(X_t)$ according to (4). The procedure would be

$$\begin{aligned} V_{t+1}(X_t) &\leftarrow V_t(X_t) + \mu(N_t(X_t))\big[\kappa_p \widehat{\mathrm{BR}}_t + \kappa_I(\beta z_t(X_t) + \alpha \widehat{\mathrm{BR}}_t) + \kappa_d(V_t(X_t) - V_t'(X_t))\big], \\ z_{t+1}(X_t) &\leftarrow z_t(X_t) + \mu(N_t(X_t))\big[\beta z_t(X_t) + \alpha \widehat{\mathrm{BR}}_t - z_t(X_t)\big], \\ V_{t+1}'(X_t) &\leftarrow V_t'(X_t) + \mu(N_t(X_t))\big[V_t(X_t) - V_t'(X_t)\big]. \end{aligned} \tag{6}$$

We call this procedure the PID TD learning algorithm. For Control, we similarly form estimates $\hat{L}_{t,\mathtt{f}}$ of $(L_g^* Q_t)(X_t, A_t, \mathtt{f})$ and obtain

$$
\begin{aligned}
Q_{t+1}(X_t, A_t) &\leftarrow Q_t(X_t, A_t) + \mu_t\big[\kappa_p \widehat{\mathrm{BR}}_t + \kappa_I(\beta z_t(X_t, A_t) + \alpha \widehat{\mathrm{BR}}_t) + \kappa_d(Q_t(X_t, A_t) - Q'_t(X_t, A_t))\big], \\
z_{t+1}(X_t, A_t) &\leftarrow z_t(X_t, A_t) + \mu_t\big[\beta z_t(X_t, A_t) + \alpha \widehat{\mathrm{BR}}_t - z_t(X_t, A_t)\big], \\
Q'_{t+1}(X_t, A_t) &\leftarrow Q'_t(X_t, A_t) + \mu_t\big[Q_t(X_t, A_t) - Q'_t(X_t, A_t)\big]. \qquad (7)
\end{aligned}
$$

where $\mu_t = \mu(N_t(X_t, A_t))$. This is the PID Q-Learning algorithm. It is worth mentioning that one can use other forms of learning rates to achieve better practical results. For example, we can choose constant or state-count independent learning rates, or use three different learning rates for the three updates in (6) and (7). The formulation in this section is chosen for simplicity and the theoretical analysis.

4 Theoretical Guarantees

In this section, we focus on the PE problem and present the theoretical analysis of PID TD Learning. We show that with proper choices of controller gains that make PID VI convergent, PID TD Learning is also convergent. Then, under synchronous update setting, we provide insights on the accelerated convergence of PID TD Learning compared to the conventional TD Learning.

4.1 Convergence Guarantee

Farahmand and Ghavamzadeh (2021) show that PID VI converges under a wide range of gains for a wide range of environments both analytically and experimentally. We show that this convergence carries over to our sample-based PID TD Learning. We first need to define some notations to express our result. Note that L_g^π is an affine linear operator. Define A_g^π to be its linear component and b_g^π to be the constant component, so that $L_g^\pi \tilde{V} = A_g^\pi \tilde{V} + b_g^\pi$. In particular,

$$
A_g^\pi := \begin{bmatrix} (1 - \kappa_p + \kappa_d - \kappa_I \alpha) I + \gamma(\kappa_p + \kappa_I \alpha)\mathcal{P}^\pi & \beta\kappa_I I & -\kappa_d I \\ (-\alpha I + \gamma\alpha \mathcal{P}^\pi) & \beta I & 0 \\ I & 0 & 0 \end{bmatrix}.
$$

The matrix A_g^π plays a critical role in the behavior of PID VI as well as PID TD Learning. Farahmand and Ghavamzadeh (2021) show that PID VI is convergent for PE if $\rho(A_g^\pi) < 1$ where $\rho(M)$ for a square matrix M is its spectral radius, the maximum of the magnitude of the eigenvalues. It turns out the condition on the controller gains needed for the convergence of PID TD Learning is weaker than the one for PID VI. We provide the following result.

Theorem 1 (Convergence of PID TD)**.** *Consider a set of controller gains g. Let $\{\lambda_i\}$ be the eigenvalues of A_g^π. If $\mathrm{Re}\{\lambda_i\} < 1$ for all i, under mild assumptions on learning rate schedule μ and the sequence (X_t) (Assumptions 1, 2), the functions V_t in PID TD Learning (6) converge to the value function V^π of the policy π, almost surely.*

The proof of Theorem 1 uses the ordinary differential equations (ODE) method for convergence of stochastic approximation algorithms (Borkar and Meyn, 2000; Borkar, 2009). The method binds the behavior of the stochastic approximation to a limiting ODE. In our case, the ODE is

$$
\dot{u}(t) = L_g^\pi u(t) - u(t) = (A_g^\pi - I)u(t) + b_g^\pi.
$$

It is shown that if this ODE converges to the stationary point $\tilde{V}^\pi$, PID TD Learning will also converge. The condition for the convergence of this linear ODE is that the eigenvalues $\{\lambda'_i\}$ of $A_g^\pi - I$ should have negative real parts. Since $\lambda'_i = \lambda_i - 1$, we get the condition in Theorem 1. Note that this condition is weaker than $\rho(A_g^\pi) < 1$ for PID VI (Farahmand and Ghavamzadeh, 2021), which is equivalent to $|\lambda_i| < 1$. In other words, PID TD Learning may be convergent even if PID VI with the same controller gains g is not.

Obtaining similar results for PID Q-Learning is technically much more challenging. The reason for this difficulty is the fact that, just like PID VI for Control, as the agent's policy changes, the dynamics of PID Q-Learning changes. Similar to Farahmand and Ghavamzadeh (2021), we leave theoretical analysis of PID Q-Learning to future work and only focus on its empirical study.

4.2 Acceleration Result

In this section, we provide theoretical insights on how PID TD Learning can show a faster convergence compared to the conventional TD Learning. Our analysis relies on the finite-sample analysis of stochastic approximation methods. Since results for the asynchronous updates are limited, we provide our acceleration results for synchronous updates. Specifically, we provide our analysis for the case that at each iteration t, a dataset $\mathcal{D}_t = \{(x, A_{x,t}, R_{x,t}, X'_{x,t})\}_{x\in\mathcal{X}}$ is given, where for each state $x \in \mathcal{X}$ it contains the random action $A_{x,t} \sim \pi(\cdot|x)$, reward $R_{x,t} \sim \mathcal{R}(x, A_{x,t})$, and $X'_{x,t} \sim \mathcal{P}^\pi(\cdot|x, A_{x,t})$. Then, all values of V, V', and z are updated simultaneously in the same manner as (6). Similarly, synchronous TD Learning applies the conventional update on all states using the dataset. Based on the analysis by Chen et al. (2020), the following theorem provides bounds on the error of both algorithms for the learning rate schedule $\mu(t) = \epsilon/(t+T)$. We focus on the choices of ϵ, T that achieve the optimal asymptotic rate.

Theorem 2. *Suppose synchronous TD Learning and synchronous PID TD Learning are run with initial value function V_0 and learning rate $\mu(t) = \epsilon/(t+T)$ to evaluate policy π. Let V_t^{TD} and V_t^{PID} be the value functions obtained by the algorithms at iteration t, and $\{c_i^{\mathrm{TD}}, c_i^{\mathrm{PID}}\}$ be constants only dependent on the MDP and controller gains. Assume $\mathcal{P}^\pi$ is diagonalizable. If $\epsilon > 2/(1-\gamma)$ and $T \geq c_1^{\mathrm{TD}}\epsilon/(1-\gamma)$, we have*

$$\mathbb{E}\left[\left\|V_t^{\mathrm{TD}} - V^\pi\right\|_\infty^2\right] \leq c_2^{\mathrm{TD}} \left\|V_0 - V^\pi\right\|_\infty^2 \left(\frac{T}{t+T}\right)^{\epsilon(1-\gamma)} + \frac{\epsilon(c_3^{\mathrm{TD}} + c_4^{\mathrm{TD}} \left\|V^\pi\right\|_\infty^2)}{\epsilon(1-\gamma)-1}\left(\frac{\epsilon}{t+T}\right).$$

Moreover, assume we initialize $V' = V_0$ and $z = 0$ in PID TD Learning and A_g^π is diagonalizable with spectral radius $\rho < 1$. If $\epsilon > 2/(1-\rho)$ and $T \geq c_1^{\mathrm{PID}}\epsilon/(1-\rho)$, we have

$$\mathbb{E}\left[\left\|V_t^{\mathrm{PID}} - V^\pi\right\|_\infty^2\right] \leq c_2^{\mathrm{PID}} \left\|V_0 - V^\pi\right\|_\infty^2 \left(\frac{T}{t+T}\right)^{\epsilon(1-\rho)} + \frac{\epsilon(c_3^{\mathrm{PID}} + c_4^{\mathrm{PID}} \left\|V^\pi\right\|_\infty^2)}{\epsilon(1-\rho)-1}\left(\frac{\epsilon}{t+T}\right).$$

The assumption on diagonalizability of $\mathcal{P}^\pi$ and A_g^π in Theorem 2 is for the sake of simplicity. In Appendix B, we provide a similar but more general result without this assumption. The upper bounds in Theorem 2 consist of two terms. The first term, which scales with the initial error $\|V_0 - V^\pi\|_\infty$ can be interpreted as the *optimization error*. It is the amount that V_t still has to change to reach V^π. The second term can be considered as the *statistical error*, which is independent of the initial error and exists even if we start from $V_0 = V^\pi$. Due to the conditions $\epsilon > 2/(1-\gamma)$ and $\epsilon > 2/(1-\rho)$, the statistical error is asymptotically dominant with rate $\mathcal{O}(t^{-1})$ compared to $\mathcal{O}(t^{-\epsilon(1-\gamma)})$ or $\mathcal{O}(t^{-\epsilon(1-\gamma)})$ of the optimization error. Note that a larger ϵ accelerates the rate of optimization error, but together with larger T (due to the condition on T) slows the convergence of the statistical error to zero. For example, it takes T steps for the statistical error to become half of its initial value. For simplicity of discussion, we consider ϵ and T fixed.

The difference between the two algorithms is in the rate that the optimization error goes to zero. This term for TD Learning is $\mathcal{O}(t^{-\epsilon(1-\gamma)})$ and for PID TD Learning is $\mathcal{O}(t^{-\epsilon(1-\rho)})$. When $\kappa_p = 1$ and $\kappa_I = \kappa_d = \alpha = 0$, we have $\rho = \gamma$, and these two rates match. With a better choice of gains, one can have $\rho < \gamma$ (Farahmand and Ghavamzadeh, 2021) and achieve a faster rate for the optimization error. Even though this term is not asymptotically dominant, we show that its speed-up can be significant in the early stages of training, especially when the policy's behavior has low stochasticity. To show this, we first need to introduce the following definition.

Definition 1. *We say policy π in MDP $(\mathcal{X}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$ is d-deterministic for some $d \in [0,1]$ if for all $x \in \mathcal{X}$, we have $\mathrm{Var}[\mathcal{R}^\pi(x)] \leq (1-d)/4$ and $\max_{x'} \mathcal{P}^\pi(x'|x) \geq d$.*

Due to our assumption that rewards are bounded within [0,1], any policy in any MDP is 0-deterministic. The value of d depends on the stochasticity of both the MDP and the policy, with a larger value corresponding to a more deterministic behavior. In the case where the policy and the MDP are both deterministic, the policy becomes 1-deterministic. The following result shows how the initial optimization error compares to the statistical error based on this measure.

Proposition 1. *Assume the same conditions as in Theorem 2. Suppose policy π is d-deterministic in the environment. Let $E^{\mathrm{TD}}_{\mathrm{opt}}(t)$ and $E^{\mathrm{TD}}_{\mathrm{stat}}(t)$ be the first and the second terms in the bound for error of TD Learning at iteration t, respectively. Define $E^{\mathrm{PID}}_{\mathrm{opt}}(t)$ and $E^{\mathrm{PID}}_{\mathrm{stat}}(t)$ similarly. Define $c = \max((\kappa_p + \kappa_I\alpha)^2, \alpha^2)$. We have*

$$\frac{E^{\mathrm{TD}}_{\mathrm{opt}}(0)}{E^{\mathrm{TD}}_{\mathrm{stat}}(0)} \geq \frac{\|V_0 - V^\pi\|^2_\infty (5\gamma^2 n(1-d) + 2)}{en(1-d)\left(1 + 40\gamma^2 \|V^\pi\|^2_\infty\right)} \quad , \quad \frac{E^{\mathrm{PID}}_{\mathrm{opt}}(0)}{E^{\mathrm{PID}}_{\mathrm{stat}}(0)} \geq \frac{\|V_0 - V^\pi\|^2_\infty (15c\gamma^2 n(1-d) + 2)}{3ecn(1-d)\left(1 + 40\gamma^2 \|V^\pi\|^2_\infty\right)}.$$

Proposition 1 shows that when the initial error $\|V_0 - V^\pi\|_\infty$ is large or the policy behaves almost deterministically (d close to 1), the optimization error can make up the most of the error bound in Theorem 2. In that case, the acceleration achieved by PID TD Learning in this term becomes significant in the early stages. It should be noted that our arguments in this section are based on the upper bounds on the errors of the algorithms as opposed to the errors themselves. This is a common limitation for theoretical comparisons of algorithms. In Section 6, we further evaluate the convergence of the algorithms empirically.

5 Gain Adaptation

The proper choice of controller gains is critical to both convergence and acceleration of our proposed algorithms. While it is possible to treat the gains as hyperparameters and tune them like any other hyperparameter, we address this by designing an automatic gain adaptation algorithm that tunes them on the fly during the runtime of the algorithm.

The design of the gain adaptation algorithm for PID TD Learning and PID Q-Learning is based on the same idea as gain adaptation in PID VI. Translating the update rule (3) to the sample-based settings faces two main challenges. First, the derivative $\partial \mathrm{BR}^\pi V_{k+1}/\partial\kappa_\cdot$ and normalization factor $\|\mathrm{BR}^\pi V_k\|^2_2$ are not readily available without access to the transition dynamics $\mathcal{P}$. Second, computing the inner product in (3) requires iterating over all states x,

$$\left\langle \mathrm{BR}^\pi V_{k+1}, \frac{\partial \mathrm{BR}^\pi V_{k+1}}{\partial \kappa_\cdot} \right\rangle = \sum_x (\mathrm{BR}^\pi V_{k+1})(x) \cdot \frac{\partial (\mathrm{BR}^\pi V_{k+1})(x)}{\partial \kappa_\cdot},$$

requiring the values of $\partial(\mathrm{BR}^\pi V_{k+1})(x)/\partial\kappa_\cdot$ and $(\mathrm{BR}^\pi V_{k+1})(x)$ for every x. We will see that using a sample (X_t, A_t, R_t, X'_t), these values can be estimated for $x = X_t$ but not for other states. A replay buffer could give us access to samples at more states or function approximation could directly provide estimates at all states. However, as these techniques suffer from memory and stability issues, a better solution is needed for this challenge.

To avoid the difficulty of the inner product term, we modify the update rule of the gains at iteration t to minimize $(\mathrm{BR}^\pi V_{t+1})(X_t)^2$ instead of $\|\mathrm{BR}^\pi V_{t+1}\|^2_2$. This modification is similar to performing stochastic gradient descent instead of gradient descent. Instead of defining the loss over the whole state space, we consider the loss on a single sampled state. Consequently, the new term only depends on the values at X_t. We get the following update:

$$\begin{aligned}
\kappa_\cdot &\leftarrow \kappa_\cdot - \eta \frac{2}{\|\mathrm{BR}^\pi V_t\|^2_2} \cdot \frac{\partial \frac{1}{2} (\mathrm{BR}^\pi V_{t+1})(X_t)^2}{\partial \kappa_\cdot} \\
&= \kappa_\cdot - \eta \frac{1}{\|\mathrm{BR}^\pi V_t\|^2_2} \cdot (\mathrm{BR}^\pi V_{t+1})(X_t) \cdot \frac{\partial (\mathrm{BR}^\pi V_{t+1})(X_t)}{\partial \kappa_\cdot}.
\end{aligned}$$

The term $(\mathrm{BR}^\pi V_{t+1})(X_t)$ above can be estimated in a similar manner to (5). Estimating the derivative $\partial(\mathrm{BR}^\pi V_{t+1})(X_t)/\partial\kappa_\cdot$ as well as $(\mathrm{BR}^\pi V_{t+1})(X_t)$ in an unbiased way is another challenging problem. It is known that forming an unbiased estimate for both of these quantities with only one sample at state X_t leads to double-sampling issues (Baird, 1995). As in prior work (Kearney et al., 2018), we use the semi-gradient trick for this problem. Specifically, we treat the $T^\pi V$ term in $\mathrm{BR}^\pi V$ as constant and ignore its derivative. This yields the estimate

$$\frac{\partial(\mathrm{BR}^\pi V_{t+1})(X_t)}{\partial\kappa_\cdot} = \frac{\partial}{\partial\kappa_\cdot}\left[r^\pi(X_t) + \gamma\sum_{x'}\mathcal{P}^\pi(x'|X_t)V_{t+1}(x') - V_{t+1}(X_t)\right] \approx -\frac{\partial V_{t+1}(X_t)}{\partial\kappa_\cdot}.$$

When calculating $\frac{\partial V_{t+1}(X_t)}{\partial\kappa_\cdot}$, we further ignore the effect of gains on V_t, setting $\frac{\partial V_t}{\partial\kappa_\cdot} \approx 0$, and also drop the learning rate $\mu(N_t(X_t))$ to absorb it into η. These derivatives can be calculated based on (6) and are given in Appendix C. Finally, the normalization term is estimated by keeping an exponential moving average with smoothing factor λ of the square of estimates of the Bellman Residual in the past iterations. The detailed version of PID TD Learning and PID Q-Learning with gain adaptaion is shown in Algorithms 1 and 2 in Appendix C.

6 Empirical Results

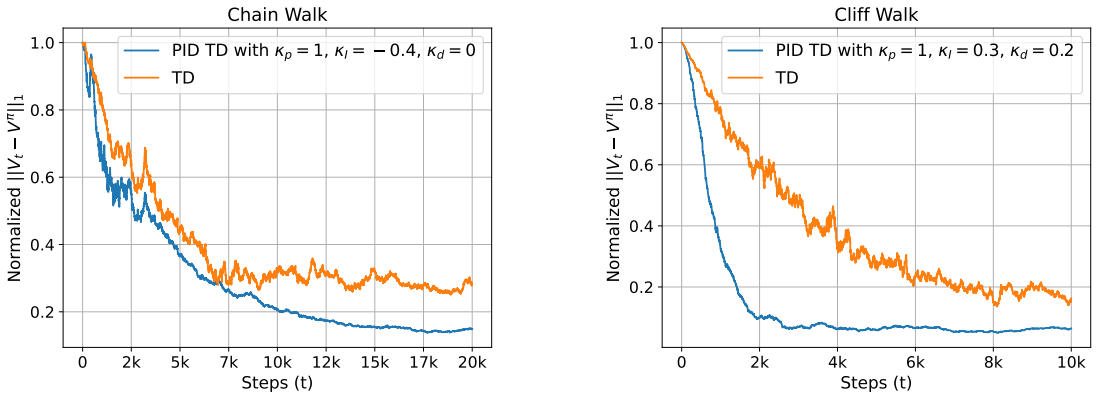

Figure 1: Comparison of PID TD Learning with Conventional TD Learning in Chain Walk (left) and Cliff Walk (right) with $\gamma = 0.99$. Each curve is averaged over 80 runs. Shaded areas show the standard error.

We empirically compare PID TD Learning and PID Q-Learning with their conventional counterparts. We conduct experiments in the 50-state Chain Walk environment with 2 actions (Farahmand and Ghavamzadeh, 2021), the Cliff Walk environment with 6×6 states and 4 actions (Rakhsha et al., 2022), and randomly generated Garnet MDPs with 50 states and 3 actions (Bhatnagar et al., 2009). Detailed descriptions of these environments and the policies evaluated can be found in Appendix D. For each sample (X_t, A_t, R_t, X'_t), we choose X_t uniformly at random and A_t is chosen according to π (for PE) or at random (for Control). We measure the error of value functions V_t and Q_t for PE and Control problems by their normalized error defined as $\|V_t - V^\pi\|_1 / \|V^\pi\|_1$ and $\|Q_t - Q^*\|_F / \|Q^*\|_F$, respectively, where $\|Q\|_F \triangleq (\sum_{x,a} Q(x,a)^2)^{\frac{1}{2}}$.

For all learning rates, we use state-count dependent schedules of the form $\mu(N_t(X_t)) = \min(\epsilon, N_t(X_t)/M)$ for some choice of ϵ and M for all algorithms (including PID Q-Learning and Q-Learning). To achieve the best results for all algorithms, we use separate learning rates for V, V', z components of PID TD Learning and Q, Q', z components of PID Q-Learning. The hyperparameters ϵ, M of all learning rate schedules are tuned by gridsearch over a range of values. The details of hyperparameter tuning are provided in Appendix F.

In Figure 1, we compare PID TD Learning with TD Learning when the gains are fixed and $\gamma = 0.99$. In this case the acceleration depends on the choice of gains and the environment. We observe that we can achieve a drastic acceleration in Cliff Walk, and a minor acceleration in Chain Walk. We further investigate the speed-up achieved by PID TD Learning in Cliff Walk. In Figure 2, we observe that with Gain Adaptation and $\gamma = 0.999$, we achieve a significant acceleration without the need to tune the controller gains. Figure 2 also shows how Gain Adaptation has modified the gains from their initial values.

To evaluate the acceleration in the Control problem, we compare PID Q-Learning with Gain Adaptation with Q-Learning. Figure 3 shows this comparison in Chain Walk with $\gamma = 0.999$, where PID Q-Learning shows acceleration. Finally, to draw a more conclusive comparison, we compare our algorithms with the conventional ones on 80 randomly generated Garnet MDPs with $\gamma = 0.99$ in Figure 4. We see that our algorithms outperform TD Learning and Q-Learning in both PE and Control problems.

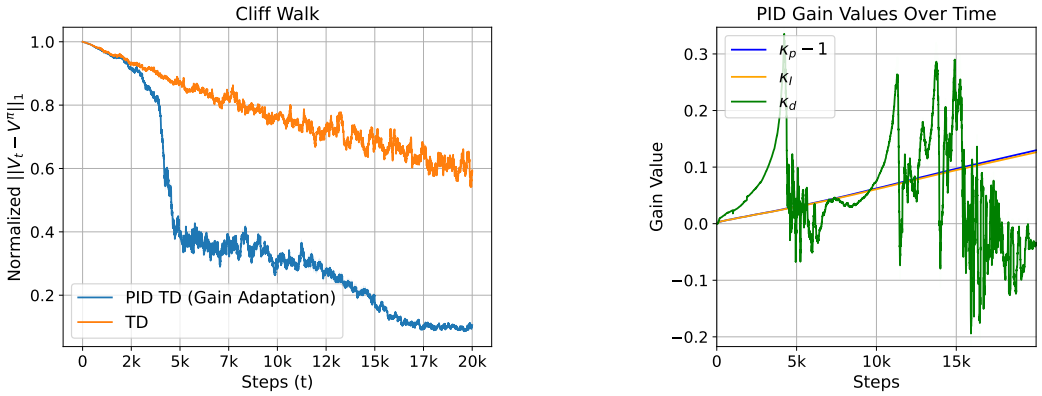

Figure 2: PID TD Learning with Gain Adaptation in Cliff Walk with $\gamma = 0.999$. *(Left)* Comparison of value errors of PID TD Learning with TD Learning. Each curve is averaged over 80 runs. Shaded area shows standard error. *(Right)* The change of gains done by Gain Adaptation through training.

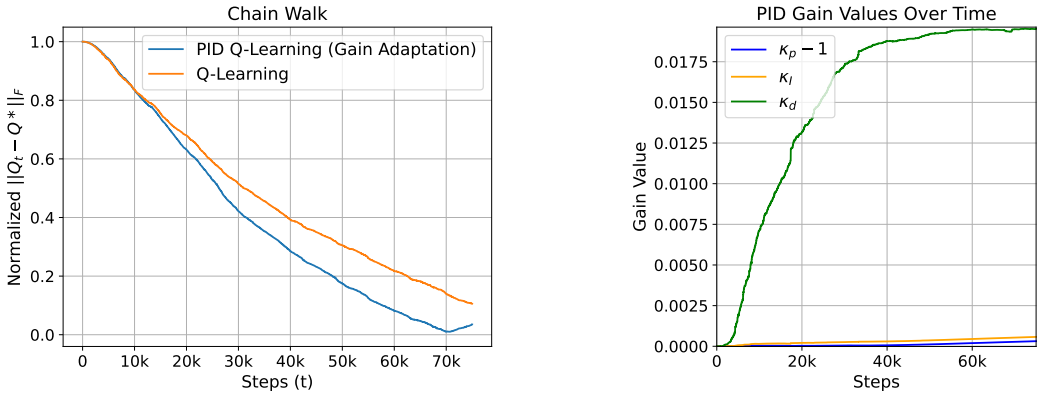

Figure 3: PID Q-Learning with Gain Adaptation in Chain Walk with $\gamma = 0.999$. *(Left)* Comparison of value errors of PID Q-Learning with Q-Learning. Each curve is averaged over 80 runs. Shaded area shows standard error. *(Right)* The change of gains done by Gain Adaptation through training.

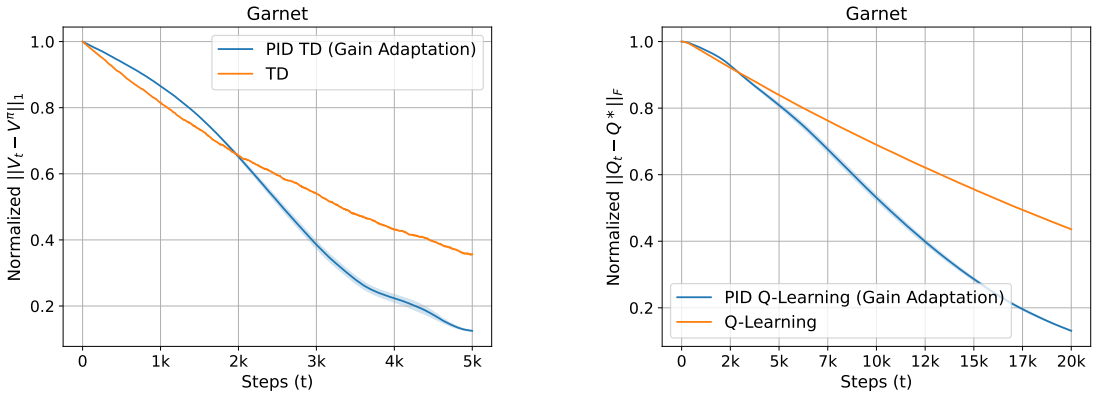

Figure 4: Comparison of PID Accelerated algorithms with the conventional ones for PE (Left) and Control (Right) problems in randomly generated Garnet environments with $\gamma = 0.99$. Each curve is an average of 80 MDPs, run for 80 times each. Shaded area shows standard error.

7 Related Work

There is a growing literature of applying acceleration techniques to RL. Similar to PID VI, which as we showed leads to PID TD Learning and PID Q-Learning, many accelerated dynamic programming methods have closely related RL algorithms. The idea of momentum has been used for faster dynamic programming algorithms by Vieillard et al. (2020); Goyal and Grand-Clément (2023) and in RL setting has led to Speedy Q-Learning (Ghavamzadeh et al., 2011) and Momentum Q-Learning (Bowen et al., 2021). Zap Q-Learning (Devraj and Meyn, 2017) is another accelerated variant of Q-Learning based on second-order optimization methods. Anderson acceleration (Anderson, 1965) has been used for Anderson VI (Geist and Scherrer, 2018) and Anchoring acceleration (Halpern, 1967) is used in Anchord VI (Lee and Ryu, 2023). Matrix splitting is used to derive Operator Splitting VI (OSVI) (Rakhsha et al., 2022) and Deflated Dynamics VI (DDVI) (Lee et al., 2024), which are both extended to the RL setting through stochastic approximation. Recently, Rakhsha et al. (2024) has introduced the Model Correcting VI (MoCoVI) and Model Correcting Dyna (MoCoDyna) algorithms that achieve acceleration through model correction.

Gain adaptation in general has a long history in RL and closely-related literature. Kesten (1958) used an adaptive mechanism in the context of stochastic approximation in the 1950s. They describe a method for choosing the learning rate of SA that is very similar to the P component of the gain adaptation procedure we naturally derive. However, the algorithm is ad hoc in nature, and is not compatible with function approximation in any natural way. First order methods of adapting hyperparameters have been proposed, including IDBD (Sutton, 1992), the recent RL focused variant TIDBD (Kearney et al., 2018), and SMD (Schraudolph, 1999) which all tune learning rates by finding the gradient with respect to the history of errors. We refer to Sutton (2022) for a more in-depth history and overview of such techniques. These approaches are limited to controlling only the learning rate of the procedure and thus only attacking the error from sampling, not the bootstrapping error.

8 Conclusion

We showed how recent advances in accelerated planning and dynamic programming, specifically the PID Value Iteration algorithm, can be used to design algorithms for the RL setting. The proposed PID TD Learning and PID Q-Learning algorithms are accompanied by a gain adaptation mechanism, which tunes their hyperparameters on the fly. We provided theoretical analysis as well as empirical studies of these algorithms.

One limitation of the current work is that the proposed algorithms are only developed for finite MDPs where the value function, and all relevant quantities, can be represented exactly. For large MDPs, for example with continuous state spaces, we need to use function approximation. Developing PID TD Learning and PID Q-Learning with function approximation is therefore one important future direction. Another limitation of this work is that the gain adaptation procedure, even though empirically reliable, does not come with a convergence guarantee. Moreover, small changes in its hyperparameters, such as its meta-learning rate η, can cause large changes in the trajectory the value function takes during training. Another interesting research direction is then to develop a gain adaptation procedure that is less sensitive to the choice of hyperparameters and has a convergence guarantee. Finally, this work shows that the dynamics of RL can be significantly influenced by the PID controller, one of the simplest controllers in the arsenal of control engineering. Developing Planning and RL algorithms based on more sophisticated controllers is another promising research direction.

Acknowledgements and Disclosure of Funding

We would like to thank the other members of the Adaptive Agents (Adage) Lab who provided feedback on a draft of this paper, and the anonymous reviewers whose comments helped us improve the clarity of the paper. AMF acknowledges the funding from the Canada CIFAR AI Chairs program, as well as the support of the Natural Sciences and Engineering Research Council of Canada (NSERC) through the Discovery Grant program (2021-03701). Resources used in preparing this research were provided, in part, by the Province of Ontario, the Government of Canada through CIFAR, and companies sponsoring the Vector Institute.

References

Donald G. Anderson. Iterative procedures for nonlinear integral equations. *J. ACM*, 12(4):547–560, oct 1965. ISSN 0004-5411. doi: 10.1145/321296.321305. 10

Leemon Baird. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings 1995*, pages 30–37. Morgan Kaufmann, 1995. ISBN 978-1-55860-377-6. 8

Amir Beck. *First-Order Methods in Optimization*. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, 2017. ISBN 9781611974997. 17

Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996. ISBN 9781886529106. 2

Shalabh Bhatnagar, Richard S Sutton, Mohammad Ghavamzadeh, and Mark Lee. Natural actor-critic algorithms. *Automatica*, 45(11):2471–2482, 2009. 8

Joseph K. Blitzstein and Jessica Hwang. *Introduction to Probability*. Chapman & Hall/CRC Texts in Statistical Science. CRC Press/Taylor & Francis Group, 2014. ISBN 9781466575578. 15

Vivek S. Borkar. *Stochastic Approximation: A Dynamical Systems Viewpoint*. Texts and Readings in Mathematics. Hindustan Book Agency, 2009. ISBN 9789386279385. 5

Vivek S. Borkar and Sean P. Meyn. The ODE method for convergence of stochastic approximation and reinforcement learning. *SIAM Journal on Control and Optimization*, 38(2):447–469, 2000. 5, 15, 16

Weng Bowen, Xiong Huaqing, Zhao Lin, Liang Yingbin, and Zhang Wei. Finite-time theory for momentum q-learning. In *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*. PMLR, 2021. 10

Jinglin Chen and Nan Jiang. Information-theoretic considerations in batch reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019. 1

Zaiwei Chen, Siva Theja Maguluri, Sanjay Shakkottai, and Karthikeyan Shanmugam. Finite-sample analysis of stochastic approximation using smooth convex envelopes. *arXiv preprint arXiv:2002.00874*, 2020. 6, 18, 19

Adithya M. Devraj and Sean Meyn. Zap q-learning. *Advances in Neural Information Processing Systems*, 30, 2017. 10

Richard C. Dorf and Robert H. Bishop. *Modern Control Systems*. Prentice Hall, 2008. ISBN 9780132270281. 3

Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research (JMLR)*, 6:503–556, 2005. 1

Eyal Even-Dar and Yishay Mansour. Learning rates for Q-learning. *Journal of Machine Learning Research (JMLR)*, 5:1–25, 2003. 1

Jianqing Fan, Zhaoran Wang, Yuchen Xie, and Zhuoran Yang. A theoretical analysis of deep q-learning. *arXiv:1901.00137v3*, 2019. 1

Amir-massoud Farahmand and Mohammad Ghavamzadeh. PID accelerated value iteration algorithm. In *International Conference on Machine Learning*. PMLR, 2021. 1, 2, 3, 5, 6, 8

Amir-massoud Farahmand, Csaba Szepesvári, and Rémi Munos. Error propagation for approximate policy and value iteration. *Advances in Neural Information Processing Systems*, 23, 2010. 1

M. Geist and B. Scherrer. Anderson acceleration for reinforcement learning. *European Workshop on Reinforcement Learning*, 2018. 10

Mohammad Ghavamzadeh, Hilbert Kappen, Mohammad Azar, and Rémi Munos. Speedy q-learning. In *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. 10

Geoffrey Gordon. Stable function approximation in dynamic programming. In *International Conference on Machine Learning (ICML)*, 1995. 1

Vineet Goyal and Julien Grand-Clément. A first-order approach to accelerated value iteration. *Operations Research*, 71(2):517–535, 2023. 10

Benjamin Halpern. Fixed points of nonexpanding maps. *Bulletin of the American Mathematical Society*, 73(6):957–961, 1967. 10

Alston S. Householder. The approximate solution of matrix problems. *Journal of the ACM (JACM)*, 5(3):205–243, 1958. 16

Alex Kearney, Vivek Veeriah, Jaden B. Travnik, Richard S. Sutton, and Patrick M. Pilarski. Tidbd: Adapting temporal-difference step-sizes through stochastic meta-descent. *arXiv preprint arXiv:1804.03334*, 2018. 8, 10

Harry Kesten. Accelerated stochastic approximation. *The Annals of Mathematical Statistics*, pages 41–59, 1958. 10

Jongmin Lee and Ernest K. Ryu. Accelerating value iteration with anchoring. *Neural Information Processing Systems*, 2023. 10

Jongmin Lee, Amin Rakhsha, Ernest K Ryu, and Amir-massoud Farahmand. Deflated dynamics value iteration. *arXiv preprint arXiv:2407.10454*, 2024. 10

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015. 1

Rémi Munos and Csaba Szepesvári. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research (JMLR)*, 9:815–857, 2008. 1

Katsuhiko Ogata. *Modern Control Engineering*. Prentice hall Upper Saddle River, NJ, fifth edition, 2010. 3

Amin Rakhsha, Andrew Wang, Mohammad Ghavamzadeh, and Amir-massoud Farahmand. Operator splitting value iteration. *Advances in Neural Information Processing Systems*, 35, 2022. 8, 10, 22

Amin Rakhsha, Mete Kemertas, Mohammad Ghavamzadeh, and Amir massoud Farahmand. Maximum entropy model correction in reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2024. 10

Nicol N. Schraudolph. Local gain adaptation in stochastic gradient descent. In *1999 Ninth International Conference on Artificial Neural Networks ICANN 99.(Conf. Publ. No. 470)*, volume 2, pages 569–574. IET, 1999. 10

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

Richard S. Sutton. Adapting bias by gradient descent: an incremental version of delta-bar-delta. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, AAAI'92, page 171–176. AAAI Press, 1992. ISBN 0262510634. 10

Richard S. Sutton. A history of meta-gradient: Gradient methods for meta-learning. *arXiv preprint arXiv:2202.09701*, 2022. 10

Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018. 2

Csaba Szepesvári. The asymptotic convergence-rate of Q-learning. In *Advances in Neural Information Processing Systems*, 1997. 1

Csaba Szepesvári. *Algorithms for Reinforcement Learning*. Morgan Claypool Publishers, 2010. 2

Gerald Teschl. *Ordinary differential equations and dynamical systems*, volume 140. American Mathematical Soc., 2012. 16

Samuele Tosatto, Matteo Pirotta, Carlo D'Eramo, and Marcello Restelli. Boosted fitted Q-iteration. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017. 1

John N. Tsitsiklis and Benjamin Van Roy. An analysis of temporal difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42:674–690, 1997. 1

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double Q-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016. 1

Nino Vieillard, Bruno Scherrer, Olivier Pietquin, and Matthieu Geist. Momentum in reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2529–2538. PMLR, 2020. 10

Martin J. Wainwright. Stochastic approximation with cone-contractive operators: Sharp ℓ_∞-bounds for q-learning. *arXiv preprint arXiv:1905.06265*, 2019. 1

Christopher J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, University of Cambride, 1989. 1

A Proofs for Convergence Results (Section 4.1)

In this section, we present the proof of Theorem 1. We first present some notation, and then the assumptions on the learning rate schedule μ and sequence of visited samples $(X_t)_{t\geq 0}$. Let $r^\pi \in \mathbb{R}^n$ be the vector of the expected immediate rewards of following the policy at each state.

Now we move on the assumptions for Theorem 1.

Assumption 1 (Properly Tapering Learning Rate Schedule)**.** *The learning rate schedule $\mu\colon \mathbb{Z} \to \mathbb{R}^+$ satisfies the following:*

(i) We have $0 < \mu(t) \leq 1$ for any $t \geq 0$, and

$$\sum_{t=0}^{\infty} \mu(t) = \infty \qquad , \qquad \sum_{t=0}^{\infty} \mu(t)^2 < \infty.$$

(ii) For some T, we have $\mu(t+1) < \mu(t)$ for all $t \geq T$.

(iii) For $z \in (0,1)$, $\sup_t \mu([zt])/\mu(t) < \infty$, where $[\cdot]$ is the integer part of a number.

(iv) For $z \in (0,1)$,

$$\lim_{t\to\infty} \left(\sum_{i=0}^{[zt]} \mu(i)\right) \Bigg/ \left(\sum_{i=0}^{t} \mu(i)\right) = 1.$$

Examples of learning rate schedules that satisfy Assumption 1 includes $\mu(t) = \frac{1}{t+1}$. The next assumption is on the balanced updates of states.

Assumption 2 (Balanced Updates of States)**.** *The sequence of visited states $(X_t)_t$ and learning rate schedule μ is such that we have*

(i) There exists deterministic $\Delta > 0$, such that for all $x \in \mathcal{X}$

$$\liminf_{t\to\infty} \frac{N_t(x)}{t} \geq \Delta \qquad \text{a.s.}$$

(ii) If $T_t(z) \triangleq \min\{t' > t\colon \sum_{i=t+1}^{t'} \mu(i) > z\}$, for any $z > 0$ and states $x_1, x_2 \in \mathcal{X}$, the following limit exists

$$\lim_{t\to\infty} \frac{\sum_{i=N_t(x_1)}^{N_{T_t(z)}(x_1)} \mu(i)}{\sum_{i=N_t(x_2)}^{N_{T_t(z)}(x_2)} \mu(i)}.$$

Intuitively, Assumption 2 asserts that all states are visited often enough and get balanced sum of learning rates. Before presenting the proof for Theorem 1, we first prove the following auxiliary lemma.

Lemma 1. *Assume policy π in the environment is d-deterministic and $x \in \mathcal{X}$ is arbitrary. Let R and X' be the random obtained reward and next state after following policy π from x in the environment. Let $W = R + \gamma V(X') - (T^\pi V)(x)$ for an arbitrary $V\colon \mathcal{X} \to \mathbb{R}$. We have*

$$\mathbb{E}\left[W^2\right] \leq \frac{1-d}{4} + 5\gamma^2(1-d)\left\|V\right\|_\infty^2.$$

Moreover, for some $\tilde{V}, \boldsymbol{f}$, let $\hat{L}$ be the estimator of $(L_g^\pi \tilde{V})(x, \boldsymbol{f})$ derived in PID TD Learning's update (4) according to the sample (x, R, X'). Assume $\tilde{W} = \hat{L} - (L_g^\pi \tilde{V})(x, \boldsymbol{f})$ is its noise. We have

$$\mathbb{E}\left[\tilde{W}^2\right] \leq \max((\kappa_p + \kappa_I\alpha)^2, \alpha^2)\left(\frac{1-d}{4} + 5\gamma^2(1-d)\left\|\tilde{V}\right\|_\infty^2\right).$$

Proof. For the first part, we write

$$\begin{aligned}\mathbb{E}[W^2] &= \mathbb{E}\left[\left(R+\gamma V(X') - r^\pi(x) - \gamma\mathbb{E}[V(X')]\right)^2\right] \\ &= \mathbb{E}\left[(R - r^\pi(X))^2\right] + \gamma^2\mathbb{E}\left[\left(V(X') - \mathbb{E}[V(X')]\right)^2\right] + 2\mathbb{E}[(R - r^\pi(x))(V(X') - \mathbb{E}[V(X')])] \\ &= \mathrm{Var}[R] + \gamma^2\mathrm{Var}[V(X')] \\ &\leq \frac{1-d}{4} + \gamma^2\mathrm{Var}[V(X')],\end{aligned}$$

where the last inequality is by the definition of d-deterministic MDP. Now let $p^* = \max_{x'} \mathcal{P}^\pi(x'|x)$ and $x^* = \arg\max_{x'} \mathcal{P}^\pi(x'|x)$. Due to the law of total variance (Blitzstein and Hwang, 2014, Example 9.5.5), we have

$$\begin{aligned}\mathrm{Var}[V(X')] &= p^*\mathrm{Var}[V(X')|X' = x^*] + (1-p^*)\mathrm{Var}[V(X')|X' \neq x^*] \\ &\qquad + p^*(1-p^*)\Big(\mathbb{E}[V(X')|X' = x*] - \mathbb{E}[V(X')|X' \neq x*]\Big)^2 \\ &\leq (1-p^*)\left\|V\right\|_\infty^2 + 4p^*(1-p^*)\left\|V\right\|_\infty^2 \\ &\leq 5(1-d)\left\|V\right\|_\infty^2.\end{aligned}$$

Together, we obtain

$$\mathbb{E}[W^2] \leq \frac{1-d}{4} + 5(1-d)\left\|V\right\|_\infty^2.$$

For the second part, we consider three cases. If $\mathtt{f} = \mathtt{v}$, we have

$$\tilde{W} = (\kappa_p + \kappa_I\alpha)W.$$

If $\mathtt{f} = \mathtt{z}$, we have

$$\tilde{W} = \alpha W.$$

If $\mathtt{f} = \mathtt{v}'$, we have

$$\tilde{W} = 0.$$

Combining all cases, we get

$$\tilde{W}^2 \leq \max((\kappa_p + \kappa_I\alpha)^2, \alpha^2)W^2,$$

which means

$$\mathbb{E}\left[\tilde{W}^2\right] \leq \max((\kappa_p + \kappa_I\alpha)^2, \alpha^2)\left(\frac{1-d}{4} + 5\gamma^2(1-d)\left\|\tilde{V}\right\|_\infty^2\right).$$

□

Proof of Theorem 1

Proof. We show the claim by applying the result by Borkar and Meyn (2000, Theorem 2.5) to our algorithm. We describe how PID TD Learning (6) is a special case of a convergent asynchronous stochastic approximation in (Borkar and Meyn, 2000). PID TD Learning updates three entries of $\tilde{V}_t$ at each iteration. Therefore, our set of indices that are updated (noted by $Y(n)$ in the original paper) is $Y_t \triangleq \{(X_t, \mathtt{v}), (X_t, \mathtt{z}), (X_t, \mathtt{v'})\}$. The number of times the value for $(x, \mathtt{f}) \in \mathcal{X} \times \{\mathtt{v}, \mathtt{z}, \mathtt{v'}\}$ is updated (noted by $\nu(i, n)$ in the original paper) is $N_t(x)$ and the communication delays are zero in our case. With these choices, the asynchronous stochastic approximation of (Borkar and Meyn, 2000, Equation 2.8) becomes

$$\tilde{V}_{t+1}(X_t, \mathtt{f}) \leftarrow \tilde{V}_t(X_t, \mathtt{f}) + \mu(N_t(X_t))f[\tilde{V}_t, D_t](X_t, \mathtt{f}) \quad (\forall \mathtt{f} \in \{\mathtt{v}, \mathtt{z}, \mathtt{v'}\}),$$

and the other entries in $\tilde{V}_{t+1}$ remain the same as $\tilde{V}_t$. Here, $D_t \in \mathcal{D}$ for all t are independently and identically distributed (i.i.d.), and $f\colon \mathbb{R}^{3n} \times \mathcal{D} \to \mathbb{R}^{3n}$ can be an arbitrary function. To obtain PID TD Learning in this form, we define $\mathcal{D} \triangleq (\mathcal{X} \times [0,1])^n$. For any t, we choose D_t to be a dataset $\{(R_{x,t}, X'_{x,t})\}_{x\in\mathcal{X}}$ where for each state x contains the random reward $R_{x,t} \sim \mathcal{R}^\pi(x)$ and $X'_{x,t} \sim \mathcal{P}^\pi(\cdot|x)$. Then we define f such that for all $x \in \mathcal{X}$,

$$
\begin{aligned}
f[\tilde{V}_t, D_t](x, \mathtt{v}) &\triangleq \kappa_p b_{x,t} + \kappa_I(\beta z_t(x) + \alpha b_{x,t}) + \kappa_d(V_t(x) - V'_t(x)),\\
f[\tilde{V}_t, D_t](x, \mathtt{z}) &\triangleq \beta z_t(x) + \alpha b_{x,t} - z_t(x),\\
f[\tilde{V}_t, D_t](x, \mathtt{v'}) &\triangleq V_t(x) - V'_t(x).
\end{aligned}
$$

where $b_{x,t} = R_{x,t} + \gamma V_t(X'_{x,t}) - V_t(x)$. This yields the exact same PID TD Learning updates.

The function $h(\tilde{V}) = \mathbb{E}[f(\tilde{V}, D_1)]$ in (Borkar and Meyn, 2000) is equal to $L_g^\pi \tilde{V} - \tilde{V}$ in our setting. Note that h is Lipschitz since it is an affine linear operator. The function $h_\infty(\tilde{V})$ exists and is equal to $(A_g^\pi - I)\tilde{V}$. Therefore, we require the origin point to be an asymptotically stable equilibrium of the ODE

$$\dot{u}(t) = h_\infty(u(t)) = (A_g^\pi - I)u(t),$$

which is satisfied due to the assumption on the eigenvalues of A_g^π and the fact that the solution of the above ODE is $\exp[(A_g^\pi - I)t]u_0$ (Teschl, 2012) for any starting point u_0.

Furthermore, we note that the unique globally asymptotically stable equilibrium of ODE

$$\dot{u}(t) = h(u(t)) = L_g^\pi u(t) - u(t) = (A_g^\pi - I)u(t) + b_g^\pi.$$

is $-(A_g^\pi - I)^{-1} b_g^\pi$ which is equal to $\tilde{V}^\pi$ due to $\tilde{V}^\pi = L_g^\pi \tilde{V}^\pi = A_g^\pi \tilde{V}^\pi + b_g^\pi$.

Finally, note that Lemma 1 established the required property of the noise. The remaining assumptions of Borkar and Meyn (2000) are satisfied due to Assumption 1 and 2.

□

B Proofs for Acceleration Results (Section 4.2)

Before presenting the proof of the Theorems, we introduce these definitions.

Definition 2. *Let $f\colon \mathbb{R}^d$ be a convex, differentiable function. Then f is said to be L-smooth with respect to (w.r.t.) norm $\|\cdot\|$ if and only if*

$$f(y) \le f(x) + \langle \nabla f(x), y - x\rangle + \frac{L}{2}\|y - x\|^2 \qquad \forall x, y \in \mathbb{R}^d.$$

Definition 3. *For any non-singular matrix $S \in \mathbb{R}^{d\times d}$, we define the vector norm $\|v\|_{2,S} \triangleq \|Sv\|_2$. For any matrix A, we let $\|A\|_{2,S}$ be the matrix norm of A induced by the vector norm $\|\cdot\|_{2,S}$.*

We also need the following lemmas.

Lemma 2. *Let $A \in \mathbb{R}^{d\times d}$ and $\delta \ge 0$. If A is not diagonalizable, further assume $\delta > 0$. There exists an invertible matrix S such that $\|A\|_{2,S} \le \rho(A) + \delta$ and for any $v \in \mathbb{R}^d$, $\|v\|_{2,S} \le \|v\|_\infty$.*

Proof. The existence of S' such that $\|A\|_{2,S'} \le \rho(A) + \delta$ is a consequence of the proof of Theorem 4.4 in Householder (1958). Due to equivalence of norms in finite dimensions, there exists $u > 0$ such that for any $v \in \mathbb{R}^d$, $\|v\|_{2,S'} \le u\|v\|_\infty$. Define $S = \frac{1}{u}S'$. Consequently, $\|v\|_{2,S} \le \|v\|_\infty$ for any v. Now from Theorem 2.10 in Householder (1958), we have $\|A\|_{2,S'} = \left\|S'AS'^{-1}\right\|_2$ and $\|A\|_{2,S} = \left\|SAS^{-1}\right\|_2$ which means

$$\|A\|_{2,S} = \left\|SAS^{-1}\right\|_2 = \left\|S'AS'^{-1}\right\|_2 = \|A\|_{2,S'} \le \rho(A) + \delta.$$

□

Lemma 3. *For any invertible matrix S, the function $f\colon \mathbb{R}^d \to \mathbb{R}$ defined as $f(x) = \frac{1}{2}\|x\|_{2,S}^2$ is 1-smooth w.r.t. $\|\cdot\|_{2,S}$.*

Proof. Let $g(x) = \frac{1}{2}\|x\|_2^2$. By definition, $f(x) = g(Sx)$. We write

$$\begin{aligned} f(x) + \langle \nabla f(x), y - x\rangle + \frac{1}{2}\|y-x\|_{2,S}^2 &= g(Sx) + \left\langle S^\top \nabla g(Sx), y - x\right\rangle + \frac{1}{2}\|Sy - Sx\|_2^2 \\ &= g(Sx) + \langle \nabla g(Sx), Sy - Sx\rangle + \frac{1}{2}\|Sy - Sx\|_2^2 . \end{aligned}$$

By Beck (2017, Example 5.11), g is 1-smooth, that is for any u, v,

$$g(u) + \langle \nabla g(u), v - u\rangle + \frac{1}{2}\|v-u\|_2^2 \geq g(v).$$

Together, we conclude

$$f(x) + \langle \nabla f(x), y - x\rangle + \frac{1}{2}\|y-x\|_{2,S}^2 \geq g(Sy) = f(y).$$

□

Lemma 4. *Assume a dataset $\{(x, R_x, X'_x)\}_x$ is given, where for each state x contains the random reward $R_x \sim \mathcal{R}^\pi(x)$ and $X'_x \sim \mathcal{P}^\pi(\cdot|x)$, and π is d-deterministic in the environment. For an arbitrary value function V, define $W\colon \mathcal{X} \to \mathbb{R}$ as*

$$W(x) \triangleq R_x + \gamma V(X'_x) - r^\pi(x) - \gamma(\mathcal{P}^\pi V)(x).$$

Moreover, for some $\tilde{V}$ and $x, \boldsymbol{f}$, let $\hat{L}(x, \boldsymbol{f})$ be the estimator of $(L_g^\pi \tilde{V})(x, \boldsymbol{f})$ derived in PID TD Learning's update (4) according to the sample (x, R_x, X'_x). Define

$$\tilde{W}(x, \boldsymbol{f}) = \hat{L}(x, \boldsymbol{f}) - (L_g^\pi \tilde{V})(x, \boldsymbol{f}).$$

We have

$$\begin{aligned} \mathbb{E}\left[\|W\|_\infty^2\right] &\leq n\left(\frac{1-d}{4} + 5\gamma^2(1-d)\|V\|_\infty^2\right), \\ \mathbb{E}\left[\left\|\tilde{W}\right\|_\infty^2\right] &\leq 3n\max((\kappa_p + \kappa_I\alpha)^2, \alpha^2)\left(\frac{1-d}{4} + 5\gamma^2(1-d)\left\|\tilde{V}\right\|_\infty^2\right). \end{aligned}$$

Proof. According to Lemma 1, for any $x, \mathtt{f}$,

$$\begin{aligned} \mathbb{E}\left[W(x)^2\right] &\leq \frac{1-d}{4} + 5\gamma^2(1-d)\|V\|_\infty^2, \\ \mathbb{E}\left[\tilde{W}(x, \mathtt{f})^2\right] &\leq \max((\kappa_p + \kappa_I\alpha)^2, \alpha^2)\left(\frac{1-d}{4} + 5\gamma^2(1-d)\left\|\tilde{V}\right\|_\infty^2\right). \end{aligned}$$

The result follows from the fact that for any random vector $Z = [Z_1, \ldots, Z_k]^\top$,

$$\mathbb{E}\left[\|Z\|_\infty^2\right] \leq \mathbb{E}\left[\sum_i Z_i^2\right] = \sum_i E\left[Z_i^2\right].$$

□

B.1 Proof of Theorem 2

We prove the more general result than Theorem 2 without any diagonalizablity assumptions. Theorem 2 is the special case of the following when $\delta^{\mathrm{TD}} = \delta^{\mathrm{PID}} = 0$.

Theorem 3. *Suppose synchronous TD Learning and PID TD Learning are run with initial value function V_0 and learning rate $\mu(t) = \epsilon/(t+T)$ to evaluate policy π. Let $V_t^{\mathrm{TD}}, V_t^{\mathrm{PID}}$ be the value functions obtained with each algorithm at iteration t. Assume $\delta^{\mathrm{TD}}, \delta^{\mathrm{PID}} \geq 0$. If $\mathcal{P}^\pi$ is not diagonalizable, we further assume $\delta^{\mathrm{TD}} > 0$, and if A_g^π is not diagonalizable, we assume $\delta^{\mathrm{PID}} > 0$. If $\epsilon > 2/(1-\gamma-\delta^{\mathrm{TD}})$ and $T \geq c_1^{\mathrm{PID}}\epsilon/(1-\gamma-\delta^{\mathrm{TD}})$, we have*

$$\mathbb{E}\left[\left\|V_t^{\mathrm{TD}} - V^\pi\right\|_\infty^2\right] \leq c_2^{\mathrm{PID}} \left\|V_0 - V^\pi\right\|_\infty^2 \left(\frac{T}{t+T}\right)^{\epsilon(1-\gamma-\delta^{\mathrm{TD}})} + \frac{\epsilon(c_3^{\mathrm{PID}} + c_4^{\mathrm{PID}} \left\|V^\pi\right\|_\infty^2)}{\epsilon(1-\gamma-\delta^{\mathrm{TD}})-1} \cdot \left(\frac{\epsilon}{t+T}\right).$$

Here, $\{c_i^{\mathrm{PID}}\}$ are constants dependent on the MDP and δ^{TD}. Moreover, assume we initialize $V' = V_0, z \equiv 0$ in PID TD Learning and A_g^π has spectral radius $\rho < 1$. If $\epsilon > 2/(1-\rho-\delta^{\mathrm{PID}})$ and $T \geq c_1^{\mathrm{PID}}\epsilon/(1-\rho-\delta^{\mathrm{PID}})$, we have

$$\mathbb{E}\left[\left\|V_t^{\mathrm{PID}} - V^\pi\right\|_\infty^2\right] \leq c_2^{\mathrm{PID}} \left\|V_0 - V^\pi\right\|_\infty^2 \left(\frac{T}{t+T}\right)^{\epsilon(1-\rho-\delta^{\mathrm{PID}})} + \frac{\epsilon(c_3^{\mathrm{PID}} + c_4^{\mathrm{PID}} \left\|V^\pi\right\|_\infty^2)}{\epsilon(1-\rho-\delta^{\mathrm{PID}})-1} \cdot \left(\frac{\epsilon}{t+T}\right).$$

Here, $\{c_i^{\mathrm{PID}}\}$ are constants dependent on the MDP, controller gains, and δ^{PID}.

Proof of Theorem 3 for TD Learning

Proof. Since $\mathcal{P}^\pi$ is a stochastic matrix, we have $\rho(\mathcal{P}^\pi) = 1$. Based on Lemma 2, let S be the non-singular matrix such that $\|\gamma\mathcal{P}^\pi\|_{2,S} \leq \gamma + \delta^{\mathrm{TD}}$. For any two V_1 and V_2 we have

$$\|T^\pi V_1 - T^\pi V_2\|_{2,S} = \|\gamma\mathcal{P}^\pi(V_1 - V_2)\|_{2,S} \leq \|\gamma\mathcal{P}^\pi\|_{2,S} \|V_1 - V_2\|_{2,S} \leq (\gamma + \delta^{\mathrm{TD}}) \|V_1 - V_2\|_{2,S},$$

which means T^π is a $(\gamma + \delta^{\mathrm{TD}})$-contraction w.r.t. $\|\cdot\|_{2,S}$. Moreover, $\frac{1}{2}\|x\|_{2,S}^2$ is 1-smooth according to Lemma 3. Consequently, we use $\|\cdot\|_{2,S}$ as the norms $\|\cdot\|_c$ and $\|\cdot\|_s$ in Chen et al. (2020).

Assume the policy is d-deterministic. Define the noise $W_t \colon \mathcal{X} \to \mathbb{R}$ at iteration t as

$$W_t(x) \triangleq R_{x,t} + \gamma V_t(X'_{x,t}) - r^\pi(x) - \gamma(\mathcal{P}^\pi V_t)(x).$$

By Lemma 4, we can bound the conditional noise at each iteration as

$$\mathbb{E}\left[\|W_t\|_\infty^2 \middle| V_0, W_0, \ldots, W_{t-1}, V_t\right] \leq C^{\mathrm{TD}} + B^{\mathrm{TD}} \|V_t\|_\infty^2,$$

where

$$C^{\mathrm{TD}} \triangleq \frac{n(1-d)}{4}, \qquad B^{\mathrm{TD}} \triangleq 5\gamma^2 n(1-d).$$

There exists a constant $l^{\mathrm{TD}} > 0$ by the equivalence of norms and Lemma 2 such that for all $x \in \mathbb{R}^n$:

$$l^{\mathrm{TD}} \|x\|_\infty \leq \|x\|_{2,S} \leq \|x\|_\infty. \tag{8}$$

Based on these, define the constant

$$c_1^{\mathrm{TD}} \triangleq \frac{8(B^{\mathrm{TD}} + 2)}{{l^{\mathrm{TD}}}^2}.$$

By Corollary 2.1.2 of Chen et al. (2020), choosing the norm $\|\cdot\|_e$ to be $\|\cdot\|_\infty$, $\mu = L = 1$, the error at iteration t is bounded as

$$\mathbb{E}\left[\|V_t - V^\pi\|_{2,S}^2\right] \leq \|V_0 - V^\pi\|_{2,S}^2 \left(\frac{T}{t+T}\right)^{\epsilon(1-\gamma-\delta^{\mathrm{TD}})} + \frac{16e\epsilon^2\left(C^{\mathrm{TD}} + 2B^{\mathrm{TD}} \|V^\pi\|_{2,S}^2\right)}{{l^{\mathrm{TD}}}^2(\epsilon(1-\gamma-\delta^{\mathrm{TD}})-1)} \cdot \left(\frac{1}{t+T}\right).$$

By Equation (8), this immediately gives us a bound on the infinity norm.

$$\begin{aligned}\mathbb{E}\left[\|V_t - V^\pi\|_\infty^2\right] &\leq \frac{1}{{l^{\mathrm{TD}}}^2}\mathbb{E}\left[\|V_t - V^\pi\|_{2,S}^2\right] \\ &\leq \frac{1}{{l^{\mathrm{TD}}}^2}\|V_0 - V^\pi\|_{2,S}^2\left(\frac{T}{t+T}\right)^{\epsilon(1-\gamma-\delta^{\mathrm{TD}})} + \frac{16e\epsilon^2\left(C^{\mathrm{TD}} + 2B^{\mathrm{TD}}\|V^\pi\|_{2,S}^2\right)}{{l^{\mathrm{TD}}}^4(\epsilon(1-\gamma-\delta^{\mathrm{TD}})-1)}\cdot\left(\frac{1}{t+T}\right) \\ &\leq \frac{1}{{l^{\mathrm{TD}}}^2}\|V_0 - V^\pi\|_\infty^2\left(\frac{T}{t+T}\right)^{\epsilon(1-\gamma-\delta^{\mathrm{TD}})} + \frac{16e\epsilon^2\left(C^{\mathrm{TD}} + 2B^{\mathrm{TD}}\|V^\pi\|_\infty^2\right)}{{l^{\mathrm{TD}}}^4(\epsilon(1-\gamma-\delta^{\mathrm{TD}})-1)}\cdot\left(\frac{1}{t+T}\right).\end{aligned}$$

This gives the statement of theorem by defining

$$c_2^{\mathrm{TD}} \triangleq \frac{1}{{l^{\mathrm{TD}}}^2}, \quad c_3^{\mathrm{TD}} \triangleq \frac{16eC^{\mathrm{TD}}}{{l^{\mathrm{TD}}}^4}, \quad c_4^{\mathrm{TD}} \triangleq \frac{32eB^{\mathrm{TD}}}{{l^{\mathrm{TD}}}^4}.$$

□

Proof of Theorem 3 for PID TD Learning

Proof. The proof follows the exact steps in the proof for TD Learning. Based on Lemma 2, let S be the non-singular matrix such that $\left\|A_g^\pi\right\|_{2,S} \leq \rho + \delta^{\mathrm{PID}}$. For any two $\tilde{V}_1$ and $\tilde{V}_2$ we have

$$\left\|L_g^\pi\tilde{V}_1 - L_g^\pi\tilde{V}_2\right\|_{2,S} = \left\|A_g^\pi(\tilde{V}_1 - \tilde{V}_2)\right\|_{2,S} \leq \left\|A_g^\pi\right\|_{2,S}\left\|\tilde{V}_1 - \tilde{V}_2\right\|_{2,S} \leq (\rho + \delta^{\mathrm{PID}})\left\|\tilde{V}_1 - \tilde{V}_2\right\|_{2,S},$$

which means L_g^π is a $(\rho + \delta^{\mathrm{PID}})$-contraction w.r.t. $\|\cdot\|_{2,S}$. Moreover, $\frac{1}{2}\|x\|_{2,S}^2$ is 1-smooth according to Lemma 3. Consequently, we use $\|\cdot\|_{2,S}$ as the norms $\|\cdot\|_c$ and $\|\cdot\|_s$ in Chen et al. (2020).

Assume the policy is d-deterministic. Define the noise $\tilde{W}_t \colon \mathcal{X} \times \{\mathtt{v}, \mathtt{z}, \mathtt{v'}\} \to \mathbb{R}$ at iteration t as

$$\begin{aligned}\tilde{W}_t(x, \mathtt{v}) &\triangleq (\kappa_p + \kappa_I\alpha)(R_{x,t} + \gamma V_t(X'_{x,t}) - r^\pi(x) - \gamma(\mathcal{P}^\pi V_t)(x)), \\ \tilde{W}_t(x, \mathtt{z}) &\triangleq \alpha(R_{x,t} + \gamma V_t(X'_{x,t}) - r^\pi(x) - \gamma(\mathcal{P}^\pi V_t)(x)), \\ \tilde{W}_t(x, \mathtt{v'}) &\triangleq 0.\end{aligned}$$

By Lemma 4, we can bound the noise at each iteration as

$$\mathbb{E}\left[\left\|\tilde{W}_t\right\|_\infty^2 \middle| \tilde{V}_0, \tilde{W}_0, \ldots, \tilde{W}_{t-1}, \tilde{V}_t\right] \leq C^{\mathrm{PID}} + B^{\mathrm{PID}}\left\|\tilde{V}_t\right\|_\infty^2,$$

where

$$C^{\mathrm{PID}} \triangleq \frac{3n}{4}\max((\kappa_p + \kappa_I\alpha)^2, \alpha^2)(1-d), \qquad B^{\mathrm{PID}} \triangleq 15\gamma^2 n \cdot \max((\kappa_p + \kappa_I\alpha)^2, \alpha^2)(1-d).$$

There exists a constant $l > 0$ by the equivalence of norms and Lemma 2 such that for all $x \in \mathbb{R}^{3n}$:

$$l^{\mathrm{PID}}\|x\|_\infty \leq \|x\|_{2,S} \leq \|x\|_\infty. \tag{9}$$

Based on these, define the constants

$$c_1^{\mathrm{PID}} \triangleq \frac{8(B^{\mathrm{PID}} + 2)}{{l^{\mathrm{PID}}}^2}.$$

By Corollary 2.1.2 of Chen et al. (2020), choosing the norm $\|\cdot\|_e$ to be $\|\cdot\|_\infty$, the error at iteration t is bounded as

$$\mathbb{E}\left[\left\|\tilde{V}_t - \tilde{V}^\pi\right\|_{2,S}^2\right] \leq \left\|\tilde{V}_0 - \tilde{V}^\pi\right\|_{2,S}^2\left(\frac{T}{t+T}\right)^{\epsilon(1-\rho-\delta^{\mathrm{PID}})} + \frac{16e\epsilon^2\left(C^{\mathrm{PID}} + 2B^{\mathrm{PID}}\left\|\tilde{V}^\pi\right\|_{2,S}^2\right)}{{l^{\mathrm{PID}}}^2(\epsilon(1-\rho-\delta^{\mathrm{PID}})-1)}\cdot\left(\frac{1}{t+T}\right).$$

By Equation (9), this immediately gives us a bound on the infinity norm.

$$\begin{aligned}\mathbb{E}\left[\left\|\tilde{V}_t-\tilde{V}^\pi\right\|_\infty^2\right] &\le \frac{1}{{l^{\mathrm{PID}}}^2}\mathbb{E}\left[\left\|\tilde{V}_t-\tilde{V}^\pi\right\|_{2,S}^2\right]\\ &\le \frac{1}{{l^{\mathrm{PID}}}^2}\left\|\tilde{V}_0-\tilde{V}^\pi\right\|_{2,S}^2\left(\frac{T}{t+T}\right)^{\epsilon(1-\rho-\delta^{\mathrm{PID}})}+\frac{16e\epsilon^2\left(C^{\mathrm{PID}}+2B^{\mathrm{PID}}\left\|\tilde{V}^\pi\right\|_{2,S}^2\right)}{{l^{\mathrm{PID}}}^4(\epsilon(1-\rho-\delta^{\mathrm{PID}})-1)}\cdot\left(\frac{1}{t+T}\right)\\ &\le \frac{1}{{l^{\mathrm{PID}}}^2}\left\|\tilde{V}_0-\tilde{V}^\pi\right\|_\infty^2\left(\frac{T}{t+T}\right)^{\epsilon(1-\rho-\delta^{\mathrm{PID}})}+\frac{16e\epsilon^2\left(C^{\mathrm{PID}}+2B^{\mathrm{PID}}\left\|\tilde{V}^\pi\right\|_\infty^2\right)}{{l^{\mathrm{PID}}}^4(\epsilon(1-\rho-\delta^{\mathrm{PID}})-1)}\cdot\left(\frac{1}{t+T}\right).\end{aligned}$$

Since $\left\|V_t^{\mathrm{PID}}-V^\pi\right\|_\infty \le \left\|\tilde{V}_t-\tilde{V}^\pi\right\|_\infty$, $\left\|\tilde{V}^\pi\right\|_\infty = \left\|V^\pi\right\|_\infty$, and $\left\|\tilde{V}_0-\tilde{V}^\pi\right\|_\infty = \left\|V_0-V^\pi\right\|_\infty$ we finally get

$$\mathbb{E}\left[\left\|V_t^{\mathrm{PID}}-V^\pi\right\|_\infty^2\right] \le \frac{1}{{l^{\mathrm{PID}}}^2}\left\|V_0-V^\pi\right\|_\infty^2\left(\frac{T}{t+T}\right)^{\epsilon(1-\rho-\delta^{\mathrm{PID}})}+\frac{16e\epsilon^2\left(C^{\mathrm{PID}}+2B^{\mathrm{PID}}\left\|V^\pi\right\|_\infty^2\right)}{{l^{\mathrm{PID}}}^4(\epsilon(1-\rho-\delta^{\mathrm{PID}})-1)}\cdot\left(\frac{1}{t+T}\right).$$

This gives the statement of theorem by defining

$$c_2^{\mathrm{PID}} \triangleq \frac{1}{{l^{\mathrm{PID}}}^2}, \quad c_3^{\mathrm{PID}} \triangleq \frac{16eC^{\mathrm{PID}}}{{l^{\mathrm{PID}}}^4}, \quad c_4^{\mathrm{PID}} \triangleq \frac{32eB^{\mathrm{PID}}}{{l^{\mathrm{PID}}}^4}.$$

□

B.2 Proof of Proposition 1

Proof.

$$\frac{E_{\mathrm{opt}}^{\mathrm{TD}}(0)}{E_{\mathrm{stat}}^{\mathrm{TD}}(0)} = \frac{\left\|V_0-V^\pi\right\|_\infty^2}{{l^{\mathrm{TD}}}^2}\cdot\frac{{l^{\mathrm{TD}}}^4(\epsilon(1-\gamma)-1)T}{16e\epsilon^2\left(C^{\mathrm{TD}}+2B^{\mathrm{TD}}\left\|V^\pi\right\|_\infty^2\right)}$$

Due to the theorem conditions $T \ge \epsilon c_1^{\mathrm{TD}}/(1-\gamma)$ and $\epsilon \ge 2/(1-\gamma)$, which means $\epsilon(1-\gamma)-1 \ge \epsilon(1-\gamma)/2$. We continue

$$\begin{aligned}\frac{E_{\mathrm{opt}}^{\mathrm{TD}}(0)}{E_{\mathrm{stat}}^{\mathrm{TD}}(0)} &\ge \frac{\left\|V_0-V^\pi\right\|_\infty^2}{{l^{\mathrm{TD}}}^2}\cdot\frac{{l^{\mathrm{TD}}}^4\epsilon(1-\gamma)\cdot\epsilon c_1^{\mathrm{TD}}}{32e\epsilon^2\left(C^{\mathrm{TD}}+2B^{\mathrm{TD}}\left\|V^\pi\right\|_\infty^2\right)(1-\gamma)}\\ &= \frac{\left\|V_0-V^\pi\right\|_\infty^2{l^{\mathrm{TD}}}^2c_1^{\mathrm{TD}}}{32e\left(C^{\mathrm{TD}}+2B^{\mathrm{TD}}\left\|V^\pi\right\|_\infty^2\right)}\\ &= \frac{\left\|V_0-V^\pi\right\|_\infty^2\cdot 8(5\gamma^2n(1-d)+2)}{32e\left(n(1-d)/4+10\gamma^2n(1-d)\left\|V^\pi\right\|_\infty^2\right)}\\ &= \frac{\left\|V_0-V^\pi\right\|_\infty^2(5\gamma^2n(1-d)+2)}{en(1-d)\left(1+40\gamma^2\left\|V^\pi\right\|_\infty^2\right)}.\end{aligned}$$

Similarly for PID TD Learning, define $c = \max((\kappa_p + \kappa_I \alpha)^2, \alpha^2)$:

$$\begin{aligned}
\frac{E_{\text{opt}}^{\text{PID}}(0)}{E_{\text{stat}}^{\text{PID}}(0)} &= \frac{\|V_0 - V^\pi\|_\infty^2}{{l^{\text{PID}}}^2} \cdot \frac{{l^{\text{PID}}}^4 (\epsilon(1-\rho) - 1)T}{16e\epsilon^2 \left(C^{\text{PID}} + 2B^{\text{PID}} \|V^\pi\|_\infty^2\right)} \\
&\geq \frac{\|V_0 - V^\pi\|_\infty^2}{{l^{\text{PID}}}^2} \cdot \frac{{l^{\text{PID}}}^4 \epsilon(1-\rho) \cdot \epsilon c_1^{\text{PID}}}{32e\epsilon^2 \left(C^{\text{PID}} + 2B^{\text{PID}} \|V^\pi\|_\infty^2\right)(1-\rho)} \\
&= \frac{\|V_0 - V^\pi\|_\infty^2 {l^{\text{PID}}}^2 c_1^{\text{PID}}}{32e \left(C^{\text{PID}} + 2B^{\text{PID}} \|V^\pi\|_\infty^2\right)} \\
&= \frac{\|V_0 - V^\pi\|_\infty^2 \cdot 8(15\gamma^2 nc(1-d) + 2)}{32e \left(3nc(1-d)/4 + 30\gamma^2 nc(1-d) \|V^\pi\|_\infty^2\right)} \\
&= \frac{\|V_0 - V^\pi\|_\infty^2 (15\gamma^2 nc(1-d) + 2)}{3enc(1-d) \left(1 + 40\gamma^2 \|V^\pi\|_\infty^2\right)}.
\end{aligned}$$

□

C Details of Gain Adaptation

Tables 1 and 2 show the semi-gradients used for gain adaptation for Policy Evaluation and Control respectively. Algorithms 1 and 2 show the detailed description of the algorithm for Policy Evaluation and Control respectively.

Table 1: Semi-gradients of the Bellman residual used in the gain adaptation updates for Policy Evaluation. The learning rates are dropped to absorb them into η.

	Estimated semi-gradient of $(\text{BR}^\pi V_{t+1})(X_t)^2$
κ_p	$(R_t + \gamma V_{t+1}(X_t') - V_{t+1}(X_t)) \cdot (R_t + \gamma V_t(X_t') - V_t(X_t))$
κ_I	$(R_t + \gamma V_{t+1}(X_t') - V_{t+1}(X_t)) \cdot [\beta z_t(X_t) + \alpha(R_t + \gamma V_t(X_t') - V_t(X_t))]$
κ_d	$(R_t + \gamma V_{t+1}(X_t') - V_{t+1}(X_t)) \cdot (V_t(X_t) - V_t'(X_t))$

Table 2: Semi-gradients of the Bellman residual used in the gain adaptation updates for Control. The learning rates are dropped to absorb them into η.

	Estimated semi-gradient of $(\text{BR}^* Q_{t+1})(X_t, A_t)^2$
κ_p	$(R_t + \gamma \max_{a \in \mathcal{A}} Q_{t+1}(X_t', a) - Q_{t+1}(X_t, A_t)) \cdot (R_t + \gamma Q_t(X_t', A_t') - Q_t(X_t, A_t))$
κ_I	$(R_t + \gamma \max_{a \in \mathcal{A}} Q_{t+1}(X_t', a) - Q_{t+1}(X_t, A_t)) \cdot [\beta z_t(X_t, A_t) + \alpha(R_t + \gamma \max_{a \in \mathcal{A}} Q_t(X_t', a) - Q_t(X_t, A_t))]$
κ_d	$(R_t + \gamma \max_{a \in \mathcal{A}} Q_{t+1}(X_t', a) - Q_{t+1}(X_t, A_t)) \cdot (Q_t(X_t, A_t) - Q_t'(X_t, A_t))$

D Description of the Environments

D.1 Chain Walk

The environment consists of 50 states that are connected in a circular chain. The agent has two actions available, moving left or right. Upon taking an action, the agent succeeds with probability 0.7, stays in place with probability 0.1, and moves in the opposite direction with probability 0.2. The agent receives a reward of 1 when entering state 10, a reward of -1 when entering state 40, and a reward of 0 otherwise. The policy evaluated in the PE experiments is to always move left.

Algorithm 1 PID TD Learning with Gain Adaptation

Initialize V_1, V_1', z_1, $\texttt{previous\_V}_1$, $\texttt{running\_BR}_1$, and N_1 to zero on all states.
Initialize the gains to $\kappa_p = 1$, $\kappa_I = 0$, $\kappa_d = 0$.
for $t = 1, \ldots, K$ **do**
    Observe state X_t, take action $A_t \sim \pi(\cdot \mid X_t)$, receive reward R_t, and observe next state X_t'.
    Set $\delta' \leftarrow R_t + \gamma \cdot \texttt{previous\_V}_t(X_t') - \texttt{previous\_V}_t(X_t)$.
    Set $\delta \leftarrow R_t + \gamma V_t(X_t') - V_t(X_t)$.
    Update the gains:

$$\kappa_p \leftarrow \kappa_p + \eta \frac{\delta\delta'}{\texttt{running\_BR}_t(X_t) + \epsilon}$$
$$\kappa_I \leftarrow \kappa_I + \eta \frac{\delta(\beta z_t(X_t) + \alpha\delta')}{\texttt{running\_BR}_t(X_t) + \epsilon}$$
$$\kappa_d \leftarrow \kappa_d + \eta \frac{\delta(V_t(X_t) - V_t'(X_t))}{\texttt{running\_BR}_t(X_t) + \epsilon}.$$

    Set $\texttt{update} \leftarrow V_t(X_t) + \kappa_p\delta + \kappa_d(V_t(X_t) - V_t'(X_t)) + \kappa_I(\beta z_t(X_t) + \alpha\delta)$.
    Set $N_{t+1}(X_t) \leftarrow N_t(X_t) + 1$.
    Update the running values on the new states asynchronously:

$$\texttt{running\_BR}_{t+1}(X_t) \leftarrow (1-\lambda) \cdot \texttt{running\_BR}_t(X_t) + \lambda\delta^2$$
$$\texttt{previous\_V}_{t+1}(X_t) \leftarrow V_t(X_t)$$
$$V_{t+1}(X_t) \leftarrow (1 - \mu(N_t(X_t)))V_t(X_t) + \mu(N_t(X_t)) \cdot \texttt{update}$$
$$V_{t+1}'(X_t) \leftarrow (1 - \mu(N_t(X_t)))V_t'(X_t) + \mu(N_t(X_t))V_t(X_t)$$
$$z_{t+1}(X_t) \leftarrow (1 - \mu(N_t(X_t)))z_t(X_t) + \mu(N_t(X_t))(\beta z_t(X_t) + \alpha\delta).$$

end for

D.2 Cliff Walk

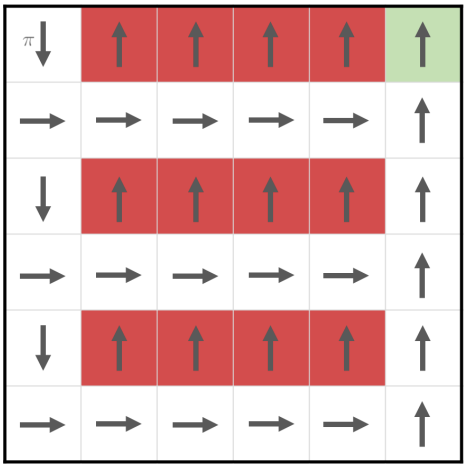

Figure 5: A visualization of Cliff Walk, taken from Rakhsha et al. (2022). The arrows depict the optimal policy.

A 6 by 6 grid world is used, visualized in Figure 5. The agent starts on the top left. Its goal is to end up on the top right. There are 12 cliff tiles, and the agent is stuck in them if it falls in. Moreover, the agent is stuck in the goal state once entering it. Upon making a move in the goal state, it receives a reward of 20. Making a move in a cliff receives a reward of -32, -16, or -8 depending on whether the

Algorithm 2 PID Q-Learning with Gain Adaptation

Initialize Q_1, Q'_1, z_1, $\texttt{previous\_Q}_1$, $\texttt{running\_BR}_1$ to zero on all state-action pairs, and N_1 to zero on all states.
Initialize the gains to $\kappa_p = 1$, $\kappa_I = 0$, $\kappa_d = 0$.
for $t = 1, \ldots, K$ **do**
    Let π be the policy derived from Q_t.
    Observe state X_t, take action $A_t \sim \pi(\cdot \mid X_t, A_t)$, receive reward R_t, and observe next state X'_t. Let $A'_t \leftarrow \max_a Q_t(X'_t, a)$.
    Set $\delta' \leftarrow R_t + \gamma \cdot \texttt{previous\_Q}_t(X'_t, A'_t) - \texttt{previous\_Q}_t(X_t, A_t)$.
    Set $\delta \leftarrow R_t + \gamma Q_t(X'_t, A'_t) - Q_t(X_t, A_t)$.
    Update the gains:

$$\kappa_p \leftarrow \kappa_p + \eta \frac{\delta\delta'}{\texttt{running\_BR}_t(X_t, A_t) + \epsilon}$$
$$\kappa_I \leftarrow \kappa_I + \eta \frac{\delta(\beta z_t(X_t, A_t) + \alpha\delta')}{\texttt{running\_BR}_t(X_t, A_t) + \epsilon}$$
$$\kappa_d \leftarrow \kappa_d + \eta \frac{\delta(Q_t(X_t, A_t) - Q'_t(X_t, A_t))}{\texttt{running\_BR}_t(X_t, A_t) + \epsilon}.$$

    Set $\texttt{update} \leftarrow Q_t(X_t, A_t) + \kappa_p\delta + \kappa_d(Q_t(X_t, A_t) - Q'_t(X_t, A_t)) + \kappa_I(\beta z_t(X_t, A_t) + \alpha\delta)$.
    Set $N_t(X_t) \leftarrow N_{t-1}(X_t) + 1$.
    Update the running values on the new state-action pair asynchronously:

$$\texttt{running\_BR}_{t+1}(X_t, A_t) \leftarrow (1 - \lambda) \cdot \texttt{running\_BR}_t(X_t, A_t) + \lambda\delta^2$$
$$\texttt{previous\_Q}_{t+1}(X_t, A_t) \leftarrow Q_t(X_t, A_t)$$
$$Q_{t+1}(X_t, A_t) \leftarrow (1 - \mu(N_t(X_t)))Q_t(X_t, A_t) + \mu(N_t(X_t)) \cdot \texttt{update}$$
$$Q'_{t+1}(X_t, A_t) \leftarrow (1 - \mu(N_t(X_t)))Q'_t(X_t, A_t) + \mu(N_t(X_t))Q_t(X_t, A_t)$$
$$z_{t+1}(X_t, A_t) \leftarrow (1 - \mu(N_t(X_t)))z_t(X_t, A_t) + \mu(N_t(X_t))(\beta z_t(X_t, A_t) + \alpha\delta).$$

end for

cliff is on the top, middle, or bottom respectively. Otherwise, it receives a reward of -1. The agent has four possible actions corresponding to moving up, down, left, and right. If the agent attempts to move off the grid, it simply stays in place. Otherwise, its action succeeds with probability 0.9, and moves in one of the other three directions at random with uniform probability The policy evaluated in the TD experiments is a random walk.

D.3 Garnet

The environment is randomly generated. They consist of 50 states, and 3 actions per state. To build the environment, for each action and state, (x, a), we pick 5 other random states $\mathcal{X}_{x,a}$. For 10 randomly chosen states x, we set $r(x)$ from a uniform distribution between 0 and 1. We set $r(x)$ is zero on all other states. Then, when taking action a and from state x, we receive reward $r(x)$ and move to any state in $\mathcal{X}_{x,a}$ with equal probability. The policy evaluated in the TD experiments is a random walk.

E Additional Experiments

Figure 6 shows the performance of gain adaptation on Chain Walk when $\gamma = 0.99$, and the corresponding movement of the controller gains. Figure 7 shows the performance of gain adaptation on Cliff Walk when $\gamma = 0.99$, and the corresponding movement of the controller gains.

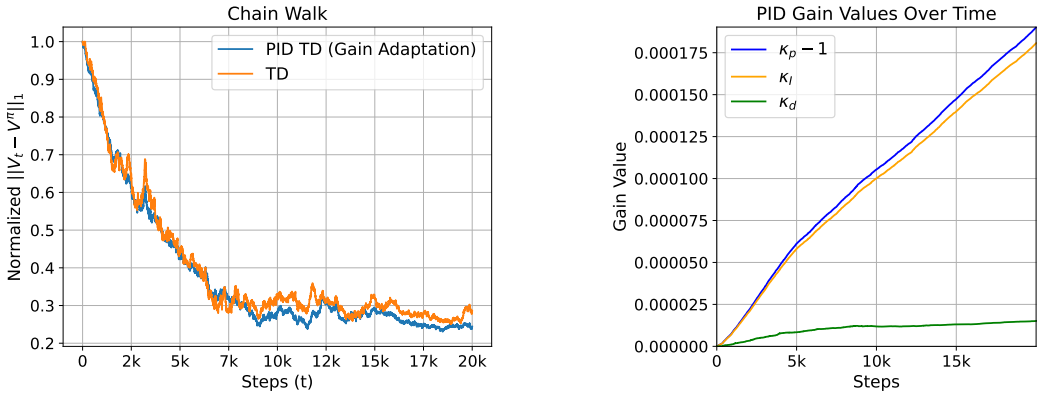

Figure 6: PID TD Learning with Gain Adaptation in Chain Walk with $\gamma = 0.99$. *(Left)* Comparison of value errors of PID TD Learning with TD Learning. Each curve is averaged over 80 runs. Shaded area shows standard error. *(Right)* The change of gains done by Gain Adaptation through training.

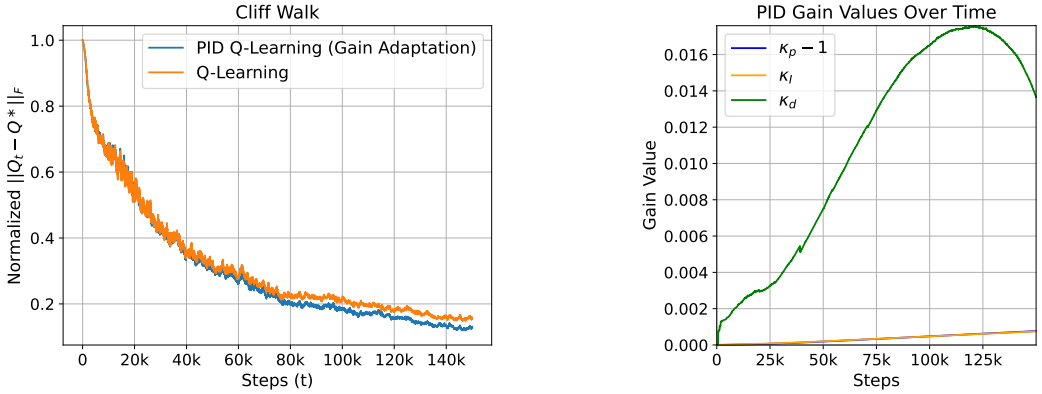

Figure 7: Q-Learning with Gain Adaptation in Cliff Walk with $\gamma = 0.99$. *(Left)* Comparison of value errors of PID Q-Learning with Q-Learning. Each curve is averaged over 80 runs. Shaded area shows standard error. *(Right)* The change of gains done by Gain Adaptation through training.

F Details of Experimental Setup

We pick the hyperparameters such that a normalized error of 0.2 is achieved the fastest, and if this error is not achieved, the final error is minimized. We fix $\alpha = 0.05$, $\beta = 0.95$, and $\lambda = 0.5$ throughout all the experiments.

For the Garnet (PE) experiments in Figure 4, we perform a grid search on $\eta \in \{0.1, 0.01, 0.001, 0.0001\}$, $\epsilon \in \{0.1, 0.01\}$. Similarly, for the Garnet (Control) experiments, we use $\epsilon = 0.1$ and perform a grid search over $\eta \in \{10^{-5}, 5 \times 10^{-5}, 10^{-6}\}$. The learning rates we perform a grid search over in these tests is listed in Table 3. The grid search is separately performed for each instance of the sampled Garnet. For TD Learning and Q-learning, the rates searched over are the same as that of the P component in Table 3. On each randomly generated Garnet environment, 80 runs are performed and the average trajectory is found. The variation of this average trajectory among all the 80 Garnet environments is shaded in Figure 4.

For the Cliff Walk policy evaluation experiments in Figure 2, we set $\eta = 10^{-5}$ and $\epsilon = 0.1$. For the Chain Walk (Control) experiments in Figure 3, we set $\eta = 4 \times 10^{-8}$ and $\epsilon = 10^{-4}$. For the Chain Walk (PE) experiments in Figure 6, we set $\eta = 5 \times 10^{-7}$ and $\epsilon = 10^{-1}$. For the Cliff Walk (Control) experiments in Figure 7, we set $\eta = 10^{-8}$ and $\epsilon = 10^{-1}$. For picking the learning rate

for the I and D component, we also consider learning rates of the form $\min(0.25, N_t(X_t)/M)$ with $M \in \{\infty, 10, 100, 500, 1000, 10000\}$.

	Learning Rates $\min(\epsilon, N_t(X_t)/M)$ searched through (formatted ϵ : corresponding set of M)
P Component	1: $\{10, 50, 100, 500, 1000, 10000\}$ 0.75: $\{10, 50, 100, 500, 1000\}$ 0.5: $\{10, 50, 100, 500, 1000\}$ 0.25: $\{10, 50, 100\}$ 0.1: $\{10, 50, 100\}$ 0.01: $\{10000\}$ 0.001: $\{10000\}$ 0.0001: $\{10000\}$
I Component	1: $\{\infty, 100\}$ 0.5: $\{\infty\}$ 0.1: $\{\infty\}$ 0: $\{\infty\}$
D Component	1: $\{\infty, 100\}$ 0.5: $\{\infty\}$ 0.25: $\{\infty\}$ 0.1: $\{\infty\}$ 0.01: $\{\infty\}$ 0: $\{\infty\}$

Table 3: All the learning rates searched through in the Garnet experiments.