Journal of Artificial Intelligence Research 1 (1993) 1-15 Submitted 6/91; published 9/91

# Proximal Gradient Temporal Difference Learning: Stable Reinforcement Learning with Polynomial Sample Complexity

**Bo Liu** BOLIU@AUBURN.EDU
*Auburn University*
*Auburn, AL, 36849, USA*

**Ian Gemp** IMGEMP@CS.UMASS.EDU
*UMass Amherst*
*Amherst, MA 01002, USA*

**Mohammad Ghavamzadeh** MGH@FB.COM
*Facebook AI Research*
*Menlo Park, CA 94025, USA*

**Ji Liu** JLIU@CS.ROCHESTER.EDU
*University of Rochester*
*Rochester, NY 14627, USA*

**Sridhar Mahadevan** SMAHADEV@ADOBE.COM
*Adobe Research*
*San Jose, CA 95110, USA*

**Marek Petrik** MPETRIK@CS.UNH.EDU
*University of New Hampshire*
*Durham, NH 03824, USA*

## Abstract

In this paper, we introduce proximal gradient temporal difference learning, which provides a principled way of designing and analyzing true stochastic gradient temporal difference learning algorithms. We show how gradient TD (GTD) reinforcement learning methods can be formally derived, not by starting from their original objective functions, as previously attempted, but rather from a primal-dual saddle-point objective function. We also conduct a saddle-point error analysis to obtain finite-sample bounds on their performance. Previous analyses of this class of algorithms use stochastic approximation techniques to prove asymptotic convergence, and do not provide any finite-sample analysis. We also propose an accelerated algorithm, called GTD2-MP, that uses proximal "mirror maps" to yield improved convergence rate. The results of our theoretical analysis imply that the GTD family of algorithms are comparable and may indeed be preferred over existing least squares TD methods for off-policy learning, due to their linear complexity. We provide experimental results showing the improved performance of our accelerated gradient TD methods.

## 1. Introduction

Obtaining a true stochastic gradient temporal difference method has been a longstanding goal of reinforcement learning (RL) for almost three decades (Bertsekas & Tsitsiklis, 1996; Sutton & Barto, 1998) ever since it was discovered that the original TD method was unstable in many off-policy applications, such as Q-learning, where the target behavior being learned and the exploratory behavior producing samples differ. Sutton et al. (Sutton, Szepesvári, & Maei, 2008; Sutton, Maei, Precup, Bhatnagar, Silver, Szepesvári, & Wiewiora, 2009) proposed a family of gradient-based

temporal difference (GTD) algorithms, which yielded several interesting properties. A key property of this class of GTD algorithms is that they are asymptotically convergent in the off-policy setting. However, the original derivation of these methods was somewhat ad-hoc, as the derivation from the original loss functions involved some non-mathematical steps (such as an arbitrary decomposition of the resulting product of gradient terms). Consequently, the resulting convergence analysis was also weakened, and limited to showing the asymptotic convergence using stochastic approximation (Borkar, 2008). Despite these shortcomings, gradient TD was a significant advance, as previous work on off-policy methods, such as TD($\lambda$), do not have convergence guarantees in the off-policy setting. A further appealing property of these algorithms is the first-order computational complexity that allows them to scale more gracefully to high-dimensional problems, unlike the widely used least-squares TD (LSTD) approaches (Bradtke & Barto, 1996) that only perform well with moderate size reinforcement learning (RL) problems, due to their quadratic (w.r.t. the dimension of the feature space) computational cost per iteration.

Unfortunately, despite the nomenclature, GTD algorithms are *not true stochastic gradient methods with respect to their original objective functions*, as pointed out by Szepesvari (Szepesvári, 2010). The reason is not surprising: the gradient of the objective function involves products of terms, which cannot be sampled directly. Consequently, their original derivation involved a rather ad-hoc splitting of terms, which was justified more or less from intuition. In this paper, we take a major step forward in resolving this problem by showing a principled way of designing true stochastic gradient TD algorithms by using a primal-dual saddle point objective function, derived from the original objective functions, coupled with the powerful machinery of *operator splitting* (Bauschke & Combettes, 2011). A significant advantage of our approach is that it enables undertaking a precise *finite sample analysis* of convergence, which provides deeper insight into the actual running times of the GTD methods beyond the standard asymptotic analysis.

Since in real-world applications of RL, we have access to only a finite amount of data, finite-sample analysis of gradient TD algorithms is essential as it clearly shows the effect of the number of samples (and the parameters that play a role in the sampling budget of the algorithm) on their final performance. However, most of the work on the finite-sample analysis in RL has been focused on batch RL (or approximate dynamic programming) algorithms (e.g., (Kakade & Langford, 2002; Munos & Szepesvári, 2008; Antos, Szepesvari, & Munos, 2008; Lazaric, Ghavamzadeh, & Munos, 2010a)), especially those that are least squares TD (LSTD)-based (e.g., (Lazaric, Ghavamzadeh, & Munos, 2010b; Ghavamzadeh, Lazaric, Maillard, & Munos, 2010; Ghavamzadeh, Lazaric, Munos, & Hoffman, 2011; Lazaric, Ghavamzadeh, & Munos, 2012)), and more importantly restricted to the on-policy setting. In this paper, we provide the finite-sample analysis of the GTD family of algorithms, a relatively novel class of gradient-based TD methods that are guaranteed to converge even in the off-policy setting, and for which, to the best of our knowledge, no finite-sample analysis has been reported. This analysis is challenging because **1)** the stochastic approximation methods that have been used to prove the asymptotic convergence of these algorithms do not address convergence rate analysis; **2)** as we explain in detail in Section 2.1, the techniques used for the analysis of the stochastic gradient methods cannot be applied here; **3)** finally, the difficulty of finite-sample analysis in the off-policy setting. It should also be noted that there exists very little literature on the finite-sample analysis in the off-policy setting, even for the LSTD-based algorithms that have been extensively studied.

The major contributions of this paper include

- The first finite-sample analyses of the TD algorithms with linear computational complexity, which is also one of the first few finite-sample analyses of off-policy convergent TD algorithms.
- A novel framework for designing gradient-based TD algorithms with Bellman Error based objective functions, as well as the design and analysis of several improved GTD methods that result from our novel approach of formulating gradient TD methods as true stochastic gradient algorithms w.r.t. a saddle-point objective function.

We then use the techniques applied in the analysis of the stochastic gradient methods to propose a unified finite-sample analysis for the previously proposed GTD algorithms as well as our novel gradient TD algorithms. Finally, given the results of our analysis, we study the GTD class of algorithms from several different perspectives, including acceleration in convergence, learning with biased importance sampling factors, etc.

## 2. Preliminaries

Reinforcement Learning (RL) (Bertsekas & Tsitsiklis, 1996; Sutton & Barto, 1998) is a subfield of machine learning, studying a class of problems in which an agent interacts with an unfamiliar, dynamic and stochastic environment, with the goal of optimizing some measure of its long-term performance. This interaction is conventionally modeled as a Markov decision process (MDP). An MDP is defined as the tuple $(\mathcal{S}, \mathcal{A}, P^a_{ss'}, R, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are the sets of states and actions, $P^a_{ss'}$ is the transition kernel specifying the probability of transition from state $s \in \mathcal{S}$ to state $s' \in \mathcal{S}$ by taking action $a \in \mathcal{A}$, $R(s,a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function bounded by $R_{\max}$, and $0 \leq \gamma < 1$ is a discount factor. A stationary policy $\pi : \mathcal{S} \times \mathcal{A} \to [0,1]$ is a probabilistic mapping from states to actions. The main objective of an RL algorithm is to find an optimal policy. In order to achieve this goal, a key step in many algorithms is to calculate the value function of a given policy $\pi$, i.e., $V^\pi : \mathcal{S} \to \mathbb{R}$, a process known as *policy evaluation*. It is known that $V^\pi$ is the unique fixed-point of the *Bellman operator* $T^\pi$, i.e.,

$$V^\pi = T^\pi V^\pi = R^\pi + \gamma P^\pi V^\pi, \tag{1}$$

where $R^\pi$ and $P^\pi$ are the reward function and transition kernel of the Markov chain induced by policy $\pi$. In Eq. (1), we may imagine $V^\pi$ as an $|\mathcal{S}|$-dimensional vector and write everything in vector/matrix form. In the following, to simplify the notation, we often drop the dependence of $T^\pi$, $V^\pi$, $R^\pi$, and $P^\pi$ on $\pi$.

Off-policy learning refers to learning about one way of behaving, termed as the *target policy*, from data generated by another way of selecting actions, termed as the *behavior policy*. The target policy is often a deterministic policy that approximates the optimal policy. On the other hand, the behavior policy is often stochastic, exploring all possible actions in each state in order to find the optimal policy. There are several benefits of learning with behavior policies. First, it allows freeing the behavior policy from the target policy and thus has a greater variety of exploration strategies to be used. Secondly, it enables learning from training data generated by unrelated controllers, including manual human control, and from previously collected data. The third reason for interest in off-policy learning is that it permits learning about multiple target policies (e.g., optimal policies for multiple sub-goals) from a single stream of data generated by a single behavior policy, i.e., parallel learning is allowed for off-policy learning (Maei, 2011).

In the paper, we denote by $\pi_b$, the behavior policy that generates the data, and by $\pi$, the target policy that we would like to evaluate. They are the same in the on-policy setting and different in the off-policy setting. For each state-action pair $(s_i, a_i)$, such that $\pi_b(a_i|s_i) > 0$, we define the importance-weighting factor $\rho_i = \pi(a_i|s_i)/\pi_b(a_i|s_i)$ with $\rho_{\max} \geq 0$ being its maximum value over the state-action pairs.

When $\mathcal{S}$ is large or infinite, we often use a linear approximation architecture for $V^\pi$ with parameters $\theta \in \mathbb{R}^d$ and $L$-bounded basis functions $\{\varphi_i\}_{i=1}^d$, i.e., $\varphi_i : \mathcal{S} \to \mathbb{R}$ and $\max_i ||\varphi_i||_\infty \leq L$. We denote by $\phi(\cdot) = \big(\varphi_1(\cdot), \ldots, \varphi_d(\cdot)\big)^\top$ the feature vector and by $\mathcal{F}$ the linear function space spanned by the basis functions $\{\varphi_i\}_{i=1}^d$, i.e., $\mathcal{F} = \big\{f_\theta \mid \theta \in \mathbb{R}^d \text{ and } f_\theta(\cdot) = \phi(\cdot)^\top\theta\big\}$. We may write the approximation of $V$ in $\mathcal{F}$ in the vector form as $\hat{v} = \Phi\theta$, where $\Phi$ is the $|\mathcal{S}| \times d$ feature matrix. When only $n$ training samples of the form $\mathcal{D} = \big\{\big(s_i, a_i, r_i = r(s_i, a_i), s_i'\big)\big\}_{i=1}^n,\ s_i \sim \xi,\ a_i \sim \pi_b(\cdot|s_i),\ s_i' \sim P(\cdot|s_i, a_i)$, are available ($\xi$ is a distribution over the state space $\mathcal{S}$), we may write the *empirical Bellman operator* $\hat{T}$ for a function in $\mathcal{F}$ as

$$\hat{T}(\hat{\Phi}\theta) = \hat{R} + \gamma\hat{\Phi}'\theta,$$

where $\hat{\Phi}$ (resp. $\hat{\Phi}'$) is the empirical feature matrix of size $n \times d$, whose $i$-th row is the feature vector $\phi(s_i)^\top$ (resp. $\phi(s_i')^\top$), and $\hat{R} \in \mathbb{R}^n$ is the reward vector, whose $i$-th element is $r_i$. $\phi(s_i)$ (resp. $\phi(s_i')$) will be denoted as $\phi$ (resp. $\phi'$) for short. We denote by $\delta_i(\theta) = r_i + \gamma\phi_i'^\top\theta - \phi_i^\top\theta$, the TD error for the $i$-th sample $(s_i, r_i, s_i')$ and define $\Delta\phi_i = \phi_i - \gamma\phi_i'$. Finally, we define the matrices $A$ and $C$, and the vector $b$ as

$$A := \mathbb{E}\big[\rho_i\phi_i(\Delta\phi_i)^\top\big],\ \ b := \mathbb{E}\left[\rho_i\phi_i r_i\right],\ \ C := \mathbb{E}[\phi_i\phi_i^\top], \tag{2}$$

where the expectations are w.r.t. $\xi$ and $P^{\pi_b}$. We also denote by $\Xi$, the diagonal matrix whose elements are $\xi(s)$, and $\xi_{\max} := \max_s \xi(s)$. For each sample $i$ in the training set $\mathcal{D}$, we can calculate an unbiased estimate of $A$, $b$, and $C$ as follows:

$$\hat{A}_i := \rho_i\phi_i\Delta\phi_i^\top, \quad \hat{b}_i := \rho_i r_i\phi_i, \quad \hat{C}_i := \phi_i\phi_i^\top. \tag{3}$$

### 2.1 Gradient-based TD Algorithms

The class of gradient-based TD (GTD) algorithms was proposed by Sutton et al. (Sutton et al., 2008, 2009). These algorithms target two objective functions: the *norm of the expected TD update* (NEU) and the *mean-square projected Bellman error* (MSPBE), defined as (see e.g., (Maei, 2011))[1]

$$\text{NEU}(\theta) = ||\Phi^\top\Xi(T\hat{v} - \hat{v})||^2\,, \tag{4}$$

$$\text{MSPBE}(\theta) = ||\hat{v} - \Pi T\hat{v}||_\xi^2 = ||\Phi^\top\Xi(T\hat{v} - \hat{v})||_{C^{-1}}^2\,, \tag{5}$$

where $C = \mathbb{E}[\phi_i\phi_i^\top] = \Phi^\top\Xi\Phi$ is the covariance matrix defined in Eq. (2) and is assumed to be non-singular, and for $\forall x \in \mathbb{R}^{d\times 1}, ||x||_{C^{-1}}^2 = x^\top C^{-1}x$. $\Pi = \Phi(\Phi^\top\Xi\Phi)^{-1}\Phi^\top\Xi$ is the orthogonal projection operator onto the function space $\mathcal{F}$, i.e., for any bounded function $g$, $\Pi g = \arg\min_{f\in\mathcal{F}}||g - f||_\xi = \arg\min_{f\in\mathcal{F}}(g-f)^\top\text{diag}(\xi)(g-f)$. From Eq. (4) and Eq. (5), it is clear that NEU and MSPBE are square unweighted and weighted by $C^{-1}$, $\ell_2$-norms of the quantity $\Phi^\top\Xi(T\hat{v} - \hat{v})$, respectively, and thus, the two objective functions can be unified as

$$J(\theta) = ||\Phi^\top\Xi(T\hat{v} - \hat{v})||_{M^{-1}}^2 = ||\mathbb{E}[\rho_i\delta_i(\theta)\phi_i]||_{M^{-1}}^2, \tag{6}$$

1. It is important to note that $T$ in Eq. (4) and Eq. (5) is $T^\pi$, the Bellman operator of the target policy $\pi$.

with $M$ equal to the identity matrix $I$ for NEU and to the covariance matrix $C$ for MSPBE. The second equality in (6) holds because of the following lemma from Section 4.2 in (Maei, 2011).

**Lemma 1.** *(Importance-weighting for off-policy TD) Let $\mathcal{D} = \big\{\big(s_i, a_i, r_i, s_i'\big)\big\}_{i=1}^n,\ s_i \sim \xi,\ a_i \sim \pi_b(\cdot|s_i),\ s_i' \sim P(\cdot|s_i, a_i)$ be a training set generated by the behavior policy $\pi_b$ and $T$ be the Bellman operator of the target policy $\pi$. Then, we have*

$$\Phi^\top \Xi (T\hat{v} - \hat{v}) = \mathbb{E}\big[\rho_i \delta_i(\theta)\phi_i\big] = b - A\theta.$$

*Proof.* We give a proof sketch here. Refer to Section 4.2 in (Maei, 2011) for a detailed proof.

$$\begin{aligned}
&\Phi^\top \Xi (T\hat{v} - \hat{v}) \\
&= \sum_{s,a,s'} \xi(s)\pi(a|s)P(s'|s,a)\delta(\theta|s,a,s')\phi(s) \\
&= \sum_{s,a,s'} \xi(s)\frac{\pi(a|s)}{\pi_b(a|s)}\pi_b(a|s)P(s'|s,a)\delta(\theta|s,a,s')\phi(s) \\
&= \sum_{s,a,s'} \xi(s)\rho(s,a)\pi_b(a|s)P(s'|s,a)\delta(\theta|s,a,s')\phi(s) \\
&= \mathbb{E}[\rho_t \delta_t(\theta)\phi_t] \\
&= b - A\theta
\end{aligned}$$

■

Motivated by minimizing the NEU and MSPBE objective functions using the stochastic gradient methods, the GTD and GTD2 algorithms were proposed with the following update rules:

$$\textbf{GTD:} \qquad \begin{aligned} y_{t+1} &= y_t + \alpha_t\big(\rho_t\delta_t(\theta_t)\phi_t - y_t\big), \\ \theta_{t+1} &= \theta_t + \alpha_t\rho_t\Delta\phi_t(y_t^\top\phi_t), \end{aligned} \tag{7}$$

$$\textbf{GTD2:} \qquad \begin{aligned} y_{t+1} &= y_t + \alpha_t\big(\rho_t\delta_t(\theta_t) - \phi_t^\top y_t\big)\phi_t, \\ \theta_{t+1} &= \theta_t + \alpha_t\rho_t\Delta\phi_t(y_t^\top\phi_t). \end{aligned} \tag{8}$$

However, it has been shown that the above update rules do not update the value function parameter $\theta$ in the gradient direction of NEU and MSPBE, and thus, NEU and MSPBE are not the true objective functions of the GTD and GTD2 algorithms (Szepesvári, 2010). Consider the NEU objective function in (4). Taking its gradient w.r.t. $\theta$, we obtain

$$\begin{aligned}
-\frac{1}{2}\nabla \mathrm{NEU}(\theta) &= -\big(\nabla\mathbb{E}\big[\rho_i\delta_i(\theta)\phi_i^\top\big]\big)\mathbb{E}\big[\rho_i\delta_i(\theta)\phi_i\big] \\
&= -\big(\mathbb{E}\big[\rho_i\nabla\delta_i(\theta)\phi_i^\top\big]\big)\mathbb{E}\big[\rho_i\delta_i(\theta)\phi_i\big] \\
&= \mathbb{E}\big[\rho_i\Delta\phi_i\phi_i^\top\big]\mathbb{E}\big[\rho_i\delta_i(\theta)\phi_i\big].
\end{aligned}$$

If the gradient can be written as a single expectation, then it is straightforward to use a stochastic gradient method. However, we have a product of two expectations in (9), and unfortunately,

due to the correlation between them, the sample product (with a single sample) won't be an unbiased estimate of the gradient. To tackle this, the GTD algorithm uses an auxiliary variable $y_t$ to estimate $\mathbb{E}\big[\rho_i\delta_i(\theta)\phi_i\big]$, and thus, the overall algorithm is no longer a true stochastic gradient method w.r.t. NEU. It can be easily shown that the same problem exists for GTD2 w.r.t. the MSPBE objective function. This prevents us from using the standard convergence analysis techniques of stochastic gradient descent methods to obtain a finite-sample performance bound for the GTD and GTD2 algorithms.

It should be also noted that in the original publications of GTD/GTD2 algorithms (Sutton et al., 2008, 2009), the authors discussed handling the off-policy scenario using both importance and rejection sampling. In rejection sampling, which was mainly used in (Sutton et al., 2008, 2009), a sample $(s_i, a_i, r_i, s_i')$ is rejected and the parameter $\theta$ is not updated if $\pi(a_i|s_i) = 0$. This sampling strategy is not efficient since a lot of samples will be discarded if $\pi_b$ and $\pi$ are very different.

### 2.2 Related Work

Before we present a finite-sample performance bound for GTD and GTD2, it would be helpful to give a brief overview of the existing literature on the finite-sample analysis of the TD algorithms. The convergence rate of the TD algorithms mainly depends on $(d, n, \nu)$, where $d$ is the size of the approximation space (the dimension of the feature vector), $n$ is the number of samples, and $\nu$ is the smallest eigenvalue of the sample-based covariance matrix $\hat{C} = \hat{\Phi}^\top\hat{\Phi}$, i.e., $\nu = \lambda_{\min}(\hat{C})$.

Antos et al.(Antos et al., 2008) proved an error bound of $O(\frac{d\log d}{n^{1/4}})$ for LSTD in bounded spaces. Lazaric et al. (Lazaric et al., 2010b) proposed an LSTD analysis in linear spaces and obtained a tighter bound of $O(\sqrt{\frac{d\log d}{n\nu}})$ and later used it to derive a bound for the least-squares policy iteration (LSPI) algorithm (Lazaric et al., 2012). Tagorti et al. (Tagorti & Scherrer, 2014) recently proposed the first convergence analysis for LSTD($\lambda$) and derived a bound of $\tilde{O}(d/\nu\sqrt{n})$. The analysis is a bit different than the one in (Lazaric et al., 2010b) and the bound is weaker in terms of $d$ and $\nu$. Another recent result is by (Prashanth, Korda, & Munos, 2014) that uses stochastic approximation to solve LSTD$(0)$, where the resulting algorithm is exactly TD$(0)$ with random sampling (samples are drawn i.i.d. and not from a trajectory), and report a Markov design bound (the bound is computed only at the states used by the algorithm) of $O(\sqrt{\frac{d}{n\nu}})$ for LSTD$(0)$. All these results are for the on-policy setting, except the one by (Antos et al., 2008) that also holds for the off-policy formulation. Another result in the off-policy setting is by (Pires & Szepesvari, 2012) that uses a bounding trick and improves the result of (Antos et al., 2008) by a $\log d$ factor. Another line of work is by (Yu, 2012), which provides error bounds of LSTD algorithms for a wide range of problems including the scenario that $||A||_\xi$ is unbounded, which is beyond the scope of the aforementioned literature and our paper.

The line of research reported here has much in common with work on proximal reinforcement learning (Mahadevan, Liu, Thomas, Dabney, Giguere, Jacek, Gemp, & Liu, 2014), which explores first-order reinforcement learning algorithms using *mirror maps* (Bubeck, 2014; Juditsky, Nemirovskii, & Tauvel, 2008) to construct primal-dual spaces. This work began originally with a dual space formulation of first-order sparse TD learning (Mahadevan & Liu, 2012). The saddle point formulation for off-policy TD learning was initially explored in (Liu, Mahadevan, & Liu, 2012), where the objective function is the norm of the approximation residual of a linear inverse problem (Pires & Szepesvari, 2012). A sparse off-policy GTD2 algorithm with regularized dual averaging is introduced by Qin et al. (Qin & Li, 2014). These studies provide different approaches

to formulating the problem 1) as a variational inequality problem (Juditsky et al., 2008; Mahadevan et al., 2014), 2) as a linear inverse problem (Liu et al., 2012), or 3) as a quadratic objective function (MSPBE) using two-time-scale solvers (Qin & Li, 2014). In this paper, we are going to explore the true nature of the GTD algorithms as stochastic gradient algorithms w.r.t the convex-concave saddle-point formulations of NEU and MSPBE.

## 3. Saddle-Point Formulation of GTD Algorithms

In this section, we show how the GTD and GTD2 algorithms can be formulated as true stochastic gradient (SG) algorithms by writing their respective objective functions, NEU and MSPBE, in the form of a convex-concave saddle-point. As discussed earlier, this new formulation of GTD and GTD2 as true SG methods allows us to use the convergence analysis techniques for SGs in order to derive finite-sample performance bounds for these RL algorithms. Moreover, it allows us to use more efficient algorithms that have been recently developed to solve SG problems, such as *stochastic Mirror-Prox* (SMP) (Juditsky et al., 2008), to derive more efficient versions of GTD and GTD2.

A particular type of convex-concave saddle-point formulation is formally defined as

$$\min_{\theta} \max_{y} \big(L(\theta, y) = \langle b - A\theta, y\rangle + F(\theta) - K(y)\big), \tag{10}$$

where $F(\theta)$ is a convex function and $K(y)$ is a smooth convex function such that

$$K(y) - K(x) - \langle \nabla K(x), y - x\rangle \leq \frac{L_K}{2}||x - y||^2. \tag{11}$$

Next we follow (Juditsky et al., 2008; Nemirovski, Juditsky, Lan, & Shapiro, 2009; Chen, Lan, & Ouyang, 2013) and define the following error function for the saddle-point problem (10).

**Definition 1.** *The error function of the saddle-point problem* (10) *at each point* $(\theta', y')$ *is defined as*

$$\mathrm{Err}(\theta', y') = \max_{y} \; L(\theta', y) - \min_{\theta} \; L(\theta, y'). \tag{12}$$

In this paper, we consider the saddle-point problem (10) with $F(\theta) = 0$ and $K(y) = \frac{1}{2}||y||_M^2$, i.e.,

$$\min_{\theta} \max_{y} \Big(L(\theta, y) = \langle b - A\theta, y\rangle - \frac{1}{2}||y||_M^2\Big), \tag{13}$$

where $A$ and $b$ were defined by Eq. (2), and $M$ is a positive definite matrix. It can be shown that $K(y) = \frac{1}{2}||y||_M^2$ satisfies the condition in Eq. (11) by using the Taylor expansion of $y$ about $x$, i.e.,

$$\frac{1}{2}||y||_M^2 \geq \frac{1}{2}||x||_M^2 + M(y - x) + \frac{L_K}{2}||y - x||_M^2$$

We first show in Proposition 1 that if $(\theta^*, y^*)$ is the saddle-point of problem (13), then $\theta^*$ will be the optimum of NEU and MSPBE defined in Eq. (6). We then prove in Proposition 2 that GTD and GTD2 in fact find this saddle-point.

**Proposition 1.** *For any fixed* $\theta$*, we have* $\frac{1}{2}J(\theta) = \max_y L(\theta, y)$*, where* $J(\theta)$ *is defined by Eq.* (6).

Readers familiar with Fenchel duality or Legendre transform can easily prove this by using the fact that the Legendre-Fenchel convex conjugate function (Boyd & Vandenberghe, 2004) of $f = \frac{1}{2}||Ax - b||_{M^{-1}}$ is $f^* = \frac{1}{2}||Ax - b||_M$, and

$$f(x) = \frac{1}{2}||Ax - b||^2_{M^{-1}} = f^{**}(x) = \max_y \big(y^\top (Ax - b) - \frac{1}{2}||y||^2_M\big)$$

The second equality holds since $f(x)$ is convex. We can also prove this via another way as follows.

*Proof.* Since $L(\theta, y)$ is an unconstrained quadratic program w.r.t. $y$, the optimal $y^*(\theta) = \arg\max_y L(\theta, y)$ can be analytically computed as

$$y^*(\theta) = M^{-1}(b - A\theta).$$

The result follows by plugging $y^*$ into (13) and using the definition of $J(\theta)$ in Eq. (6) and Lemma 1. ■

**Proposition 2.** *GTD and GTD2 are true stochastic gradient algorithms w.r.t. the objective function $L(\theta, y)$ of the saddle-point problem* (13) *with $M = I$ and $M = C = \Phi^\top \Xi \Phi$ (the covariance matrix), respectively.*

*Proof.* It is easy to see that the gradient updates of the saddle-point problem (13) (ascending in $y$ and descending in $\theta$) may be written as

$$\begin{aligned} y_{t+1} &= y_t + \alpha_t \left(b - A\theta_t - My_t\right), \\ \theta_{t+1} &= \theta_t + \alpha_t A^\top y_t. \end{aligned}$$

We denote $\hat{M} := I$ (resp. $\hat{M} := \hat{C}$) for GTD (resp. GTD2). We may obtain the update rules of GTD and GTD2 by replacing $A$, $b$, and $C$ in (14) with their unbiased estimates $\hat{A}$, $\hat{b}$, and $\hat{C}$ from Eq. (3), which completes the proof. ■

## 4. Finite-Sample Analysis

In this section, we provide a finite-sample analysis for a revised version of the GTD/GTD2 algorithms. We first describe the revised GTD algorithms in Section 4.1 and then dedicate the rest of Section 4 to their sample analysis. Note that from now on we use the $M$ matrix (and its unbiased estimate $\hat{M}_t$) to have a unified analysis of GTD and GTD2 algorithms. As described earlier, $M$ is replaced by the identity matrix $I$ in GTD and by the covariance matrix $C$ (and its unbiased estimate $\hat{C}_t$) in GTD2.

### 4.1 The Revised GTD Algorithms

The revised GTD algorithms that we analyze in this paper (see Algorithm 1) have three differences with the standard GTD algorithms of Eqs. (7) and (8) (and Eq. (14)).

- We guarantee that the parameters $\theta$ and $y$ remain bounded by projecting them onto bounded convex feasible sets $\Theta$ and $Y$ defined in Assumption 2. In Algorithm 1, we denote by $\Pi_\Theta$ and $\Pi_Y$, the projection onto sets $\Theta$ and $Y$, respectively. This is standard in stochastic approximation algorithms and has been used in off-policy TD($\lambda$) (Yu, 2012) and actor-critic algorithms (e.g., (Bhatnagar, Sutton, Ghavamzadeh, & Lee, 2009)).

- After $n$ iterations ($n$ is the number of training samples in $\mathcal{D}$), the algorithms return the weighted (by the step size) average of the parameters at all the $n$ iterations (see Eq. (15)).
- The step-size $\alpha_t$ is selected as described in the proof of Proposition 3 in the Appendix. Note that this fixed step size of $O(1/\sqrt{n})$ is required for the high-probability bound in Proposition 3 (see (Nemirovski et al., 2009) for more details).

**Algorithm 1** Revised GTD Algorithms

1: **for** $t = 1, \ldots, n$ **do**
2: Update parameters

$$y_{t+1} = \Pi_Y \Big( y_t + \alpha_t (\hat{b}_t - \hat{A}_t \theta_t - \hat{M}_t y_t) \Big)$$
$$\theta_{t+1} = \Pi_\Theta \Big( \theta_t + \alpha_t \hat{A}_t^\top y_t \Big)$$

3: **end for**
4: OUTPUT

$$\bar{\theta}_n := \frac{\sum_{t=1}^n \alpha_t \theta_t}{\sum_{t=1}^n \alpha_t} \quad , \quad \bar{y}_n := \frac{\sum_{t=1}^n \alpha_t y_t}{\sum_{t=1}^n \alpha_t} \tag{15}$$

### 4.2 Assumptions

In this section, we make several assumptions on the MDP and basis functions that are used in our finite-sample analysis of the revised GTD algorithms. These assumptions are quite standard and are similar to those made in the prior work on GTD algorithms (Sutton et al., 2008, 2009; Maei, 2011) and those made in the analysis of SG algorithms (Nemirovski et al., 2009).

**Assumption 2.** *(**Feasibility Sets**) We define the bounded closed convex sets $\Theta \subset \mathbb{R}^d$ and $Y \subset \mathbb{R}^d$ as the feasible sets in Algorithm 1. We further assume that the saddle-point $(\theta^*, y^*)$ of the optimization problem* (13) *belongs to $\Theta \times Y$. We also define $D_\theta := \big[ \max_{\theta \in \Theta} ||\theta||_2^2 - \min_{\theta \in \Theta} ||\theta||_2^2 \big]^{1/2}$, $D_y := \big[ \max_{y \in Y} ||y||_2^2 - \min_{y \in Y} ||y||_2^2 \big]^{1/2}$, and $R = \max \big\{ \max_{\theta \in \Theta} ||\theta||_2, \max_{y \in Y} ||y||_2 \big\}$.*

**Assumption 3.** *(**Non-singularity**) We assume that the covariance matrix $C = \mathbb{E}[\phi_i \phi_i^\top]$ and matrix $A = \mathbb{E}\big[\rho_i \phi_i (\Delta \phi_i)^\top\big]$ are non-singular.*

**Assumption 4.** *(**Boundedness**) We assume the features $(\phi_i, \phi_i')$ have uniformly bounded second moments. This together with the boundedness of features (by $L$) and importance weights (by $\rho_{\max}$) guarantees that the matrices $A$ and $C$, and vector $b$ are uniformly bounded.*

This assumption guarantees that for any $(\theta, y) \in \Theta \times Y$, the unbiased estimators of $b - A\theta - My$ and $A^\top y$, i.e.,

$$\mathbb{E}[\hat{b}_t - \hat{A}_t \theta - \hat{M}_t y] = b - A\theta - My,$$
$$\mathbb{E}[\hat{A}_t^\top y] = A^\top y,$$

all have bounded variance, i.e.,

$$\mathbb{E}\big[||\hat{b}_t - \hat{A}_t\theta - \hat{M}_t y - (b - A\theta - My)||^2\big] \leq \sigma_1^2,$$
$$\mathbb{E}\big[||\hat{A}_t^\top y - A^\top y||^2\big] \leq \sigma_2^2, \tag{16}$$

where $\sigma_1$ and $\sigma_2$ are non-negative constants. We further define

$$\sigma^2 = \sigma_1^2 + \sigma_2^2. \tag{17}$$

Assumption 4 also gives us the following "light-tail" assumption. There exist constants $M_{*,\theta}$ and $M_{*,y}$ such that

$$\mathbb{E}[\exp\{\frac{||\hat{b}_t - \hat{A}_t\theta - \hat{M}_t y||^2}{M_{*,\theta}^2}\}] \leq \exp\{1\},$$
$$\mathbb{E}[\exp\{\frac{||\hat{A}_t^\top y||^2}{M_{*,y}^2}\}] \leq \exp\{1\}. \tag{18}$$

This "light-tail" assumption is equivalent to the assumption in Eq. 3.16 in (Nemirovski et al., 2009) and is necessary for the high-probability bound of Proposition 3. We will show how to compute $M_{*,\theta}, M_{*,y}$ in the Appendix.

### 4.3 Finite-Sample Performance Bounds

The finite-sample performance bounds that we derive for the GTD algorithms in this section are for the case that the training set $\mathcal{D}$ has been generated as discussed in Section 2. We further discriminate between the on-policy ($\pi = \pi_b$) and off-policy ($\pi \neq \pi_b$) scenarios. The sampling scheme used to generate $\mathcal{D}$, in which the first state of each tuple, $s_i$, is an i.i.d. sample from a distribution $\xi$, also considered in the original GTD and GTD2 papers is for the analysis of these algorithms, and not used in the experiments (Sutton et al., 2008, 2009). Another scenario that can motivate this sampling scheme is when we are given a set of high-dimensional data generated either in an on-policy or off-policy manner, and $d$ is so large that the value function of the target policy cannot be computed using a least-squares method (that involves matrix inversion), and iterative techniques similar to GTD/GTD2 are required.

We first derive a high-probability bound on the error function of the saddle-point problem (13) at the GTD solution $(\bar{\theta}_n, \bar{y}_n)$. Before stating this result in Proposition 3, we report the following lemma that is used in its proof.

**Lemma 2.** *The induced $\ell_2$-norm of matrix $A$ and the $\ell_2$-norm of vector $b$ are bounded by*

$$||A||_2 \leq (1+\gamma)\rho_{\max} L^2 d, \quad ||b||_2 \leq \rho_{\max} L \mathrm{R}_{\max}.$$

*Proof.* See the Appendix. ■

**Proposition 3.** *Let $(\bar{\theta}_n, \bar{y}_n)$ be the output of the GTD algorithm after $n$ iterations (see Eq. (15)). Then, with probability at least $1-\delta$, we have*

$$\mathrm{Err}(\bar{\theta}_n, \bar{y}_n) \leq \sqrt{\frac{5}{n}}(8 + 2\log\frac{2}{\delta})R^2 \tag{19}$$
$$\times \left(\rho_{\max} L\Big(2(1+\gamma)Ld + \frac{R_{\max}}{R}\Big) + \tau + \frac{\sigma}{R}\right),$$

*where* $\mathrm{Err}(\bar{\theta}_n, \bar{y}_n)$ *is the error function of the saddle-point problem* (13) *defined by Eq.* (12)*,* $R$ *is defined in Assumption 2,* $\sigma$ *is from Eq.* (17)*, and* $\tau = \sigma_{\max}(M)$ *is the largest singular value of* $M$*, which means* $\tau = 1$ *for GTD and* $\tau = \sigma_{\max}(C)$ *for GTD2.*

*Proof.* We give a proof sketch here. The proof of Proposition 3 heavily relies on Proposition 3.2 in (Nemirovski et al., 2009). We just need to map our convex-concave *stochastic* saddle-point problem in Eq. (13), i.e.,

$$\min_{\theta\in\Theta}\max_{y\in Y}\left(L(\theta, y) = \langle b - A\theta, y\rangle - \frac{1}{2}||y||_M^2\right)$$

The details of verifying the conditions are in the Appendix. Note that in (Nemirovski et al., 2009) the robust stochastic approximation technique is used, mainly by combining aggressive step-sizes (i.e., large constant step-sizes) and iterative averaging (Polyak & Juditsky, 1992) (also termed as *Polyak's averaging*). The choice of the constant step-size is described in the Appendix and the iterative averaging is shown in Eq. (15). ■

**Theorem 1.** *Let* $\bar{\theta}_n$ *be the output of the GTD algorithm after* $n$ *iterations (see Eq.* (15)*). Then, with probability at least* $1 - \delta$*, we have*

$$\frac{1}{2}||A\bar{\theta}_n - b||_\xi^2 \leq \tau\xi_{\max}\,\mathrm{Err}(\bar{\theta}_n, \bar{y}_n).$$

*Proof.* From Proposition 1, for any $\theta$, we have

$$\max_y\; L(\theta, y) = \frac{1}{2}||A\theta - b||_{M^{-1}}^2.$$

Given Assumption 3, the system of linear equations $A\theta = b$ has a solution $\theta^*$, i.e., the (off-policy) fixed-point $\theta^*$ exists, and thus, we may write

$$\begin{aligned}\min_\theta\;\max_y\; L(\theta, y) &= \min_\theta\;\frac{1}{2}||A\theta - b||_{M^{-1}}^2\\ &= \frac{1}{2}||A\theta^* - b||_{M^{-1}}^2 = 0.\end{aligned}$$

In this case, we also have[2]

$$\begin{aligned}\min_\theta\; L(\theta, y) \leq \max_y\;\min_\theta\; L(\theta, y) &\leq \min_\theta\;\max_y\; L(\theta, y)\\ &= \frac{1}{2}||A\theta^* - b||_{M^{-1}}^2 = 0.\end{aligned} \tag{20}$$

From Eq. (20), for any $(\theta, y) \in \Theta \times Y$ including $(\bar{\theta}_n, \bar{y}_n)$, we may write

$$\begin{aligned}\mathrm{Err}(\bar{\theta}_n, \bar{y}_n) &= \max_y\; L(\bar{\theta}_n, y) - \min_\theta\; L(\theta, \bar{y}_n)\\ &\geq \max_y\; L(\bar{\theta}_n, y) = \frac{1}{2}||A\bar{\theta}_n - b||_{M^{-1}}^2.\end{aligned}$$

2. We may write the second inequality as an equality for our saddle-point problem defined by Eq. (13).

Since $||A\bar{\theta}_n - b||_\xi^2 \leq \tau\xi_{\max} ||A\bar{\theta}_n - b||_{M^{-1}}^2$, where $\tau$ is the largest singular value of $M$, we have

$$\frac{1}{2}||A\bar{\theta}_n - b||_\xi^2 \leq \frac{\tau\xi_{\max}}{2}||A\bar{\theta}_n - b||_{M^{-1}}^2 \leq \tau\xi_{\max} \operatorname{Err}(\bar{\theta}_n, \bar{y}_n). \tag{21}$$

The proof follows by combining Eq. (21) and Proposition (3). ■

With the results of Proposition 3 and Theorem 1, we are now ready to derive finite-sample bounds on the performance of GTD/GTD2 in both on-policy and off-policy settings.

#### 4.3.1 ON-POLICY PERFORMANCE BOUND

In this section, we consider the on-policy setting in which the behavior and target policies are equal, i.e., $\pi_b = \pi$, and the sampling distribution $\xi$ is the stationary distribution of the target policy $\pi$ (and the behavior policy $\pi_b$). We use Lemma 3 to derive our on-policy bound. The proof of this lemma can be found in (Geist, Scherrer, Lazaric, & Ghavamzadeh, 2012).

**Lemma 3.** *For any parameter vector $\theta$ and corresponding $\hat{v} = \Phi\theta$, the following equality holds*

$$V - \hat{v} = (I - \gamma\Pi P)^{-1}\left[(V - \Pi V) + \Phi C^{-1}(b - A\theta)\right].$$

Using Lemma 3, we derive the following performance bound for GTD/GTD2 in the on-policy setting.

**Proposition 4.** *Let $V$ be the value of the target policy and $\bar{v}_n = \Phi\bar{\theta}_n$, where $\bar{\theta}_n$ defined by* (15)*, be the value function returned by on-policy GTD/GTD2. Then, with probability at least $1 - \delta$, we have*

$$||V - \bar{v}_n||_\xi \leq \frac{1}{1-\gamma}\left(||V - \Pi V||_\xi + \frac{L}{\nu}\sqrt{2d\tau\xi_{\max}\operatorname{Err}(\bar{\theta}_n, \bar{y}_n)}\right)$$

*where $\operatorname{Err}(\bar{\theta}_n, \bar{y}_n)$ is upper-bounded by Eq.* (19) *in Proposition 3, with $\rho_{\max} = 1$ (on-policy setting).*

*Proof.* See the Appendix. ■

**Remark:** It is important to note that Proposition 4 shows that the error in the performance of the GTD/GTD2 algorithm in the on-policy setting is of $O\left(\frac{L^2 d\sqrt{\tau\xi_{\max}\log\frac{1}{\delta}}}{n^{1/4}\nu}\right)$. Also note that the term $\frac{\tau}{\nu}$ in the GTD2 bound is the conditioning number of the covariance matrix $C$.

#### 4.3.2 OFF-POLICY PERFORMANCE BOUND

In this section, we consider the off-policy setting in which the behavior and target policies are different, i.e., $\pi_b \neq \pi$, and the sampling distribution $\xi$ is the stationary distribution of the behavior policy $\pi_b$. We assume that off-policy fixed-point solution exists, i.e., there exists a $\theta^*$ satisfying $A\theta^* = b$. Note that this is a direct consequence of Assumption 3 in which we assumed that the matrix $A$ in the off-policy setting is non-singular. We use Lemma 4 to derive our off-policy bound. The proof of this lemma can be found in (Kolter, 2011). Note that $\kappa(\bar{D})$ in his proof is equal to $\sqrt{\rho_{\max}}$ in our paper.

**Lemma 4.** *If $\Xi$ satisfies the following linear matrix inequality*

$$\begin{bmatrix} \Phi^\top \Xi \Phi & \Phi^\top \Xi P \Phi \\ \Phi^\top P^\top \Xi \Phi & \Phi^\top \Xi \Phi \end{bmatrix} \succeq 0 \tag{22}$$

*and let $\theta^*$ be the solution to $A\theta^* = b$, then we have*

$$||V - \Phi\theta^*||_\xi \leq \frac{1+\gamma\sqrt{\rho_{\max}}}{1-\gamma}||V - \Pi V||_\xi. \tag{23}$$

Note that the condition on $\Xi$ in Eq. (22) guarantees that the behavior and target policies are not too far away from each other. Using Lemma 4, we derive the following performance bound for GTD/GTD2 in the off-policy setting.

**Proposition 5.** *Let $V$ be the value of the target policy and $\bar{v}_n = \Phi\bar{\theta}_n$, where $\bar{\theta}_n$ is defined by* (15)*, be the value function returned by off-policy GTD/GTD2. Also let the sampling distribution $\Xi$ satisfy the condition in Eq.* (22)*. Then, with probability at least $1-\delta$, we have*

$$\begin{aligned} ||V - \bar{v}_n||_\xi \leq & \frac{1+\gamma\sqrt{\rho_{\max}}}{1-\gamma}||V - \Pi V||_\xi \\ & + \sqrt{\frac{2\tau_C \tau \xi_{\max}}{\sigma_{\min}(A^\top M^{-1} A)} \mathrm{Err}(\bar{\theta}_n, \bar{y}_n)}, \end{aligned} \tag{24}$$

*where $\tau_C = \sigma_{\max}(C)$.*

*Proof.* See the Appendix. ■

### 4.4 Accelerated Algorithm

As discussed at the beginning of Section 3, this saddle-point formulation not only gives us the opportunity to use the techniques for the analysis of SG methods to derive finite-sample performance bounds for the GTD algorithms, as we showed in Section 4, but it also allows us to use the powerful algorithms that have been recently developed to solve the SG problems and derive more efficient versions of GTD and GTD2. Stochastic Mirror-Prox (SMP) (Juditsky et al., 2008) is an "almost dimension-free" non-Euclidean extra-gradient method that deals with both smooth and non-smooth stochastic optimization problems (see (Juditsky & Nemirovski, 2011) and (Bubeck, 2014) for more details). Using SMP, we propose a new version of GTD/GTD2, called GTD-MP/GTD2-MP, with the following update formula:[3]

$$\begin{aligned} y_t^m &= y_t + \alpha_t(\hat{b}_t - \hat{A}_t\theta_t - \hat{M}_t y_t), \qquad \theta_t^m = \theta_t + \alpha_t \hat{A}_t^\top y_t, \\ y_{t+1} &= y_t + \alpha_t(\hat{b}_t - \hat{A}_t\theta_t^m - \hat{M}_t y_t^m), \ \theta_{t+1} = \theta_t + \alpha_t \hat{A}_t^\top y_t^m. \end{aligned}$$

After $T$ iterations, these algorithms return $\bar{\theta}_T := \frac{\sum_{t=1}^T \alpha_t \theta_t}{\sum_{t=1}^T \alpha_t}$ and $\bar{y}_T := \frac{\sum_{t=1}^T \alpha_t y_t}{\sum_{t=1}^T \alpha_t}$. The details of the algorithm are shown in Algorithm 2, and the experimental comparison study between GTD2 and GTD2-MP is reported in Section 7.

3. For simplicity, we only describe mirror-prox GTD methods where the mirror map is identity, which can also be viewed as extragradient (EG) GTD methods. (Mahadevan et al., 2014) gives a more detailed discussion of a broad range of mirror maps in RL.

**Algorithm 2** GTD2-MP

1: **for** $t = 1, \ldots, n$ **do**
2: Update parameters

$$\begin{aligned}
\delta_t(\theta_t) &= r_t - \theta_t^\top \Delta\phi_t \\
y_t^m &= y_t + \alpha_t(\rho_t\delta_t - \phi_t^\top y_t)\phi_t \\
\theta_t^m &= \theta_t + \alpha_t\rho_t\Delta\phi_t(\phi_t^\top y_t) \\
\delta_t^m(\theta_t^m) &= r_t - (\theta_t^m)^\top \Delta\phi_t \\
y_{t+1} &= y_t + \alpha_t(\rho_t\delta_t^m - \phi_t^\top y_t^m)\phi_t \\
\theta_{t+1} &= \theta_t + \alpha_t\rho_t\Delta\phi_t(\phi_t^\top y_t^m)
\end{aligned}$$

3: **end for**
4: OUTPUT

$$\bar{\theta}_n := \frac{\sum_{t=1}^n \alpha_t\theta_t}{\sum_{t=1}^n \alpha_t} \quad , \quad \bar{y}_n := \frac{\sum_{t=1}^n \alpha_t y_t}{\sum_{t=1}^n \alpha_t}$$

## 5. Further Analysis

In this section, we discuss different aspects of the proximal gradient TD framework from several perspectives, such as acceleration, learning with inexact importance weight factor $\rho_t$, finite-sample analysis with Markov sampling condition, and discussion of TDC algorithm.

### 5.1 Acceleration Analysis

In this section, we are going to discuss the convergence rate of the accelerated algorithms using off-the-shelf accelerated solvers for saddle-point problems. For simplicity, we will discuss the error bound of $\frac{1}{2}||A\theta - b||^2_{M^{-1}}$, and the corresponding error bound of $\frac{1}{2}||A\theta - b||^2_\xi$ and $\|V - \bar{v}_n||_\xi$ can be likewise derived. As can be seen from the above analysis, the convergence rate of the GTD algorithms family is

$$(\mathbf{GTD/GTD2}): \quad O\left(\frac{\tau + ||A||_2 + \sigma}{\sqrt{n}}\right)$$

In this section, we raise an interesting question: what is the "optimal" GTD algorithm? To answer this question, we review the convex-concave formulation of GTD2. According to convex programming complexity theory (Juditsky et al., 2008), the un-improvable convergence rate of the stochastic saddle-point problem in Eq. (13) is

$$(\mathbf{Optimal}): \quad O\left(\frac{\tau}{n^2} + \frac{||A||_2}{n} + \frac{\sigma}{\sqrt{n}}\right)$$

There are many readily available stochastic saddle-point solvers, such as the stochastic Mirror-Prox (SMP) (Juditsky et al., 2008) algorithm, which leads to our proposed GTD2-MP algorithm. GTD2-

MP is able to accelerate the convergence rate of our gradient TD method to:

$$(\mathbf{GTD2-MP}): \quad O\left(\frac{\tau+||A||_2}{n}+\frac{\sigma}{\sqrt{n}}\right).$$

### 5.2 Learning with Biased $\rho_t$

The importance weight factor $\rho_t$ is lower bounded by 0, but yet may have an arbitrarily large upper bound. In real applications, the importance weight factor $\rho_t$ may not be estimated exactly, i.e., the estimation $\hat{\rho}_t$ is a biased estimation of the true $\rho_t$. To this end, the stochastic gradient we obtained is not the unbiased gradient of $L(\theta, y)$ anymore. This falls into a broad category of learning with inexact stochastic gradient, or termed as stochastic gradient methods with an inexact oracle (Devolder, 2011). Given the inexact stochastic gradient, the convergence rate and performance bound become much worse than the results with exact stochastic gradient. Based on the analysis by (Juditsky et al., 2008), we have the error bound for inexact estimation of $\rho_t$.

**Proposition 6.** *Let $\bar{\theta}_n$ be defined as above. Assume at the $t$-th iteration, $\hat{\rho}_t$ is the estimation of the importance weight factor $\rho_t$ with bounded bias such that $\mathbb{E}[\hat{\rho}_t - \rho_t] \leq \epsilon$. The convergence rates of GTD/GTD2 algorithms with iterative averaging are as follows,*

$$||A\bar{\theta}_n - b||^2_{M^{-1}} \leq O\left(\frac{\tau+||A||_2+\sigma}{\sqrt{n}}\right) + O(\epsilon)$$

This implies that the inexact estimation of $\rho_t$ may cause disastrous estimation error, which implies that an exact estimation of $\rho_t$ is very important.

### 5.3 Finite-Sample Analysis of Online Learning

Another more challenging scenario is the online learning scenario, where the samples are interactively generated by the environment, or by an interactive agent. The difficulty lies in that the sample distribution does not follow the i.i.d sampling condition anymore, but follows an underlying Markov chain $\mathcal{M}$. If the Markov chain $\mathcal{M}$'s mixing time is small enough, i.e., the sample distribution reduces to the stationary distribution of $\pi_b$ very fast, our analysis still applies. However, it is usually the case that the underlying Markov chain's mixing time $\tau_{\text{mix}}$ is not small enough. The analysis can be conducted by extending the result of recent work (Duchi, Agarwal, Johansson, & Jordan, 2012) from strongly convex loss functions to saddle-point problems. Following this line of research, Wang et al. (Wang, Chen, Liu, Ma, & Liu, 2017) conducted the finite-sample analysis of GTD2(0) algorithms in the Markov noise setting, which is the same in convergence rate order but different in the constant factors.

### 5.4 Discussion of TDC Algorithm

Now we discuss the limitation of our analysis with regard to the temporal difference with correction (TDC) algorithm (Sutton et al., 2009). Interestingly, the TDC algorithm seems not to have an explicit saddle-point representation, since it incorporates the information of the optimal $y_t^*(\theta_t)$ into the update of $\theta_t$, a quasi-stationary condition which is commonly used in two-time-scale stochastic approximation approaches. An intuitive answer to the advantage of TDC over GTD2 is that the TDC update of $\theta_t$ can be considered as incorporating the prior knowledge into

the update rule: for a stationary $\theta_t$, if the optimal $y_t^*(\theta_t)$ has a closed-form solution or is easy to compute, then incorporating this $y_t^*(\theta_t)$ into the update law tends to accelerate the algorithm's convergence performance. For the GTD2 update, note that there is a sum of two terms where $y_t$ appears, which are $\rho_t(\phi_t - \gamma\phi_t')(y_t^\top \phi_t) = \rho_t\phi_t(y_t^\top \phi_t) - \gamma\rho_t\phi_t'(y_t^\top \phi_t)$. Replacing $y_t$ in the first term with $y_t^*(\theta_t) = \mathbb{E}[\phi_t\phi_t^\top]^{-1}\mathbb{E}[\rho_t\delta_t(\theta_t)\phi_t]$, we have the TDC update rule. There are two key factors that impedes the finite-sample analysis of TDC algorithm with the saddle-point approach. Firstly, in contrast to GTD/GTD2, TDC is a two-time scale algorithm where $\lim_{t\to\infty} \frac{\alpha_t}{\beta_t} = 0$. Secondly, note that TDC does not minimize *any* objective functions, thus does not have a stochastic primal-dual formulation as GTD and GTD2, and the asymptotic convergence of TDC requires more restrictions than GTD2 as shown by (Sutton et al., 2009).

### 5.5 Recent Results of Related Work

The proximal gradient temporal difference learning framework introduces many stochastic optimization techniques that facilitate theoretical analysis of reinforcement learning algorithms, such as Polyak's iterative averaging, projections, and constant stepsizes, which were first introduced by Liu et al. (2012). We briefly review several significant research advances on the finite-sample analysis of linear temporal difference learning algorithms since the finite-sample analysis of GTD algorithms which was first published (Liu, Liu, Ghavamzadeh, Mahadevan, & Petrik, 2015). The first line of work is the analysis of GTD algorithm family (Dalal, Szörényi, Thoppe, & Mannor, 2018a; Dalal, Thoppe, Szorenyi, & Mannor, 2018b) with different learning settings. Dalal et al. (2018b) conducted a finite-sample analysis of the two-time-scale GTD, GTD2, and TDC algorithms using a concentration bound for stochastic approximation methods via Alekseev's Formula (Kamal, 2010; Thoppe & Borkar, 2015). Using this approach, the convergence rate of GTD2 w.r.t *mean-square error* (MSE) $||V - \hat{v}_n||_\xi^2$ proposed in (Dalal et al., 2018b) is $\tilde{O}(n^{-(1-\chi)\frac{2}{3}})$, where $\chi$ is a tuning parameter that influences the stepsizes used by the algorithms, and a special "sparse projection" is used (Dalal et al., 2018a). This framework enables the finite-sample analysis of the TDC algorithm, which cannot be analyzed from the saddle-point perspective, as explained in Section 5.4. The other work (Wang et al., 2017) investigates the convergence rate assuming a Markov sampling condition and Robbins-Monro stepsizes. Table 1 presents a comparison of existing approaches. It should be noted that all of the analyses are for algorithms employing projections and iterative averaging. Lakshminarayanan et al. (Lakshminarayanan & Szepesvari, 2018) also studied the impact of constant stepsizes and Polyak's iterative averaging on the TD algorithm with projections in the i.i.d setting, yet the result has not been extended to GTD algorithm family yet.

The second line of work aims to adopt new stochastic saddle-point solvers into the proximal gradient TD framework and proposes new algorithms for acceleration, regularization and variance reduction. Mahadevan et al. (2014) investigated proximal gradient TD with an $\ell_1$-regularizer to enhance sparsity. Du et al. (Du, Chen, Li, Xiao, & Zhou, 2017) introduced the stochastic variance reduced saddle-point solver (Palaniappan & Bach, 2016) to reach a linear convergence rate for a fixed set of samples. The third line of work focuses on the analysis of gradient temporal difference learning algorithms with eligibility traces (Yu, 2017). To the best of our knowledge, the only asymptotic convergence analysis for GTD($lambda$) with $\lambda > 0$ is proposed by Yu (2017), and the finite-sample analysis still remains an open problem. Comprehensive studies of different temporal difference learning algorithms with eligibility traces have been conducted in (Dann, Neumann, & Peters, 2014; White & White, 2016).

| **Method** | **GTD2** (Liu et al., 2015) | **GTD2** (Wang et al., 2017) | **GTD2** (Dalal et al., 2018b) |
|---|---|---|---|
| **Sampling** | i.i.d | Markov | i.i.d |
| **Stepsize's time-scale** | single-time-scale | single-time-scale | two-time-scale |
| **Stepsize choice** | constant | Robbins-Monro | $\alpha_n = 1/n^{1-\chi}$, $\beta_n = 1/n^{(1-\chi)\frac{2}{3}}$ |
| **Projection** | Y | Y | Y |
| **Iterative averaging** | Y | Y | Y |
| **Conv. rate of MSE** | $\tilde{O}(1/\sqrt{n})$ | $\tilde{O}(\frac{\sum_{t=1}^n \alpha_t^2}{\sum_{t=1}^n \alpha_t})$ | $\tilde{O}(n^{-2(1-\chi)/3})$ |

Table 1: Comparison of Existing Methods

## 6. Control Learning Extension

In this section, we are going to discuss the control learning extension of the family of proximal gradient algorithms. To this end, we will first present a lemma bridging the connection between the forward-view and backward-view perspectives. Then based on the GQ (Maei & Sutton, 2010) algorithm, we propose the control learning extension of the GTD2-MP algorithm, which is termed as the GQ-MP algorithm.

### 6.1 Extension to Eligibility Trace

The $T$ operator looks one-step ahead, but it would be beneficial to look multiple steps ahead (Sutton & Barto, 1998), which gives rise to the *multiple-step Bellman operator* $T^\lambda$, otherwise known as the $\lambda$-weighted Bellman operator (Sutton & Barto, 1998), which is an arithmetic mean of the power series of $T$, i.e.,

$$T^\lambda = (1-\lambda)\sum\nolimits_{i=0}^{\infty} \lambda^i T^{i+1}, \lambda \in (0,1)$$

Correspondingly, the *multiple-step TD error*, also termed as the $\lambda$-TD error $\delta^\lambda$ w.r.t $\theta$ is defined as

$$\mathbb{E}[\delta^\lambda(\theta)] = T^\lambda \hat{v} - \hat{v} = T^\lambda \Phi\theta - \Phi\theta$$

The objective function as in Eq. (6) is changed accordingly as follows by replacing $T$ with $T^\lambda$ ,

$$\mathrm{J}(\theta) = ||\Phi^\top \Xi(T^\lambda \hat{v} - \hat{v})||^2_{M^{-1}} = ||\mathbb{E}[\rho_i \phi_i \delta_i^\lambda(\theta)]||^2_{M^{-1}} \tag{25}$$

This is called the *forward view* since it calls for looking multiple steps ahead, which is difficult to implement in practice. To this end, the *backward view* using eligibility traces is easy to implement. The eligibility trace is defined in a recursive way as

$$\begin{aligned} e_0 =& 0 \\ e_t =& \rho_t \gamma \lambda e_{t-1} + \phi_t \end{aligned}$$

We will introduce Theorem 2 to bridge the gap between the backward and forward view.

**Theorem 2.** *(Maei, 2011; Geist & Scherrer, 2014) There is an equivalence between the forward view and backward view such that*

$$\mathbb{E}[\phi_i \delta_i^\lambda(\theta)] = \mathbb{E}[e_i \delta_i(\theta)] \tag{26}$$

The details of the forward view and the backward view can be seen in (Sutton & Barto, 1998), Theorem 11 in (Maei, 2011), and Proposition 6 in (Geist & Scherrer, 2014). A natural extension to Eq. (26) multiplies the importance ratio factor on both sides of the equality as follows,

$$\mathbb{E}[\rho_i \phi_i \delta_i^{\lambda}(\theta)] = \mathbb{E}[\rho_i e_i \delta_i(\theta)]$$

### 6.2 Greedy-GQ($\lambda$) Algorithm

With the help of Theorem 2, we can convert the objective formulation in Eq. (25) to

$$J(\theta) = ||\mathbb{E}[\rho_i e_i \delta_i(\theta)]||^2_{M^{-1}}.$$

The corresponding primal-dual formulation is

$$J(\theta) = \max_y \left( \langle \mathbb{E}[\rho_i e_i \delta_i(\theta)], y \rangle - \frac{1}{2}||y||^2_M \right)$$

and thus the new algorithm can be derived as

$$\begin{aligned} \theta_{t+1} =& \theta_t + \alpha_t \rho_t \Delta\phi_t (e_t^\top y_t) \\ y_{t+1} =& y_t + \alpha_t \left(\rho_t \delta_t e_t - M_t y_t\right). \end{aligned}$$

Correspondingly, Greedy-GQ($\lambda$) with importance sampling can be derived. Here we present the Greedy-GQ($\lambda$) algorithm. and **GQ-MP-LEARN** algorithm, which is the core step of Greedy-GQ

**Algorithm 3** Greedy-GQ($\lambda$)

Initialize $e_t = 0$, starting from $s_0$.

1: **repeat**
2: Take $a_t$ according to $\pi_b$, and arrive at $s_{t+1}$
3: Compute $a_t^* = \arg\max_a \theta^\top \phi(s_t, a)$. If $a_t = a_t^*$, then $\rho_t = \frac{1}{\pi_b(a_t|s_t)}$; otherwise $\rho_t = 0$.
4: Compute $\theta_{t+1}, y_{t+1}$ according to **GQ-MP-LEARN** Algorithm.
5: Choose action $a_t$, and get $s_{t+1}, r_{t+1}$
6: Set $t \leftarrow t + 1$;
7: **until** $s_t$ is an absorbing state;
8: Compute $\bar{\theta}_t, \bar{y}_t$

algorithm.

## 7. Empirical Evaluation

In this section, we compare the previous GTD2 method with our proposed GTD2-MP method using various domains with regard to their value function approximation performance.

**Algorithm 4** GQ-MP-LEARN

$$
\begin{aligned}
e_t &= \gamma\lambda\rho_t e_{t-1} + \phi_t \\
\delta_t &= r_t + \theta_t^\top \Delta\phi_t \\
y_t^m &= y_t + \alpha_t \left(\rho_t e_t \delta_t - (\phi_t^\top y_t)\phi_t\right) \\
\theta_t^m &= \theta_t + \alpha_t \rho_t \Delta\phi_t (e_t^\top y_t) \\
\delta_t^m &= r_t + \theta_t^{m\top} \Delta\phi_t \\
y_{t+1} &= y_t + \alpha_t \left(\rho_t e_t \delta_t^m - (\phi_t^\top y_t^m)\phi_t\right) \\
\theta_t^m &= \theta_t + \alpha_t \rho_t \Delta\phi_t (e_t^\top y_t^m)
\end{aligned}
$$

### 7.1 Baird Domain

The Baird example (Baird, 1995) is a well-known example to test the performance of off-policy convergent algorithms. Constant stepsizes $\alpha = 0.005$ for GTD2 and $\alpha = 0.004$ for GTD2-MP are chosen via comparison studies as in (Dann et al., 2014). Figure 1 shows the MSPBE curve of GTD2, GTD2-MP of $8000$ steps averaged over $200$ runs. We can see that GTD2-MP gives a significant improvement over the GTD2 algorithm wherein both the MSPBE and variance are substantially reduced.

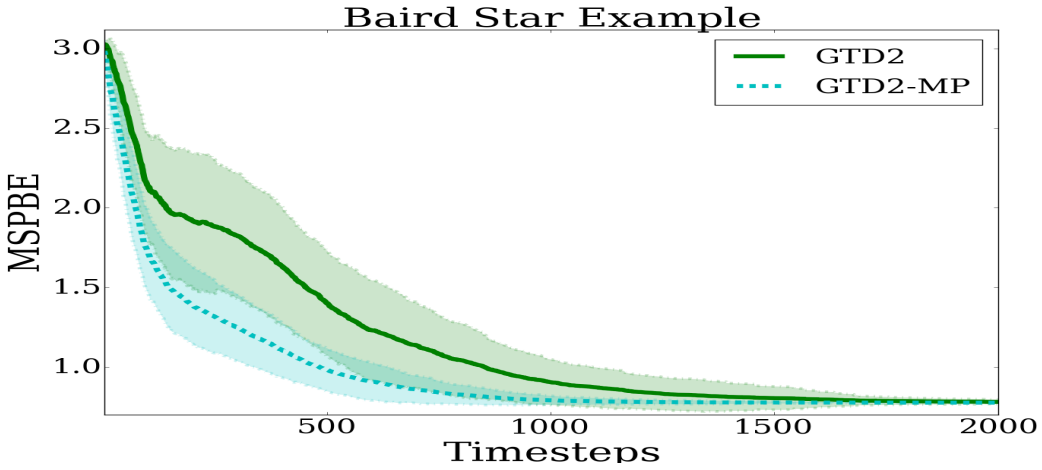


Figure 1: Off-Policy Convergence Comparison

### 7.2 50-State Chain Domain

The 50 state chain (Lagoudakis & Parr, 2003) is a standard MDP domain. There are 50 discrete states $\{s_i\}_{i=1}^{50}$ and two actions moving the agent left $s_i \to s_{\max(i-1,1)}$ and right $s_i \to s_{\min(i+1,50)}$. The actions succeed with probability $0.9$; failed actions move the agent in the opposite direction.

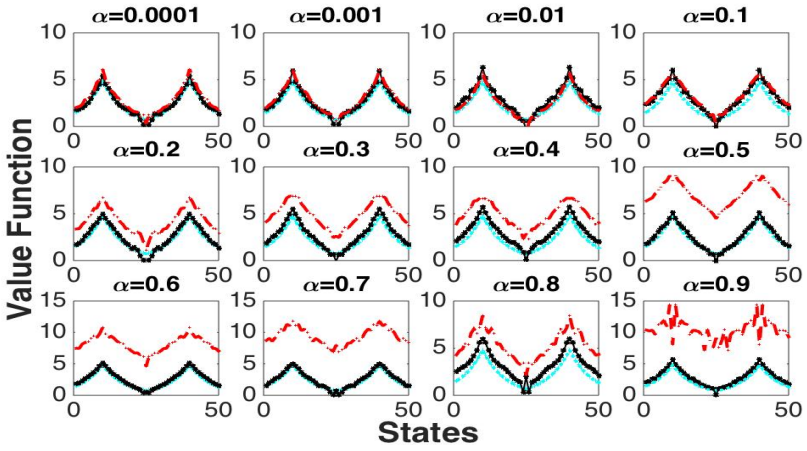


Figure 2: Chain Domain

The discount factor is $\gamma = 0.9$. The agent receives a reward of $+1$ when in states $s_{10}$ and $s_{41}$. All other states have a reward of 0. In this experiment, we compare the performance of the value approximation w.r.t different stepsizes $\alpha = 0.0001, 0.001, 0.01, 0.1, 0.2, \cdots, 0.9$ using the BEBF basis (Parr, Painter-Wakefield, Li, & Littman, 2007). Figure 2 shows the value function approximation result where the cyan curve is the true value function, the red dashed curve is the GTD result, and the black curve is the GTD2-MP result. From the figure, one can see that GTD2-MP is much more robust w.r.t. stepsize choice than the GTD2 algorithm.

### 7.3 Energy Management Domain

In this experiment, we compare the performance of the algorithms on an energy management domain. The decision maker must decide how much energy to purchase or sell subject to stochastic prices. This problem is relevant in the context of utilities as well as in settings such as hybrid vehicles. The prices are generated by a Markov chain process. The amount of available storage is limited and degrades with use. The degradation process is based on the physical properties of lithium-ion batteries and discourages fully charging or discharging the battery. The energy arbitrage problem is closely related to the broad class of inventory management problems, with the storage level corresponding to the inventory. However, there are no known results describing the structure of optimal threshold policies in energy storage.

Note that since this is an off-policy evaluation problem, the formulated $A\theta = b$ does not have a solution, and thus the optimal MSPBE($\theta^*$) (resp. MSBE($\theta^*$) ) does not reduce to 0. The result is averaged over 200 runs, and $\alpha = 0.001$ for both GTD2 and GTD2-MP is chosen via comparison studies for each algorithm. As can be seen from Figure 3, GTD2-MP performs much better than GTD2 in the transient state. Then after reaching the steady state, as can be seen from Table 2, we can see that GTD2-MP reaches a better steady-state solution than the GTD algorithm. From the steady-state reported in Table 2, we can see that GTD-MP and GTD2-MP usually reach a far better final solution than TD and GTD/GTD2 algorithms.

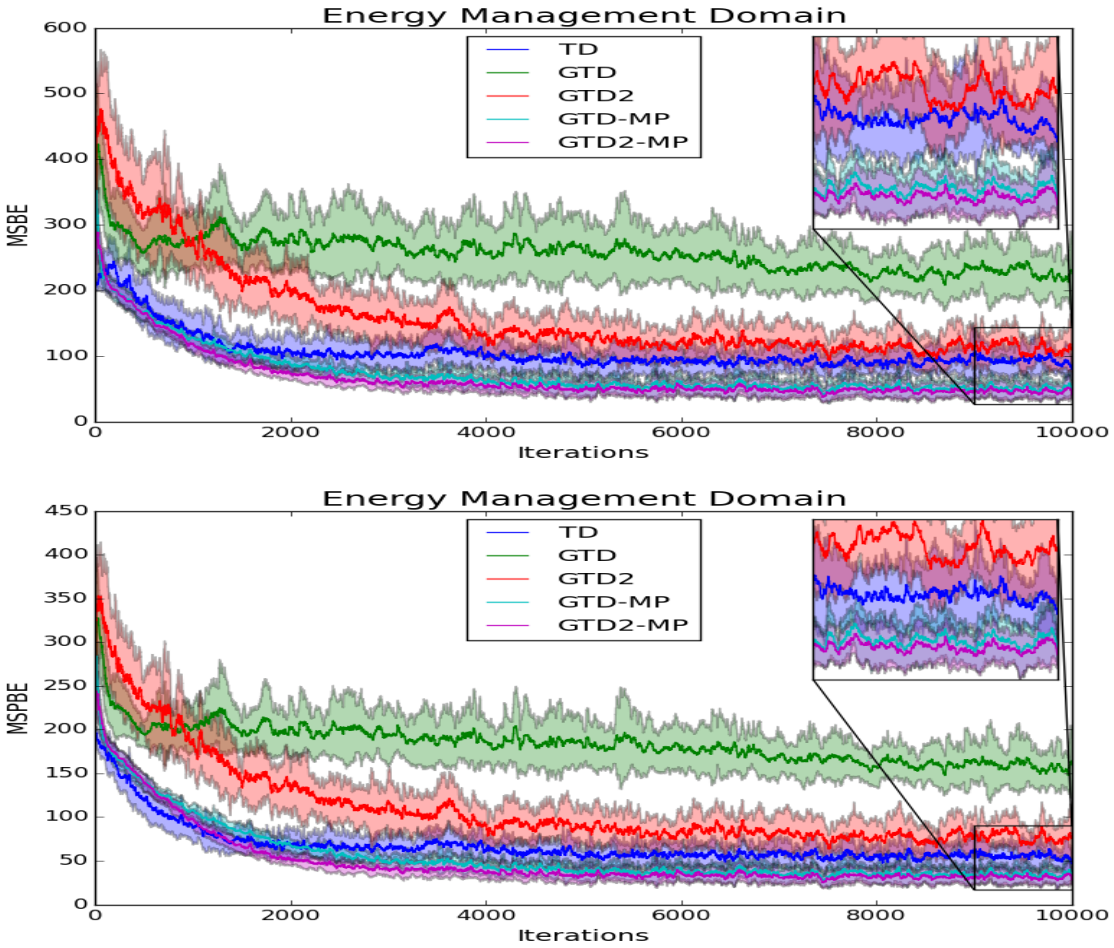


Figure 3: Energy Management Example

| Algorithm | MSPBE | MSBE |
|---|---|---|
| TD | 46.743 | 80.050 |
| GTD | 164.378 | 231.569 |
| GTD2 | 77.139 | 111.19 |
| GTD-MP | 30.170 | 44.627 |
| GTD2-MP | **27.891** | **41.028** |

Table 2: Steady State Performance Comparison of Battery Management Domain

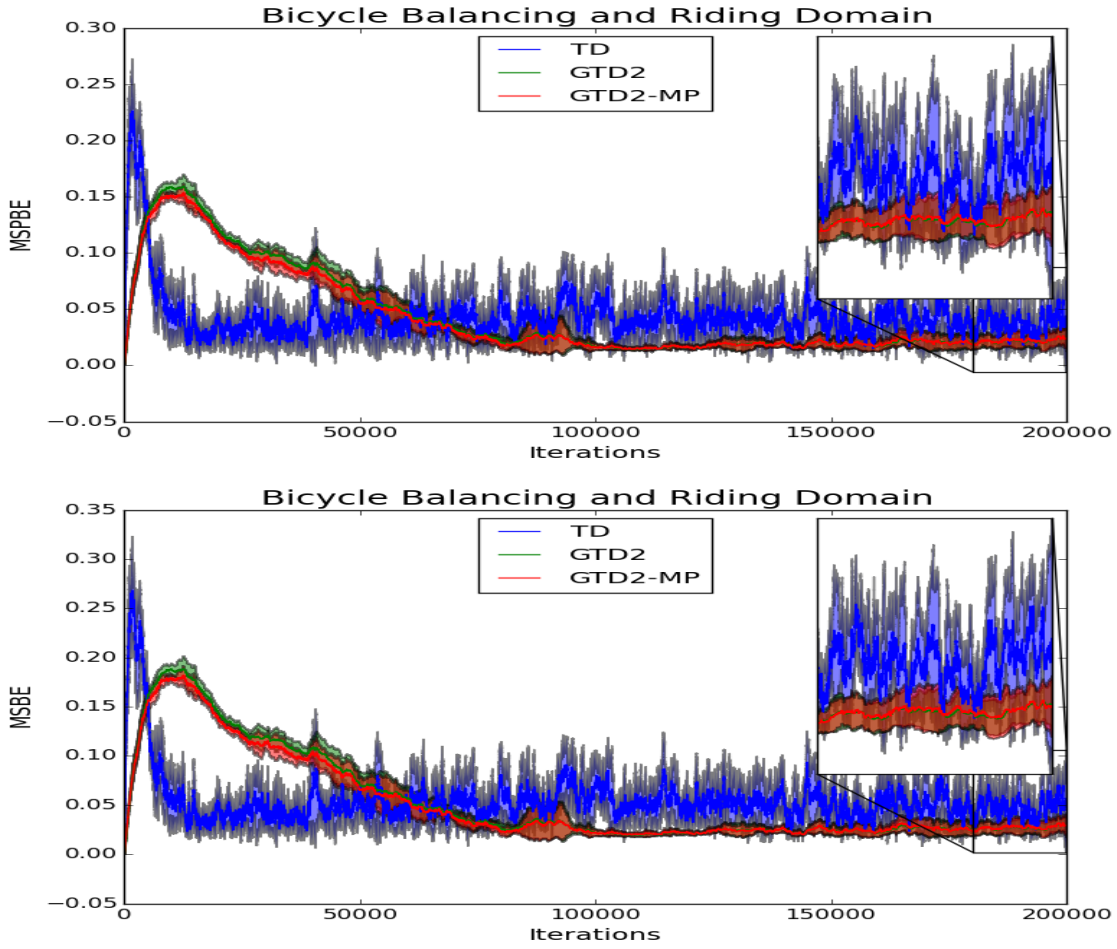


Figure 4: Energy Management Example

### 7.4 Bicycle Balancing and Riding Task

The bicycle balancing and riding domain (Randløv & Alstrøm, 1998) is a complicated domain. The goal is to learn to balance and ride a bicycle to a target position from the starting location.

To make a fair comparison, the parameter settings are identical to the parameter settings in (Lagoudakis & Parr, 2003). The samples are generated via the random walk, after which, we compare the value function approximation results of TD, GTD2, and GTD2-MP algorithm. From Figure 4, we can see that both the GTD2 and GTD2-MP algorithms reach a much better learning curve than the TD algorithm with significantly reduced variance. Besides, the GTD2 and GTD2-MP algorithms reach better steady-state solutions than the TD algorithm, as shown in Table 3.

### 7.5 Comparison with Other First-Order Policy Evaluation Algorithms

Here we give an experimental comparison between the gradient-based TD algorithms and the TD algorithm. Based on the experimental results shown above, we make the following empirical conclusions:

| Algorithm | MSPBE | MSBE |
|---|---|---|
| TD | 0.0423 | 0.0547 |
| GTD2 | 0.0244 | 0.0300 |
| GTD2-MP | **0.0238** | **0.0297** |

Table 3: Steady State Performance Comparison of Bicycle Domain

- Of all the gradient-based algorithms, GTD2-MP is the clear winner.
- For small and medium scale problems, TD is an ideal choice as it converges faster at the initial stage. On the other hand, GTD2-MP often reaches a better steady-state solution given more number of iterations.
- For large-scale problems, GTD2-MP is the clear winner over the TD method with both reduced variance and better final solution, as shown in the bicycle and energy management domain.
- There exist some domains where the $T$ operator is not differentiable, and thus only TD-based algorithms can be applied, such as the optimal stopping problem in (Choi & Van Roy, 2006).

Figure 5: Summary of Comparisons between TD and GTD algorithm family

### 7.6 Energy Management Domain (Revisited): Control Learning

Here, we compare the TD, TDC, and GTD-MP variants of GQ-Learning on the battery management domain. Using the same domain settings as in (Liu et al., 2015; Liu, Liu, Ghavamzadeh, Mahadevan, & Petrik, 2016), we train the three methods on a uniformly random behavior policy for $7,000$ iterations. We then evaluate the policy, $\theta_t$, learned by each algorithm every $100$ time steps by computing the total reward accumulated by following that policy for $10,000$ iterations, averaged over $10$ runs (see Figure 6).

All methods were run with $\theta_0$ initialized to the zeros vector. The TD variant was run with a step size of .0001 to avoid divergence of MSBE while the TDC and GTD-MP variants remained stable in terms of MSBE with a step size of $0.001$ (see Figure 7).

## 8. Summary

In this paper, we showed how gradient TD methods can be shown to be true stochastic gradient methods with respect to a saddle-point primal-dual objective function, which paved the way for the finite-sample analysis of off-policy convergent gradient-based temporal difference learning algorithms such as GTD and GTD2. Both error bound and performance bound are provided, which shows that the value function approximation bound of the GTD algorithms family is $O\left(\frac{d}{n^{1/4}}\right)$. Furthermore, two revised algorithms, namely the projected GTD2 algorithm and the accelerated GTD2-MP algorithm, are proposed.

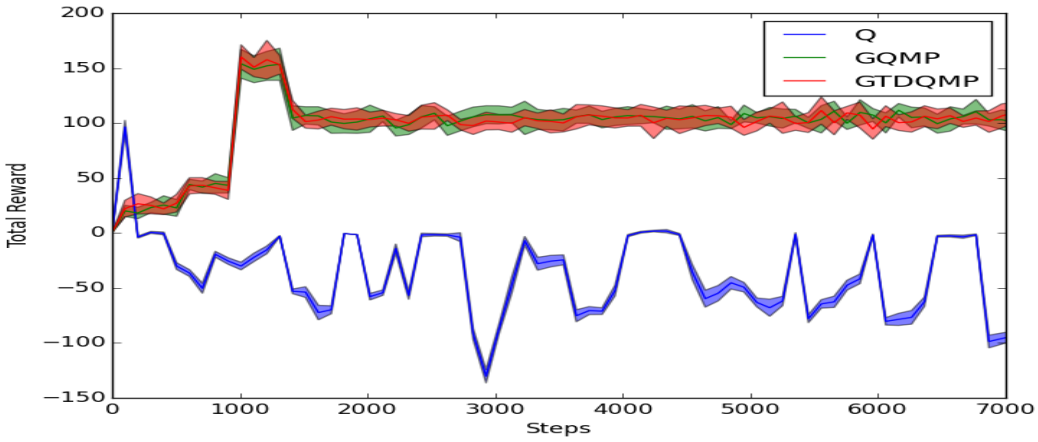


Figure 6: Average total accumulated reward for learned policies at step $t$. Shaded regions around mean total reward denote 1 standard deviation.

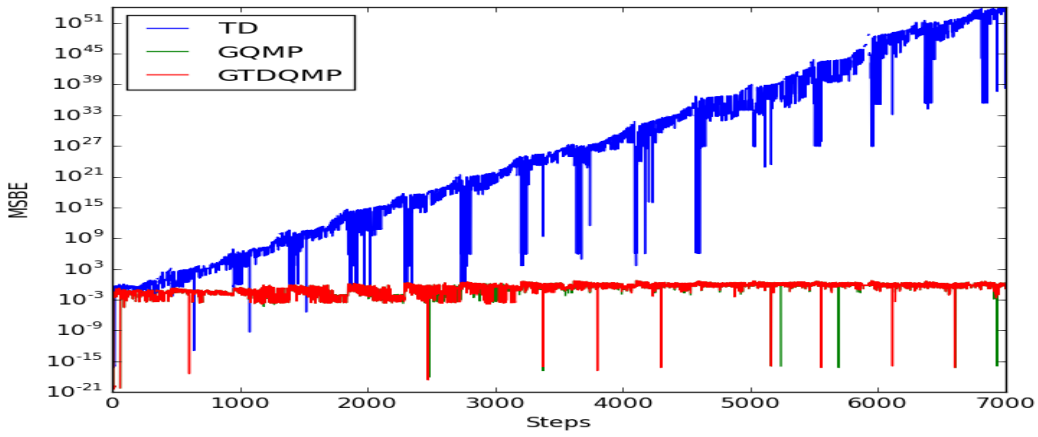


Figure 7: MSBE diverges for TD variant with step size of $0.001$. MSBE is plotted on a log scale.

There are many interesting directions for future research. Our framework can be easily used to design regularized sparse gradient off-policy TD methods.

There are several promising future research directions with our proposed proximal gradient TD learning framework. The first promising direction is to explore other compound operator splitting techniques other than primal-dual splitting. As we have shown in previous chapters, new algorithms can be designed if there exist methods that can split the operator so that the product of expectations can be avoided, and this operator splitting formulation does not have to be the primal-dual formulation. We have explored two primal-dual formulations, one is based on the convex conjugate function, and the other is based on dual norm representation. It would be interesting to see if there are any other compound operator splitting techniques that will lead to a family of new algorithms along with possibly faster convergence rate.

Another interesting future direction is to explore proximal gradient TD algorithms with transfer RL. Given multiple different but related tasks, knowledge transfer is desirable and will help faster learning, less sample complexity, and better generalization ability. There are various types of transfer learning at different levels, such as instance-level transfer, feature-level transfer, and parameter-level transfer. As we know, from a transfer learning perspective, off-policy learning is instance level transfer learning. It would be interesting to see if other transfer RL problems can be formulated as saddle-point problems and if there is similar finite-sample analysis as well. Another interesting direction is to design new objective functions for TD learning. Since Bellman error is an expectation function (of the TD error), both MSPBE and NEU are (weighted) *norm of expectations* of the TD error. This is where the biased sampling problem comes from. It would be desirable if a new set of objectives can be designed, in which the biased sampling problem can be avoided. How to combine model-free temporal difference learning and learning with a generative model (Chen, Li, & Wang, 2018) is another interesting question to explore.

## Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant Nos. IIS-1216467 and ETRI funds at Auburn University. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

## References

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

Baird, L. C. (1995). Residual algorithms: Reinforcement learning with function approximation. In *International Conference on Machine Learning*, pp. 30–37.

Bauschke, H. H., & Combettes, P. L. (2011). *Convex analysis and monotone operator theory in Hilbert spaces*. Springer.

Bertsekas, D., & Tsitsiklis, J. (1996). *Neuro-Dynamic Programming*. Athena Scientific, Belmont, Massachusetts.

Bhatnagar, S., Sutton, R., Ghavamzadeh, M., & Lee, M. (2009). Natural actor-critic algorithms. *Automatica*, *45*(11), 2471–2482.

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

Boyd, S., & Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.

Bradtke, S., & Barto, A. (1996). Linear least-squares algorithms for temporal difference learning. *Machine Learning*, *22*, 33–57.

Bubeck, S. (2014). Theory of convex optimization for machine learning. In *arXiv:1405.4980*.

Chen, Y., Lan, G., & Ouyang, Y. (2013). Optimal primal-dual methods for a class of saddle point problems. In *arXiv:1309.5548*.

Chen, Y., Li, L., & Wang, M. (2018). Scalable bilinear $\pi$ learning using state and action features. In *arXiv preprint arXiv:1804.10328*.

Choi, D., & Van Roy, B. (2006). A generalized kalman filter for fixed point approximation and efficient temporal-difference learning. *Discrete Event Dynamic Systems*, *16*(2), 207–239.

Dalal, G., Szörényi, B., Thoppe, G., & Mannor, S. (2018a). Finite sample analyses for td (0) with function approximation. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*.

Dalal, G., Thoppe, G., Szorenyi, B., & Mannor, S. (2018b). Finite sample analysis of two-timescale stochastic approximation with applications to reinforcement learning. In *Proceedings of the 31st Conference On Learning Theory*, pp. 1199–1233.

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

Devolder, O. (2011). Stochastic first order methods in smooth convex optimization. Tech. rep., Université catholique de Louvain, Center for Operations Research and Econometrics.

Du, S. S., Chen, J., Li, L., Xiao, L., & Zhou, D. (2017). Stochastic variance reduction methods for policy evaluation. In *arXiv preprint arXiv:1702.07944*.

Duchi, J., Agarwal, A., Johansson, M., & Jordan, M. (2012). Ergodic mirror descent. *SIAM Journal on Optimization*, *22*(4), 1549–1578.

Geist, M., Scherrer, B., Lazaric, A., & Ghavamzadeh, M. (2012). A Dantzig Selector Approach to Temporal Difference Learning. In *International Conference on Machine Learning*, pp. 1399–1406.

Geist, M., & Scherrer, B. (2014). Off-policy learning with eligibility traces: a survey. *The Journal of Machine Learning Research*, *15*(1), 289–333.

Ghavamzadeh, M., Lazaric, A., Maillard, O., & Munos, R. (2010). LSTD with Random Projections. In *Proceedings of the International Conference on Neural Information Processing Systems*, pp. 721–729.

Ghavamzadeh, M., Lazaric, A., Munos, R., & Hoffman, M. (2011). Finite-Sample Analysis of Lasso-TD. In *Proceedings of the 28th International Conference on Machine Learning*, pp. 1177–1184.

Juditsky, A., & Nemirovski, A. (2011). *Optimization for Machine Learning*. MIT Press.

Juditsky, A., Nemirovskii, A., & Tauvel, C. (2008). Solving variational inequalities with stochastic mirror-prox algorithm. In *arXiv:0809.0815*.

Kakade, S., & Langford, J. (2002). Approximately optimal approximate reinforcement learning. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pp. 267–274.

Kamal, S. (2010). On the convergence, lock-in probability, and sample complexity of stochastic approximation. *SIAM Journal on Control and Optimization*, *48*(8), 5178–5192.

Kolter, Z. (2011). The Fixed Points of Off-Policy TD. In *Advances in Neural Information Processing Systems 24*, pp. 2169–2177.

Lagoudakis, M., & Parr, R. (2003). Least-squares policy iteration. *Journal of Machine Learning Research*, *4*, 1107–1149.

Lakshminarayanan, C., & Szepesvari, C. (2018). Linear stochastic approximation: How far does constant step-size and iterate averaging go?. In *International Conference on Artificial Intelligence and Statistics*, pp. 1347–1355.

Lazaric, A., Ghavamzadeh, M., & Munos, R. (2010a). Analysis of a classification-based policy iteration algorithm. In *Proceedings of the Twenty-Seventh International Conference on Machine Learning*, pp. 607–614.

Lazaric, A., Ghavamzadeh, M., & Munos, R. (2010b). Finite-Sample Analysis of LSTD. In *Proceedings of 27th International Conference on Machine Learning*, pp. 615–622.

Lazaric, A., Ghavamzadeh, M., & Munos, R. (2012). Finite-sample analysis of least-squares policy iteration. *Journal of Machine Learning Research*, *13*, 3041–3074.

Liu, B., Liu, J., Ghavamzadeh, M., Mahadevan, S., & Petrik, M. (2015). Finite-sample analysis of proximal gradient td algorithms.. In *UAI*, pp. 504–513.

Liu, B., Liu, J., Ghavamzadeh, M., Mahadevan, S., & Petrik, M. (2016). Proximal gradient temporal difference learning algorithms.. In *IJCAI*, pp. 4195–4199.

Liu, B., Mahadevan, S., & Liu, J. (2012). Regularized off-policy TD-learning. In *Advances in Neural Information Processing Systems 25*, pp. 845–853.

Maei, H. (2011). *Gradient temporal-difference learning algorithms*. Ph.D. thesis, University of Alberta.

Maei, H., & Sutton, R. (2010). GQ ($\lambda$): A general gradient algorithm for temporal-difference prediction learning with eligibility traces. In *Proceedings of the Third Conference on Artificial General Intelligence*, pp. 91–96.

Mahadevan, S., & Liu, B. (2012). Sparse Q-learning with Mirror Descent. In *Proceedings of the Conference on Uncertainty in AI*.

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

Munos, R., & Szepesvári, C. (2008). Finite time bounds for fitted value iteration. *Journal of Machine Learning Research*, *9*, 815–857.

Nemirovski, A., Juditsky, A., Lan, G., & Shapiro, A. (2009). Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, *19*, 1574–1609.

Palaniappan, B., & Bach, F. (2016). Stochastic variance reduction methods for saddle-point problems. In *Advances in Neural Information Processing Systems*, pp. 1416–1424.

Parr, R., Painter-Wakefield, C., Li, L., & Littman, M. (2007). Analyzing feature generation for value function approximation. In *Proceedings of the International Conference on Machine Learning*, pp. 737–744.

Pires, B. A., & Szepesvari, C. (2012). Statistical linear estimation with penalized estimators: an application to reinforcement learning. In *Proceedings of the 29th International Conference on Machine Learning*, pp. 1535–1542.

Polyak, B., & Juditsky, A. (1992). Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, *30*(4), 838–855.

Prashanth, L., Korda, N., & Munos, R. (2014). Fast LSTD using stochastic approximation: Finite time analysis and application to traffic control. In *Machine Learning and Knowledge Discovery in Databases*, pp. 66–81. Springer.

Qin, Z., & Li, W. (2014). Sparse Reinforcement Learning via Convex Optimization. In *Proceedings of the 31st International Conference on Machine Learning*.

Randløv, J., & Alstrøm, P. (1998). Learning to drive a bicycle using reinforcement learning and shaping. In *Proceedings of the International Conference on Machine Learning*, Vol. 98, pp. 463–471.

Sutton, R., & Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. MIT Press.

Sutton, R., Maei, H., Precup, D., Bhatnagar, S., Silver, D., Szepesvári, C., & Wiewiora, E. (2009). Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *International Conference on Machine Learning*, pp. 993–1000.

Sutton, R., Szepesvári, C., & Maei, H. (2008). A convergent o(n) algorithm for off-policy temporal-difference learning with linear function approximation. In *Neural Information Processing Systems*, pp. 1609–1616.

Szepesvári, C. (2010). Algorithms for reinforcement learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, *4*(1), 1–103.

Tagorti, M., & Scherrer, B. (2014). Rate of convergence and error bounds for LSTD ($\lambda$). In *arXiv:1405.3229*.

Thoppe, G., & Borkar, V. S. (2015). A concentration bound for stochastic approximation via alekseev's formula. In *arXiv preprint arXiv:1506.08657*.

Wang, Y., Chen, W., Liu, Y., Ma, Z., & Liu, T. (2017). Finite sample analysis of the gtd policy evaluation algorithms in markov setting. In *Advances in Neural Information Processing Systems*, pp. 5504–5513.

White, A., & White, M. (2016). Investigating practical linear temporal difference learning. In *Proceedings of the 2016 International Conference on Autonomous Agents and Multiagent Systems*, pp. 494–502.

Yu, H. (2017). On convergence of some gradient-based temporal-differences algorithms for off-policy learning. In *arXiv preprint arXiv:1712.09652*.

Yu, H. (2012). Least squares temporal difference methods: An analysis under general conditions. *SIAM Journal on Control and Optimization*, *50*(6), 3310–3343.

## Appendix A. Proof of Lemma 2

*Proof.* From the boundedness of the features (by $L$) and the rewards (by $\mathrm{R}_{\max}$), we have

$$\begin{aligned}
||A||_2 &= ||\mathbb{E}[\rho_t \phi_t \Delta \phi_t^\top]||_2 \\
&\leq \max_s ||\rho(s)\phi(s)(\Delta\phi(s))^\top||_2 \\
&\leq \rho_{\max} \max_s ||\phi(s)||_2 \max_s ||\phi(s) - \gamma\phi'(s)||_2 \\
&\leq \rho_{\max} \max_s ||\phi(s)||_2 \max_s \left(||\phi(s)||_2 + \gamma||\phi'(s)||_2\right) \\
&\leq (1+\gamma)\rho_{\max} L^2 d.
\end{aligned}$$

The second inequality is obtained by the consistent inequality of matrix norm, the third inequality comes from the triangular norm inequality, and the fourth inequality comes from the vector norm inequality $||\phi(s)||_2 \leq ||\phi(s)||_\infty \sqrt{d} \leq L\sqrt{d}$. The bound on $||b||_2$ can be derived in a similar way as follows.

$$\begin{aligned}
||b||_2 &= ||\mathbb{E}[\rho_t \phi_t r_t]||_2 \\
&\leq \max_s ||\rho(s)\phi(s)r(s)||_2 \\
&\leq \rho_{\max} \max_s ||\phi(s)||_2 \max_s ||r(s)||_2 \\
&\leq \rho_{\max} L \mathrm{R}_{\max}.
\end{aligned}$$

It completes the proof. ■

## Appendix B. Proof of Proposition 3

*Proof.* The proof of Proposition 3 mainly relies on Proposition 3.2 in (Nemirovski et al., 2009). We just need to map our convex-concave *stochastic* saddle-point problem in Eq. (13), i.e.,

$$\min_{\theta\in\Theta} \max_{y\in Y} \left( L(\theta, y) = \langle b - A\theta, y\rangle - \frac{1}{2}||y||_M^2 \right)$$

to the one in Section 3 of (Nemirovski et al., 2009) and show that it satisfies all the conditions necessary for their Proposition 3.2. Assumption 2 guarantees that our feasible sets $\Theta$ and $Y$ satisfy the conditions in (Nemirovski et al., 2009), as they are non-empty bounded closed convex subsets of $\mathbb{R}^d$. We also see that our objective function $L(\theta, y)$ is *convex* in $\theta \in \Theta$ and *concave* in $y \in Y$, and also *Lipschitz continuous* on $\Theta \times Y$. It is known that in the above setting, our saddle-point problem in Eq. (13) is solvable, i.e., the corresponding *primal* and *dual* optimization problems: $\min_{\theta\in\Theta}\left[\max_{y\in Y} L(\theta, y)\right]$ and $\max_{y\in Y}\left[\min_{\theta\in\Theta} L(\theta, y)\right]$ are solvable with equal optimal values, denoted $L^*$, and pairs $(\theta^*, y^*)$ of optimal solutions to the respective problems from the set of saddle-points of $L(\theta, y)$ on $\Theta \times Y$.

For our problem, the *stochastic sub-gradient vector* $G$ is defined as

$$G(\theta, y) = \begin{bmatrix} G_\theta(\theta, y) \\ -G_y(\theta, y) \end{bmatrix} = \begin{bmatrix} -\hat{A}_t^\top y \\ -(\hat{b}_t - \hat{A}_t\theta - \hat{M}_t y) \end{bmatrix}.$$

This guarantees that the *deterministic sub-gradient vector*

$$g(\theta, y) = \begin{bmatrix} g_\theta(\theta, y) \\ -g_y(\theta, y) \end{bmatrix} = \begin{bmatrix} \mathbb{E}\big[G_\theta(\theta, y)\big] \\ -\mathbb{E}\big[G_y(\theta, y)\big] \end{bmatrix}$$

is well-defined, i.e., $g_\theta(\theta, y) \in \partial_\theta L(\theta, y)$ and $g_y(\theta, y) \in \partial_y L(\theta, y)$.

We also consider the Euclidean stochastic approximation (E-SA) setting in (Nemirovski et al., 2009) in which the *distance generating functions* $\omega_\theta : \Theta \to \mathbb{R}$ and $\omega_y : Y \to \mathbb{R}$ are simply defined as

$$\omega_\theta = \frac{1}{2}||\theta||_2^2, \qquad \omega_y = \frac{1}{2}||y||_2^2,$$

modulus 1 w.r.t. $||\cdot||_2$, and thus, $\Theta^o = \Theta$ and $Y^o = Y$ (see pp. 1581 and 1582 in (Nemirovski et al., 2009)). This allows us to equip the set $Z = \Theta \times Y$ with the distance generating function

$$\omega(z) = \frac{\omega_\theta(\theta)}{2D_\theta^2} + \frac{\omega_y(y)}{2D_y^2},$$

where $D_\theta$ and $D_y$ are defined in Assumption 2.

Now that we consider the Euclidean case and set the norms to $\ell_2$-norm, we can compute upper-bounds on the expectation of the dual norm of the stochastic sub-gradients

$$\mathbb{E}\left[||G_\theta(\theta, y)||_{*,\theta}^2\right] \le M_{*,\theta}^2, \quad \mathbb{E}\left[||G_y(\theta, y)||_{*,y}^2\right] \le M_{*,y}^2,$$

where $||\cdot||_{*,\theta}$ and $||\cdot||_{*,y}$ are the dual norms in $\Theta$ and $Y$, respectively. Since we are in the Euclidean setting and use the $\ell_2$-norm, the dual norms are also $\ell_2$-norm, and thus, to compute $M_{*,\theta}$, we need to upper-bound $\mathbb{E}\left[||G_\theta(\theta, y)||_2^2\right]$ and $\mathbb{E}\left[||G_y(\theta, y)||_2^2\right]$.

To bound these two quantities, we use the following equality that holds for any random variable $x$:

$$\mathbb{E}[||x||_2^2] = \mathbb{E}[||x - \mu_x||_2^2] + ||\mu_x||_2^2,$$

where $\mu_x = \mathbb{E}[x]$. Here is how we bound $\mathbb{E}\left[||G_\theta(\theta, y)||_2^2\right]$,

$$\begin{aligned}
\mathbb{E}\left[||G_\theta(\theta, y)||_2^2\right] &= \mathbb{E}[||\hat{A}_t^\top y||_2^2] \\
&= \mathbb{E}[||\hat{A}_t^\top y - A^\top y||_2^2] + ||A^\top y||_2^2 \\
&\le \sigma_2^2 + (||A||_2 ||y||_2)^2 \\
&\le \sigma_2^2 + ||A||_2^2 R^2,
\end{aligned}$$

where the first inequality is from the definition of $\sigma_2$ in Eq. (16) and the consistent inequality of the matrix norm, and the second inequality comes from the boundedness of the feasible sets in Assumption 2. Similarly we bound $\mathbb{E}\left[||G_y(\theta, y)||_2^2\right]$ as follows:

$$\begin{aligned}
\mathbb{E}[||G_y(\theta, y)||_2^2] &= \mathbb{E}[||\hat{b}_t - \hat{A}_t\theta - \hat{M}_t y||_2^2] \\
&= ||b - A\theta + My||_2^2 \\
&+ \mathbb{E}[||\hat{b}_t - \hat{A}_t\theta - \hat{M}_t y - (b - A\theta - My)||_2^2] \\
&\le (||b||_2 + ||A||_2||\theta||_2 + \tau||y||_2)^2 + \sigma_1^2 \\
&\le \big(||b||_2 + (||A||_2 + \tau)R\big)^2 + \sigma_1^2,
\end{aligned}$$

where these inequalities come from the definition of $\sigma_1$ in Eq. (16) and the boundedness of the feasible sets in Assumption 2. This means that in our case we can compute $M_{*,\theta}^2, M_{*,y}^2$ as

$$M_{*,\theta}^2 = \sigma_2^2 + ||A||_2^2 R^2,$$
$$M_{*,y}^2 = \big(||b||_2 + (||A||_2 + \tau)R\big)^2 + \sigma_1^2,$$

and as a result

$$\begin{aligned} M_*^2 &= 2D_\theta^2 M_{*,\theta}^2 + 2D_y^2 M_{*,y}^2 = 2R^2(M_{*,\theta}^2 + M_{*,y}^2) \\ &= R^2\left(\sigma^2 + ||A||_2^2 R^2 + \big(||b||_2 + (||A||_2 + \tau)R\big)^2\right) \\ &\leq \big(R^2\,(2||A||_2 + \tau) + R(\sigma + ||b||_2)\big)^2, \end{aligned}$$

where the inequality comes from the fact that $\forall a, b, c \geq 0, a^2 + b^2 + c^2 \leq (a + b + c)^2$. Thus, we may write $M_*$ as

$$M_* = R^2\,(2||A||_2 + \tau) + R(\sigma + ||b||_2). \tag{27}$$

Now we have all the pieces ready to apply Proposition 3.2 in (Nemirovski et al., 2009) and obtain a high-probability bound on $\mathrm{Err}(\bar{\theta}_n, \bar{y}_n)$, where $\bar{\theta}_n$ and $\bar{y}_n$ (see Eq. (15)) are the outputs of the revised GTD algorithm in Algorithm 1. From Proposition 3.2 in (Nemirovski et al., 2009), if we set the step-size in Algorithm 1 (our revised GTD algorithm) to $\alpha_t = \frac{2c}{M_*\sqrt{5n}}$, where $c > 0$ is a positive constant, $M_*$ is defined by Eq. (27), and $n$ is the number of training samples in $\mathcal{D}$, with probability of at least $1 - \delta$, we have

$$\mathrm{Err}(\bar{\theta}_n, \bar{y}_n) \leq \sqrt{\frac{5}{n}}(8 + 2\log\frac{2}{\delta})R^2\left(2||A||_2 + \tau + \frac{||b||_2 + \sigma}{R}\right). \tag{28}$$

Note that we obtain Eq. (28) by setting $c = 1$ and the "light-tail" assumption in Eq. (18) guarantees that we satisfy the condition in Eq. 3.16 in (Nemirovski et al., 2009), which is necessary for the high-probability bound in their Proposition 3.2 to hold. The proof is complete by replacing $||A||_2$ and $||b||_2$ from Lemma 2. ■

## Appendix C. Proof of Proposition 4

*Proof.* From Lemma 3, we have

$$\begin{aligned} V - \bar{v}_n = (I - \gamma\Pi P)^{-1}\times \\ \big[(V - \Pi V) + \Phi C^{-1}(b - A\bar{\theta}_n)\big]. \end{aligned}$$

Applying $\ell_2$-norm w.r.t. the distribution $\xi$ to both sides of this equation, we obtain

$$\begin{aligned} ||V - \bar{v}_n||_\xi \leq &||(I - \gamma\Pi P)^{-1}||_\xi \times \\ &\big(||V - \Pi V||_\xi + ||\Phi C^{-1}(b - A\bar{\theta}_n)||_\xi\big). \end{aligned} \tag{29}$$

Since $P$ is the kernel matrix of the target policy $\pi$ and $\Pi$ is the orthogonal projection w.r.t. $\xi$, the stationary distribution of $\pi$, we may write

$$||(I - \gamma\Pi P)^{-1}||_\xi \leq \frac{1}{1 - \gamma}.$$

Moreover, we may upper-bound the term $||\Phi C^{-1}(b - A\bar{\theta}_n)||_\xi$ in (29) using the following inequalities:

$$\begin{aligned}
&||\Phi C^{-1}(b - A\bar{\theta}_n)||_\xi \\
&\quad \leq ||\Phi C^{-1}(b - A\bar{\theta}_n)||_2 \sqrt{\xi_{\max}} \\
&\quad \leq ||\Phi||_2 ||C^{-1}||_2 ||(b - A\bar{\theta}_n)||_{M^{-1}} \sqrt{\tau \xi_{\max}} \\
&\quad \leq (L\sqrt{d})(\frac{1}{\nu})\sqrt{2\mathrm{Err}(\bar{\theta}_n, \bar{y}_n)}\sqrt{\tau \xi_{\max}} \\
&\quad = \frac{L}{\nu}\sqrt{2d\tau\xi_{\max}\mathrm{Err}(\bar{\theta}_n, \bar{y}_n)},
\end{aligned}$$

where the third inequality is the result of upper-bounding $||(b - A\bar{\theta}_n)||_M^{-1}$ using Eq. (21) and the fact that $\nu = 1/||C^{-1}||_2^2 = 1/\lambda_{\max}(C^{-1}) = \lambda_{\min}(C)$ ($\nu$ is the smallest eigenvalue of the covariance matrix $C$). ■

## Appendix D. Proof of Proposition 5

*Proof.* Using the triangle inequality, we may write

$$||V - \bar{v}_n|||_\xi \leq ||\bar{v}_n - \Phi\theta^*||_\xi + ||V - \Phi\theta^*||_\xi. \tag{30}$$

The second term on the right-hand side of Eq. (30) can be upper-bounded by Lemma 4. Now we upper-bound the first term as follows:

$$\begin{aligned}
&||\bar{v}_n - \Phi\theta^*||_\xi^2 \\
&\quad = ||\Phi\bar{\theta}_n - \Phi\theta^*||_\xi^2 \\
&\quad = ||\bar{\theta}_n - \theta^*||_C^2 \\
&\quad \leq ||\bar{\theta}_n - \theta^*||_{A^\top M^{-1} A}^2 ||(A^\top M^{-1} A)^{-1}||_2 ||C||_2 \\
&\quad = ||A(\bar{\theta}_n - \theta^*)||_{M^{-1}}^2 ||(A^\top M^{-1} A)^{-1}||_2 ||C||_2 \\
&\quad = ||A\bar{\theta}_n - b||_{M^{-1}}^2 \frac{\tau_C}{\sigma_{\min}(A^\top M^{-1} A)},
\end{aligned}$$

where $\tau_C = \sigma_{\max}(C)$ is the largest singular value of $C$, and $\sigma_{\min}(A^\top M^{-1} A)$ is the smallest singular value of $A^\top M^{-1} A$. Using the result of Theorem 1, with probability at least $1 - \delta$, we have

$$\frac{1}{2}||A\bar{\theta}_n - b||_{M^{-1}}^2 \leq \tau\xi_{\max}\mathrm{Err}(\bar{\theta}_n, \bar{y}_n).$$

Thus,

$$||\bar{v}_n - \Phi\theta^*||_\xi^2 \leq \frac{2\tau_C\tau\xi_{\max}}{\sigma_{\min}(A^\top M^{-1} A)}\mathrm{Err}(\bar{\theta}_n, \bar{y}_n) \tag{31}$$

From Eqs. (30), (23), and (31), the result of Eq. (24) can be derived, which completes the proof. ■

## Appendix E. Battery Domain

The problem represents an energy arbitrage model with multiple finite *known* price levels and a stochastic evolution given a limited storage capacity. In particular, the storage is assumed to be an electrical battery that degrades when energy is stored or retrieved. Energy prices are governed by a Markov process with states $\Theta$. There are two energy prices in each time step: $p^i : \Theta \to \mathbb{R}$ is the purchase (or input) price and $p^o : \Theta \to \mathbb{R}$ is the sell (or output) price. The parameter $\theta$ vary between 0 and 10 and their evolution is governed by a martingale with a normal distribution around the mean.

We use $s$ to denote the available battery capacity with $s_0$ denoting the initial capacity. The current state of charge is denoted by $x$ or $y$ and must satisfy that $0 \leq x_t \leq s_t$ at any time step $t$. The action is the amount of energy to charge or discharge, which is denoted by $u$. Positive $u$ indicates that energy is purchased to charge the battery; negative $u$ indicates the sale of energy.

The battery storage degrades with use. The degradation is a function of the battery capacity when charged or discharged. We use a general model of battery degradation with a specific focus on Li-ion batteries. The degradation function $d(x, u) \in \mathbb{R}$ represent the battery capacity loss after starting at the state of charge $x \geq 0$ and charging (discharging if negative) by $u$ with $-x \leq u \leq s_0$. This function indicates the loss of capacity, such that:

$$s_{t+1} = s_t - d(x_t, u_t)$$

The state set in the Markov decision problem is composed of $(x, s, \theta)$ where $x$ is the state of charge, $s$ is the battery capacity, and $\theta \in \Theta$ is the state of the price process. The available actions in a state $(x, s, \theta)$ are $u$ such that $-x \leq u \leq s-x$. The transition is from $(x_t, s_t, \theta_t)$ to $(x_{t+1}, s_{t+1}, \theta_{t+1})$ given action $u_t$ is:

$$\begin{aligned} x_{t+1} &= x_t + u_t \\ s_{t+1} &= s_t - d(x_t, u_t) \end{aligned}$$

The probability of this transition is given by $P[\theta_{t+1}|\theta_t]$. The reward for this transition is:

$$r((x_t, s_t, \theta_t), u_t) = \begin{cases} -u_t \cdot p^i - c^d \cdot d(x_t, u_t) & \text{if } u_t \geq 0 \\ -u_t \cdot p^o - c^d \cdot d(x_t, u_t) & \text{if } u_t < 0 \end{cases}.$$

That is, the reward captures the monetary value of the transaction minus a penalty for degradation of the battery. Here, $c^d$ represents the cost of a unit of lost battery capacity.

The Bellman optimality equations for this problem are:

$$\begin{aligned} q_T(x, s, \theta) &= 0 \\ v_t(x, s, \theta_t) &= \min\big\{p^i_{\theta_t}[u]_+ + p^o_{\theta_t}[u]_- + \\ &\quad + c^d\, d(x, u) + \\ &\quad + q_t(x + u, s - d(x, u), \theta_t) : \\ &\quad\quad : u \in [-x, s - x]\big\} \\ q_t(x, s, \theta_t) &= \gamma \cdot \mathrm{E}[v_{t+1}(x, s, \theta_{t+1})] \end{aligned}$$

where the expectation $\mathrm{E}[v_{t+1}(x, s, \theta_{t+1})]$ is taken over $P(\theta_{t+1}|\theta_t)$.

The value function is approximated using piece-wise linear features of three types $\phi^1$, $\phi^2$, $\phi^3$ defined as a function of the MDP state as follows:

$$\phi^1_{w,q}(x,s,\theta) = \begin{cases} [x-w]_+ & \text{if } \theta = q \\ 0 & \text{otherwise} \end{cases}$$

$$\phi^2_{w,q}(x,s,\theta) = \begin{cases} [s-w]_+ & \text{if } \theta = q \\ 0 & \text{otherwise} \end{cases}$$

$$\phi^3_{w,q}(x,s,\theta) = \begin{cases} [s+x-w]_+ & \text{if } \theta = q \\ 0 & \text{otherwise} \end{cases}$$

Here, $w \in \{0, 0.1, \ldots, 0.9, 1\}$ and $q \in \Theta$.

These features can be conveniently used to approximate a piece-wise linear function.