An MRP Formulation for Supervised Learning: Generalized Temporal Difference Learning Models

YANGCHEN PAN*, University of Oxford, United Kingdom
JUNFENG WEN†, Carleton University, Canada
CHENJUN XIAO, The Chinese University of Hong Kong (Shenzhen), China
PHILIP H.S. TORR, University of Oxford, United Kingdom

Background: Traditional supervised learning (SL) assumes data points are independently and identically distributed (i.i.d.), which overlooks dependencies in real-world data. Reinforcement learning (RL), in contrast, models dependencies through state transitions.
Objectives: This study aims to bridge SL and RL by reformulating SL problems as RL tasks, enabling the application of RL techniques to a wider range of SL scenarios. We aim to model SL data as interconnected, and develop novel temporal difference (TD) algorithms that can accommodate diverse data types. Our objectives are to (1) establish conditions where TD outperforms ordinary least squares (OLS), (2) provide convergence guarantees for the generalized TD algorithm, and (3) validate the approach empirically using synthetic and real-world datasets.
Methods: We reformulate traditional SL as a RL problem by modeling data points as a Markov Reward Process (MRP). We then introduce a concept analogous to the inverse link function in generalized linear models, allowing our TD algorithm to handle various data types. Our analysis, grounded in variance estimation, identifies conditions where TD outperforms OLS. We establish a convergence guarantee by conceptualizing the TD update rule as a generalized Bellman operator. Empirical validation begins with synthetic data progressively matching theoretical assumptions to verify our analysis, followed by evaluations on real-world datasets to demonstrate practical utility.
Results: Our theoretical analysis shows that TD can outperform OLS in estimation accuracy when data noise is correlated. Our approach generalizes across various loss functions and SL datasets. We prove that the Bellman operator in our TD framework is a contraction, ensuring convergence for both expected and stochastic TD updates. Empirically, TD outperforms SL baselines when data aligns with its assumptions, remains competitive across diverse datasets, and is robust to hyperparameter choices.
Conclusions: This study demonstrates that SL can be reformulated as a problem of interconnected data modeled by an MRP, effectively solved using TD learning. Our generalized TD is theoretically sound, with convergence guarantees, and practically effective. It generalizes OLS, offering superior performance on correlated data. This work enables RL techniques to benefit SL tasks, offering a pathway for future advancements.

JAIR Associate Editor: Marek Petrik

JAIR Reference Format:
Yangchen Pan, Junfeng Wen, Chenjun Xiao, and Philip H.S. Torr. 2025. An MRP Formulation for Supervised Learning: Generalized Temporal Difference Learning Models. *Journal of Artificial Intelligence Research* 83, Article 33 (August 2025), 30 pages. DOI: 10.1613/jair.1.19171

*Corresponding Author.
†Co-first & Corresponding author with Yangchen Pan.

Authors' Contact Information: Yangchen Pan, ORCID: 0009-0000-8297-9045, yangchen.pan@eng.ox.ac.uk, University of Oxford, Oxford, United Kingdom; Junfeng Wen, ORCID: 0000-0002-4128-1422, junfengwen@gmail.com, Carleton University, Ottawa, Canada; Chenjun Xiao, ORCID: 0009-0006-5051-7384, chenjunx@cuhk.edu.cn, The Chinese University of Hong Kong (Shenzhen), Shenzhen, China; Philip H.S. Torr, ORCID: 0009-0006-0259-5732, philip.torr@eng.ox.ac.uk, University of Oxford, Oxford, United Kingdom.

DOI: 10.1613/jair.1.19171

1 Introduction

The primary objective of statistical supervised learning (SL) is to learn the relationship between the features and the output (response) variable. To achieve this, generalized linear models [34, 31] are considered a generic algorithmic framework employed to derive objective functions. These models make specific assumptions regarding the conditional distribution of the response variable given input features, which can take forms such as Gaussian (resulting in ordinary least squares), Poisson (resulting in Poisson regression), or multinomial (resulting in logistic regression or multiclass softmax cross-entropy loss).

In recent years, reinforcement learning (RL)—widely used in interactive learning scenarios—has experienced a surge in popularity. This growth has fostered increasing synergy between RL and SL, where each paradigm complements the other in meaningful ways. In SL-assisted RL, for example, the subfield of imitation learning[18] leverages expert demonstrations to regularize or accelerate the RL training process. Weakly supervised methods[25] have also been employed to constrain the RL task space, while relabeling techniques have facilitated more effective goal-conditioned policy learning [14]. Return-conditioned policy learning naturally incorporates traditional SL techniques. A common approach is to maximize the log-likelihood of a policy conditioned not only on the state but also on the trajectory return [10]. This method is typically applied in offline RL settings, where the presence of high-return trajectories makes learning a return-conditioned policy particularly appealing. Earlier work has also drawn from SL techniques to address RL challenges. For instance, Dietterich and Wang [12] proposed using support vector machines for batch value function approximation. Similarly, kernel-based methods originally developed for SL were adapted to RL problems by Ormoneit and Sen [35] and Barreto et al. [4].

Conversely, RL has also expanded its application into traditional SL domains. RL has proven effective in fine-tuning large language models [28], aligning them with user preferences. Additionally, RL algorithms [16] have been tailored for training neural networks (NNs), treating individual network nodes as RL agents. In the realm of imbalanced classification, RL-based control algorithms have been developed, where predictions correspond to actions, rewards are based on heuristic correctness criteria, and episodes conclude upon incorrect predictions within minority classes [27]. Permative prediction [42] provides a theoretical framework that can deal with nonstationary data and it can be reframed within a RL context.

Existing approaches that propose a RL framework for solving SL problems often exhibit a heuristic nature. These approaches involve crafting specific formulations, including elements like agents, reward functions, action spaces, and termination conditions, based on intuitive reasoning tailored to particular tasks. Consequently, they lack generality, and their heuristic nature leaves theoretical assumptions, connections between optimal RL and SL solutions, and convergence properties unclear.

To the best of our knowledge, among existing work, the most generic and theoretically sound approach to converting a SL problem into a RL problem involves using the loss function of the SL task to define a reward function—typically mapping smaller losses to higher rewards [5]. However, this approach (1) does not fundamentally alter the underlying data generation assumption to that of a Markov process, and (2) consequently does not enable the application of one of RL's most distinctive algorithms—temporal-difference (TD) learning—to supervised learning problems. Therefore, it remains unclear whether there exists a unified and systematic RL formulation that both redefines the data generation process and enables the use of TD learning to model a broad class of conventional SL tasks. Such a formulation should be agnostic to learning settings, including various tasks such as ordinary least squares regression, Poisson regression, binary or multi-class classification, etc.

In this study, we introduce a generic Markov process formulation for data generation, offering an alternative to the conventional i.i.d. data assumption in SL. Specifically, when faced with a SL dataset, we view the data points as originating from a Markov reward process (MRP) [48]. To accommodate a wide range of problems, such as Poisson regression, binary or multi-class classification, we introduce a generalized TD learning model in Section 3. Section 4 explores the relationship between the solutions obtained through TD learning and the original

linear regression. Furthermore, we prove that under specific conditions with correlated noise, TD estimator is more efficient than the traditional ordinary least squares (OLS) estimator. We show that our generalized TD algorithm—which can handle various types of datasets with different loss functions—corresponds to the application of a specialized and generalized Bellman operator. We further prove that this operator is contractive and admits a unique fixed point. Based on this property, we establish convergence results under linear function approximation, as detailed in Section 5. Our paper concludes with an empirical evaluation of our TD algorithm in Section 6, verifying our theoretical results, assessing its critical design choices and practical utility when integrated with a deep neural network across various tasks, achieving competitive results and, in some cases, improvements in generalization performance. We view our work as a step towards unifying diverse learning tasks from two pivotal domains within a single, coherent theoretical framework.

2 Background

This section provides a brief overview of the relevant concepts from statistical SL and RL settings, laying the groundwork for the rest of this paper.

2.1 Conventional Supervised Learning

In the context of statistical learning, we make the assumption that data points, in the form of $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$, are independently and identically distributed (i.i.d.) according to some unknown probability distribution P. The goal is to find the relationship between the feature $\mathbf{x}$ and the response variable y given a training dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$.

In a simple linear function approximation case, a commonly seen algorithm is ordinary least squares (OLS) that optimizes the squared error objective function

$$\min_{\mathbf{w}} \ ||\mathbf{X}\mathbf{w} - \mathbf{y}||_2^2. \tag{1}$$

where $\mathbf{X}$ is the $n \times d$ feature matrix, $\mathbf{y}$ is the corresponding n-dimensional training label vector, and the $\mathbf{w}$ is the parameter vector we aim to optimize.

From a probabilistic perspective, this objective function can be derived by assuming $p(y|\mathbf{x})$ follows a Gaussian distribution with mean $\mathbf{x}^\top \mathbf{w}$ and conducting maximum likelihood estimation (MLE) for $\mathbf{w}$ with the training dataset. It is well known that $\mathbb{E}[Y|\mathbf{x}]$ is the optimal predictor [7]. For many other choices of distribution $p(y|\mathbf{x})$, generalized linear models (GLMs) [34] are commonly employed to estimate $\mathbb{E}[Y|\mathbf{x}]$. This includes OLS, Poisson regression [33] and logistic regression, etc.

An important concept in GLMs is the *inverse link function*, which we denote as f, that establishes a connection between the linear prediction (also called the **logit**), and the conditional expectation: $\mathbb{E}[Y|\mathbf{x}] = f(\mathbf{x}^\top \mathbf{w})$. For example, in logistic regression, the inverse link function is the sigmoid function. We later propose generalized TD learning models within the framework of RL that correspond to GLMs, enabling us to handle a wide range of data that are modeled by different distributions in supervised learning.

2.2 Reinforcement Learning

Reinforcement learning is often formulated within the Markov decision process (MDP) framework. An MDP can be represented as a tuple $(\mathcal{S}, \mathcal{A}, r, P, \gamma)$ [43], where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $r : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is the reward function, $P(\cdot|s, a)$ defines the transition probability, and $\gamma \in (0, 1)$ is the discount factor. Given a policy $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$, the return at time step t is $G_t = \sum_{i=0}^{\infty} \gamma^i r(S_{t+i}, A_{t+i})$, and the value of a state $s \in \mathcal{S}$ is the expected return starting from that state $v^\pi(s) = \mathbb{E}_\pi[G_t|S_t = s]$. In this work, we focus on the policy evaluation problem for a fixed policy, thus the MDP can be reduced to a Markov reward process (MRP) [48] described

by $(\mathcal{S}, r^\pi, P^\pi, \gamma)$ where $r^\pi(s) \stackrel{\text{def}}{=} \sum_a \pi(a|s) r(s,a)$ and $P^\pi(s'|s) \stackrel{\text{def}}{=} \sum_a \pi(a|s) P(s'|s,a)$. When it is clear from the context, we will slightly abuse notations and ignore the superscript π.

In policy evaluation problem, the objective is to estimate the state value function of a fixed policy π by using the trajectory $s_0, r_1, s_1, r_2, s_2, \ldots$ generated from π. In linear function approximation, the value function is approximated by a parametrized function $v(s) \approx \mathbf{x}(s)^\top \mathbf{w}$ with parameters $\mathbf{w}$ and some fixed feature mapping $\mathbf{x} : \mathcal{S} \mapsto \mathbb{R}^d$ where d is the feature dimension. Note that the state value satisfies the Bellman equation

$$v(s) = r(s) + \gamma \mathbb{E}_{S' \sim P(\cdot|s)}[v(S')]. \tag{2}$$

One fundamental approach to the evaluation problem is the temporal difference (TD) learning [46], which uses a sampled transition s_t, r_{t+1}, s_{t+1} to update the parameters $\mathbf{w}$ through stochastic fixed-point iteration based on (2) with a step-size $\alpha > 0$:

$$\mathbf{w} \leftarrow \mathbf{w} + \alpha(y_{t,td} - \mathbf{x}(s_t)^\top \mathbf{w})\mathbf{x}(s_t), \tag{3}$$

where $y_{t,td} \stackrel{\text{def}}{=} r_{t+1} + \gamma \mathbf{x}(s_{t+1})^\top \mathbf{w}$. To simplify notations and align concepts, we will use $\mathbf{x}_t \stackrel{\text{def}}{=} \mathbf{x}(s_t)$. In linear function approximation setting, TD converges to the solution that solves the system $\mathbf{A}\mathbf{w} = \mathbf{b}$ [9, 49], where

$$\mathbf{A} = \mathbb{E}[\mathbf{x}_t(\mathbf{x}_t - \gamma \mathbf{x}_{t+1})^\top] = \mathbf{X}^\top \mathbf{D}(\mathbf{I} - \gamma \mathbf{P})\mathbf{X} \qquad \text{and} \qquad \mathbf{b} = \mathbb{E}[r_{t+1}\mathbf{x}_t] = \mathbf{X}^\top \mathbf{D}\mathbf{r} \tag{4}$$

with $\mathbf{X} \in \mathbb{R}^{|\mathcal{S}| \times d}$ being the feature matrix whose rows are the state features $\mathbf{x}_t$, $\mathbf{D} \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{S}|}$ being the diagonal matrix with the stationary distribution $D(s_t)$ on the diagonal, $\mathbf{P} \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{S}|}$ being the transition probability matrix (i.e., $\mathbf{P}_{ij} = P(s_j|s_i)$) and $\mathbf{r} \in \mathbb{R}^{|\mathcal{S}|}$ being the reward vector. Note that the matrix $\mathbf{A}$ is often invertible under mild conditions [49].

3 MRP View and Generalized TD Learning

This section describes our MRP construction given the same dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ and proposes our generalized TD learning algorithm to solve it. This approach is based on the belief that these data points originate from some MRP, rather than being i.i.d. generated.

Regression. We start by considering the basic regression setting with linear models before introducing our generalized TD algorithm. Table 1 summarizes how we can view concepts in conventional SL from an RL perspective. The key is to treat the original training label as a state value that we are trying to learn, and then the reward function can be derived from the Bellman equation (2) as

$$r(s) = v(s) - \gamma \mathbb{E}_{S' \sim P(\cdot|s)}[v(S')]. \tag{5}$$

We will discuss the choice of $\mathbf{P}$ later. Note that although Equation (5) may appear familiar to those in the inverse reinforcement learning (IRL) community, where the motivation often stems from the difficulty of specifying a reward function and thus inferring it from expert demonstrations, our approach is fundamentally different from IRL. In our case, the motivation is not related to the challenge of reward specification, as this is a supervised learning problem in which even the value function is explicitly known. As a result, the reward can be directly estimated from a single transition. At each iteration (or time step in RL), the reward can be approximated using a stochastic example. For instance, assume that at iteration t (i.e., time step in RL), we obtain an example $(\mathbf{x}_i^{(t)}, y_i^{(t)})$. We use superscripts and subscripts to denote that the ith training example is sampled at the tth time step. The next example $(\mathbf{x}_j^{(t+1)}, y_j^{(t+1)})$ is then sampled according to $P(\cdot|\mathbf{x}_i^{(t)})$ and the reward can be estimated as $r^{(t+1)} = y_i^{(t)} - \gamma y_j^{(t+1)}$ by approximating the expectation in Equation (5) with a stochastic example. As one might notice, in a sequential setting, t is monotonically increasing, so from now on, we will simply use the simplified notation $(\mathbf{x}_t, y_t)$ to denote the training example sampled at time step t.

We now summarize and compare the updating rules in conventional SL and in our TD algorithm under linear function approximation. At time step t, the **conventional updating rule** based on stochastic gradient descent

Table 1. How definitions in SL correspond to those in RL

SL Definitions	RL Definitions
Feature matrix $\mathbf{X}$	Feature matrix $\mathbf{X}$
Feature of the ith example $\mathbf{x}_i$	The state feature $\mathbf{x}(s_i) = \mathbf{x}_i$
Training target y_i of $\mathbf{x}_i$	The state value $v(s_i) = y_i$

(SGD) is

$$\mathbf{w} \leftarrow \mathbf{w} + \alpha(y_t - \mathbf{x}_t^\top \mathbf{w})\mathbf{x}_t, \tag{6}$$

while our **TD updating rule** is

$$\mathbf{w} \leftarrow \mathbf{w} + \alpha(y_{t,td} - \mathbf{x}_t^\top \mathbf{w})\mathbf{x}_t \tag{7}$$

$$\text{where } \; y_{t,td} \stackrel{\text{def}}{=} r_{t+1} + \gamma \widehat{y}_{t+1} = y_t - \gamma y_{t+1} + \gamma \mathbf{x}_{t+1}^\top \mathbf{w} \tag{8}$$

and $\mathbf{x}_{t+1} \sim P(\cdot|\mathbf{x}_t)$ with the ground truth label y_{t+1}. The critical difference is that TD uses a bootstrap, so it does not cancel the γy_{t+1} term from the reward when computing the TD training target $y_{t,td}$. By setting $\gamma = 0$, the original supervised learning updating rule (6) is recovered.

Generalized TD: An extension to general learning tasks. A natural question regarding TD is how to extend it to different types of data, such as those with counting, binary, or multiclass labels. Recall that in generalized linear models (GLMs), it is assumed that the output variable $y \in \mathcal{Y}$ follows an exponential family distribution. In addition, there exists an *inverse link function* f that maps a linear prediction $z \stackrel{\text{def}}{=} \mathbf{w}^\top \mathbf{x}$ to the output/label space $\mathcal{Y}$ (i.e., $f(z) \in \mathcal{Y}, \forall z \in \mathbb{R}$). Examples of GLMs include linear regression (where y follows a Gaussian distribution, f is the identity function and the loss is the squared loss) and logistic regression (where y is Bernoulli, f is the sigmoid function and the loss is the log loss). More generally, the output may be in higher-dimensional space and both z and y will be vectors instead of scalars. As an example, multinomial regression uses the softmax function f to convert a vector $\mathbf{z}$ to another vector $\mathbf{y}$ in the probability simplex. Interested readers can refer to Banerjee et al. [3] and Helmbold et al. [17]; McCullagh and Nelder [31, Table 2.1] for more details. As per convention, we refer to z as **logit**.

Returning to the TD algorithm, the significance of logit z is that it is naturally *additive on the real line* (e.g., in logistic regression, a larger logit indicates a higher chance of being in the positive class), which mirrors the additive nature of returns (cumulative sum of rewards) in RL. It also implies that two linear predictions can be added and the resultant $z = z_1 + z_2$ can still be transformed to a valid output $f(z) \in \mathcal{Y}$. In contrast, adding two labels does not necessarily produce a valid label in general $y_1 + y_2 \notin \mathcal{Y}$. Therefore, the idea is to construct a bootstrapped target in the real line (logit space, or z-space)

$$z_{t,td} \stackrel{\text{def}}{=} r_{t+1} + \gamma \widehat{z}_{t+1} = (z_t - \gamma z_{t+1}) + \gamma \mathbf{x}_{t+1}^\top \mathbf{w} \tag{9}$$

and then convert it back to the original label space to obtain the TD target $y_{t,td} = f(z_{t,td})$. In multiclass classification problems, we often use a one-hot vector to represent the original training target. For instance, in the case of MNIST [24], the target is a ten-dimensional one-hot vector. Consequently, the reward becomes a vector with each component corresponding to a class. This can be interpreted as evaluating the policy under ten different reward functions in parallel and selecting the highest value for prediction.

Algorithm 1 provides the pseudo-code of our algorithm when using linear models. At time step t, the process begins by sampling the state $\mathbf{x}_t$, and then we sample the next state according to the predefined P. The reward is computed as the difference in logits after converting the original labels y_t, y_{t+1} into the logit space with the link function. Subsequently, the TD bootstrap target is constructed in the logit space. Finally, the TD target

is transformed back to the original label space before it is used to calculate the loss. Note that the standard regression is a special case where the (inverse) link function is simply the identity function, so it reduces to the standard update (6) with squared loss. In practice, we might need some smoothing parameter when the function f^{-1} goes to infinity. For example, in binary classification, $\mathcal{Y} = \{0, 1\}$ and the corresponding logits are $z = -\infty$ and $z = \infty$. To avoid this, we subtract/add a small value to the label before applying f^{-1} so that z is finite. In case of non-linear models such as deep neural networks (DNNs), the update to the parameters $\mathbf{w}$ can be carried out by gradient descent on the corresponding loss function, replacing y_t with $f(z_{t,td})$.

Algorithm 1 Generalized TD for SL

Input: A dataset $\mathcal{D}$
Initialize $\mathbf{w}_0$
Randomly sample a starting data point $(\mathbf{x}_0, y_0) \in \mathcal{D}$. (One can also use mini-batch starting points in NNs.)
for $t = 0, 1, 2, \ldots$ **do**
  Sample $\mathbf{x}_{t+1} \sim P(\cdot|\mathbf{x}_t)$, let y_{t+1} be its label
  $r_{t+1} = f^{-1}(y_t) - \gamma f^{-1}(y_{t+1})$ // f^{-1} converts label to logits
  $z_{t,td} = r_{t+1} + \gamma \mathbf{x}_{t+1}^\top \mathbf{w}_t$ // Bootstrap target, a separate target network is needed in DNNs
  $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \alpha(f(\mathbf{x}_t^\top \mathbf{w}_t) - f(z_{t,td}))\mathbf{x}_t$ // f converts logits back to label

We use the term *generalized* as it corresponds to generalized linear models in SL. This usage should be distinguished from a similarly named concept in van Hasselt et al. [50] within the realm of RL. Our approach applies RL to address problems typically considered within the scope of SL. This integration is non-trivial and unconventional, as SL is commonly viewed as a one-shot prediction problem. Instead, we have introduced a sequential perspective to SL. Moreover, the function $f(\cdot)$ in our model and the function $h(\cdot)$ in van Hasselt et al. [50] serve distinct purposes. In our framework, $f(\cdot)$ is employed to convert logits into appropriate outputs, such as softmax scores. In contrast, $h(\cdot)$ in van Hasselt et al. [50] and the logarithm used in van Seijen et al. [51] primarily address issues related to the scale of rewards.

4 MRP v.s. I.I.D. for Supervised Learning

As our MRP formulation is fundamentally different from the traditional i.i.d. view, this section discusses the property of TD's solution and its potential benefits in the linear setting and beyond. The section concludes with a discussion on the merits of adopting an MRP versus an i.i.d. perspective.

4.1 Connections between TD and OLS

The following proposition characterizes the connections between TD and OLS in the linear setting. Proof is in Appendix A.

Proposition 4.1. *[Connection between TD and OLS] When* $\mathbf{D}$ *has full support and* $\mathbf{X}$ *has linearly independent rows, TD and OLS have the same minimum norm solution. Moreover, any solution to the linear system* $\mathbf{X}\mathbf{w} = \mathbf{y}$ *must also be a solution to TD's linear system* $\mathbf{A}\mathbf{w} = \mathbf{b}$ *as defined in Equation* (4).

Empirical verification. Table 2 illustrates the distance between the closed-form solutions of our linear TD and OLS under various choices of the transition matrix by using a synthetic dataset (details are in Appendix B). For `Random`, each element of $\mathbf{P}$ is drawn from the uniform distribution $U(0, 1)$ and then normalized so that each row sums to one. The `Deficient` variant is exact the same as `Random`, except that the last column is set to all zeros before row normalization. This ensures that the last state is never visited from any state, thus not having

Table 2. Distance between the closed-form min-norm solutions of TD and OLS $\|\mathbf{w}_{TD} - \mathbf{w}_{LS}\|_2$. Input matrix $\mathbf{X}$ has normally distributed features with dimension $d \in \{70, 90, 110, 130\}$. Results are average over 10 runs with standard error in bracket. Details can be found in Appendix B.

Transition \ Dimension	70	90	110	130
Random	0.027 (± 0.01)	0.075 (± 0.02)	$\leq 10^{-10}$ (± 0.00)	$\leq 10^{-10}$ (± 0.00)
Uniform	0.026 (± 0.01)	0.074 (± 0.01)	$\leq 10^{-10}$ (± 0.00)	$\leq 10^{-10}$ (± 0.00)
Distance (Far)	0.028 (± 0.01)	0.075 (± 0.01)	$\leq 10^{-10}$ (± 0.00)	$\leq 10^{-10}$ (± 0.00)
Distance (Close)	0.182 (± 0.10)	0.249 (± 0.04)	$\leq 10^{-10}$ (± 0.00)	$\leq 10^{-10}$ (± 0.00)
Deficient	0.035 (± 0.01)	0.172 (± 0.04)	0.782 (± 0.18)	0.650 (± 0.25)

full support in its stationary distribution. `Uniform` simply means every element of $\mathbf{P}$ is set to $1/n$ where $n = 100$ is the number of training points. `Distance (Close)` assigns higher transition probability to points closer to the current point in the label space. Finally, `Distance (Far)` means contrary to `Distance (Close)`. The last two variants are used to see if similarity between points in the label space can play a role in the transition when using our TD algorithm.

Two key observations emerge: 1) As the feature dimension increases towards the overparameterization regime, both solutions become nearly indistinguishable, implying that designing $\mathbf{P}$ may be straightforward when employing a powerful model like NN. 2) Deficient choices for $\mathbf{P}$ with non-full support can pose issues and should be avoided. In practice, one might opt for a computationally and memory-efficient $\mathbf{P}$, such as a uniform constant matrix where every entry is set to $1/n$. Such a matrix is ergodic and, therefore, not deficient. We will delve deeper into the selection of $\mathbf{P}$ below.

4.2 Statistical Efficiency and Variance Analysis

A natural question is under what conditions the TD solution is better than the OLS solution, especially given that they may find the same solution under conditions specified above. Although the OLS estimator is known to be the best linear unbiased estimator (BLUE) under the i.i.d. assumption, our TD algorithm demonstrates the potential for a lower variance in settings with correlated noise, even when using a simple uniform transition matrix.

Recall that conventional linear models assume $y_i = \mathbf{x}_i^\top \mathbf{w}^* + \epsilon_i$, where $\mathbf{w}^*$ is the true parameters and ϵ_i is assumed to be independent noise. In contrast, under the MRP perspective, we consider the possibility of correlated noise. The following proposition shows when a TD estimator will be the most efficient one:

Proposition 4.2. *Suppose* $\mathbf{A}$ *as in Equation* (4) *is invertible and the error vector* $\boldsymbol{\epsilon}$ *satisfies* $\mathbb{E}[\boldsymbol{\epsilon}|\mathbf{X}] = \mathbf{0}$ *and* $\mathrm{Cov}[\boldsymbol{\epsilon}|\mathbf{X}] = \mathbf{C}$. *Then* $\mathbb{E}[\mathbf{w}_{TD}|\mathbf{X}] = \mathbf{w}^*$ *and the conditional covariance is*

$$\mathrm{Cov}[\mathbf{w}_{TD}|\mathbf{X}] = \mathbf{A}^{-1}\mathbf{X}^\top \mathbf{S}\mathbf{C}\mathbf{S}^\top \mathbf{X}(\mathbf{A}^{-1})^\top \tag{10}$$

where $\mathbf{S} \overset{def}{=} \mathbf{D}(\mathbf{I} - \gamma\mathbf{P})$. *Moreover, if* $\mathbf{S} = \mathbf{S}^\top$, *the TD estimator is the BLUE for problems with* $\mathbf{C} = c \cdot \mathbf{S}^{-1}, \forall c > 0$.

Proof. Recall from Equation (4) that $\mathbf{w}_{TD} = \mathbf{A}^{-1}\mathbf{b}$, $\mathbf{b} = \mathbf{X}^\top \mathbf{D}\mathbf{r}$, and $\mathbf{r} = (\mathbf{I} - \gamma\mathbf{P})\mathbf{y}$. Therefore $\mathbf{w}_{TD} - \mathbf{w}^*$ equals

$$\mathbf{w}_{TD} - \mathbf{w}^* = \mathbf{A}^{-1}(\mathbf{b} - \mathbf{A}\mathbf{w}^*) = \mathbf{A}^{-1}(\mathbf{X}^\top \mathbf{D}\mathbf{r} - \mathbf{A}\mathbf{w}^*) \tag{11}$$

$$= \mathbf{A}^{-1}(\mathbf{X}^\top \mathbf{D}(\mathbf{I} - \gamma\mathbf{P})\mathbf{y} - \mathbf{A}\mathbf{w}^*) \tag{12}$$

$$= \mathbf{A}^{-1}(\mathbf{X}^\top \mathbf{S}\mathbf{y} - \mathbf{X}^\top \mathbf{S}\mathbf{X}\mathbf{w}^*) = \mathbf{A}^{-1}\mathbf{X}\mathbf{S}\boldsymbol{\epsilon} \tag{13}$$

where we define $\mathbf{S} \stackrel{\text{def}}{=} \mathbf{D}(\mathbf{I} - \gamma\mathbf{P})$. When $\mathbb{E}[\boldsymbol{\epsilon}|\mathbf{X}] = \mathbf{0}$, the conditional expectation of the above equation is zero. Thus $\mathbf{w}_{TD}$ is conditionally unbiased and its conditional covariance is

$$\text{Cov}[\mathbf{w}_{TD}|\mathbf{X}] = \mathbf{A}^{-1}\mathbf{X}^\top\mathbf{S} \cdot \text{Cov}[\boldsymbol{\epsilon}|\mathbf{X}] \cdot \mathbf{S}^\top\mathbf{X}(\mathbf{A}^{-1})^\top \tag{14}$$

$$= \mathbf{A}^{-1}\mathbf{X}^\top\mathbf{S} \cdot \mathbf{C} \cdot \mathbf{S}^\top\mathbf{X}(\mathbf{A}^{-1})^\top \tag{15}$$

Finally, when $\mathbf{S} = \mathbf{S}^\top$ and $\mathbf{C} = c \cdot \mathbf{S}^{-1}$ for some $c > 0$, the TD estimator and its covariance become

$$\mathbf{w}_{TD} = (\mathbf{X}^\top\mathbf{S}\mathbf{X})^{-1}\mathbf{X}\mathbf{S}\mathbf{y} \qquad \text{Cov}[\mathbf{w}_{TD}|\mathbf{X}] = (c\mathbf{X}^\top\mathbf{S}\mathbf{X})^{-1}. \tag{16}$$

By using the Cholesky decomposition $\mathbf{S} = \mathbf{L}\mathbf{L}^\top$, one can see that the TD estimator is equivalent to the OLS solution to a rescaled problem $\widetilde{\mathbf{y}} = \widetilde{\mathbf{X}}\mathbf{w} + \widetilde{\boldsymbol{\epsilon}}$ where

$$\widetilde{\mathbf{y}} = \mathbf{L}^\top\mathbf{y} \qquad \widetilde{\mathbf{X}} = \mathbf{L}^\top\mathbf{X} \qquad \widetilde{\boldsymbol{\epsilon}} = \mathbf{L}^\top\boldsymbol{\epsilon} \tag{17}$$

Here $\text{Cov}[\widetilde{\boldsymbol{\epsilon}}|\mathbf{X}] = \mathbf{L}^\top\mathbf{C}\mathbf{L} = c\mathbf{I}$ so the TD estimator is the OLS solution to this problem and thus is the BLUE. □

Remark 1. This proposition identifies a situation in which our TD estimator outperforms other estimators, OLS included, in terms of efficiency. The condition $\mathbf{S} = \mathbf{S}^\top$ is needed so that there exists a corresponding symmetric covariance matrix $\mathbf{C}$ for the data. This condition implies that $\mathbf{D}\mathbf{P} = \mathbf{P}^\top\mathbf{D}$, or $D(s_i)P(s_j|s_i) = D(s_j)P(s_i|s_j)$, which means the Markov chain is reversible (i.e., detailed balance). Also note that $\mathbf{S}, \mathbf{A}$ are invertible under mild conditions (e.g., ergodic Markov chain). In such cases, the TD estimator also corresponds to the generalized least squares (GLS) estimator [1, 20]. In practice, one may be tempted to directly estimate the covariance matrix as done by feasible GLS (FGLS) [2]. However, estimating a covariance matrix is non-trivial as demonstrated in Section 6. Furthermore, it should be emphasized that GLS/FGLS methods do not naturally support incremental learning, nor are they readily adaptable to deep learning models.

Remark 2. The benefits of TD may not be limited to the situations described by the above proposition. It is challenging to mathematically describe these benefits because, under a more general $\mathbf{S}$, there is no intuitively interpretable form of the variance of TD's solution.

Furthermore, TD's benefits can be expanded by incorporating its more recent variants, such as emphatic TD [47] with transition-based γ and interest weighting, which may offer more flexibility in designing the matrix $\mathbf{S}$. Other variants include gradient TD [29] and quasi-second order accelerated TD [38, 36]. We leave these additional studies for future work.

Below, we provide a more general perspective to understand the benefits of TD in terms of variance reduction. The basic idea is that when the ground truth target variables of consecutive time steps are positively correlated, the TD target benefits from a reduction in variance.

Proposition 4.3. *Assume the estimated next-state value $\widehat{y}_{t+1}$ satisfies $\widehat{y}_{t+1} = \mathbb{E}[y_{t+1}] + \epsilon$ where ϵ is some independent noise with zero mean and standard deviation σ_ϵ. Let σ_t, σ_{t+1} be the standard deviations of y_t, y_{t+1} respectively, and $\rho_{t,t+1}$ be the Pearson correlation coefficient between y_t and y_{t+1}. If $\rho_{t,t+1} \geq \frac{\gamma^2(\sigma_{t+1}^2+\sigma_\epsilon^2)}{2\gamma\sigma_t\sigma_{t+1}}$, then $\text{Var}(y_{t,td}) \leq \text{Var}(y_t)$.*

Proof. We rewrite the TD target (8) as

$$y_{t,td} = (y_t - \gamma y_{t+1}) + \gamma\mathbb{E}[y_{t+1}] + \gamma\epsilon = y_t - \gamma(y_{t+1} - \mathbb{E}[y_{t+1}]) + \gamma\epsilon \tag{18}$$

This means we can treat y_{t+1} as a control variate and the variance of this estimate is

$$\text{Var}(y_{t,td}) = \text{Var}(y_t) + \gamma^2\text{Var}(y_{t+1}) - 2\gamma\text{Cov}(y_t, y_{t+1}) + \gamma^2\text{Var}(\epsilon) \tag{19}$$

$$= \sigma_t^2 + \gamma^2\sigma_{t+1}^2 - 2\gamma\rho_{t,t+1}\sigma_t\sigma_{t+1} + \gamma^2\sigma_\epsilon^2 \tag{20}$$

Plugging into the condition on $\rho_{t,t+1}$ would yield $\text{Var}(y_{t,td}) \leq \text{Var}(y_t)$. □

Remark. To better interpret the result, we can consider $\sigma_t = \sigma_{t+1}$, then the variance is simplified to

$$\mathrm{Var}(y_{t,td}) = \sigma_t^2 + \gamma^2\sigma_t^2 - 2\gamma\rho_{t,t+1}\sigma_t^2 + \gamma^2\sigma_\epsilon^2, \tag{21}$$

which is a convex function in γ and achieves its lowest value when

$$\gamma = \frac{\rho_{t,t+1}\sigma_t^2}{\sigma_t^2 + \sigma_\epsilon^2} \propto \rho_{t,t+1}. \tag{22}$$

This suggests that the stronger the correlation, the more (i.e., a larger γ) we might rely on the bootstrap term to reduce variance, coinciding with our intuition.

Empirical verification. Here we verify that when the outputs are indeed positively correlated, our method can generalize better than OLS. To this end, we run experiments using a Gaussian process with positive correlated outputs.

In each run, we jointly sample $n = 200$ data points (100 for training and 100 for test) from a Gaussian process where each element of the input matrix $\mathbf{X}$ is drawn from the standard normal distribution $\mathcal{N}(0, 1)$. The input dimension is $d = 70$. For the outputs, the mean function is given by $m(\mathbf{x}) = \mathbf{x}^\top \mathbf{w}^*$ where $\mathbf{w}^*$ is a vector of all ones and the covariance matrix is a block diagonal matrix

$$\mathbf{C} = \begin{bmatrix} \widetilde{\mathbf{C}} & & \\ & \ddots & \\ & & \widetilde{\mathbf{C}} \end{bmatrix}_{200\times 200} \quad \text{with} \quad \widetilde{\mathbf{C}} = \begin{bmatrix} 1 & \rho & \dots & \rho \\ \rho & 1 & \dots & \rho \\ \vdots & \vdots & \ddots & \vdots \\ \rho & \rho & \dots & 1 \end{bmatrix}_{10\times 10} \tag{23}$$

where ρ is a tuning parameter for the correlation. When $\rho > 0$, this structure ensures that all ten points within the same "cluster" are positively correlated. Finally, we add an independent noise (with zero mean and standard deviation of 0.1) to each output before using them for training and testing.

For our TD method, we set $\gamma = 0.99$ and design the probability matrix as an interpolation between a covariance matrix $\mathbf{C}$ and $\mathbf{1} - \mathbf{C}$, defined as $\mathbf{P} = (1 - \eta)(\mathbf{1} - \mathbf{C}) + \eta\mathbf{C}$, followed by normalization to ensure that it forms a valid stochastic matrix. We vary η over the set {0.5, 0.6, 0.7, 0.8, 0.9} and the correlation coefficient ρ over {0.1, 0.3, 0.5, 0.7, 0.9}. As η gets closer to one, the transition matrix agrees more with the covariance matrix so that it is more likely to transition to positively correlated points. With 100 training points, we learn both the TD solution $\mathbf{w}_{TD}$ and OLS solution $\mathbf{w}_{LS}$ and plot their test RMSE. The experiment is repeated 50 times and Figure 1 reports the mean and standard deviation for different (ρ, η) values.

The results are shown in Figure 1. Under different levels of correlation, as indicated by various colors, TD (solid lines) increasingly outperforms OLS (dashed lines) when the transition probability matrix $\mathbf{P}$ more closely aligns with the covariance. $\mathbf{P}$ is designed so that a higher transition probability is assigned between two points when their outputs exhibit a stronger positive correlation. Although there is no direct theoretical result explicitly proving the degradation of OLS performance under increasingly correlated data, existing works, such as Lemma 3 in Song et al. [45] and Proposition 4 in Panaganti et al. [39], indirectly suggest that its generalization error may be higher than in the i.i.d. setting. This is because their upper bounds on the error include a linear dependence on either the number of steps or the sample size, whereas in the i.i.d. case, the dependence is typically less costly.

Practical choices: MRP view v.s. i.i.d. view. We conclude this section by discussing the practical considerations on either using the MRP or i.i.d. view. In machine learning, the selection of algorithms often depends on whether their underlying assumptions align with the ground truth of the data – a factor that is fundamentally unknowable. Thus, the decision often boils down to the user's belief – whether to treat the data as MRP or i.i.d. It is noteworthy that the TD algorithm demonstrates greater generality as it: 1) accommodates the solution of the linear system as demonstrated in Proposition 4.1 and is equivalent to OLS under certain conditions; 2) offers a

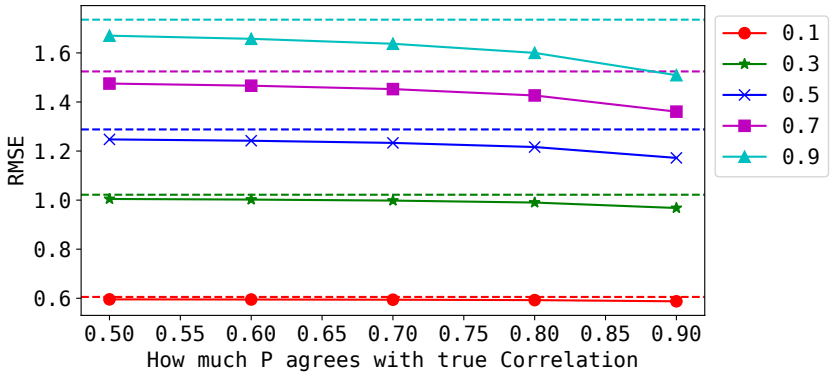


Fig. 1. Comparing TD (solid) with OLS (dashed) with different settings.

straightforward approach to setting the discount factor to zero, effectively reducing the bootstrap target to the original SL training target.

The theoretical and empirical results from this section suggest: 1) in the absence of prior knowledge, adopting TD with small γ values and a uniform $\mathbf{P}$ could be beneficial for computational and memory efficiency since we can simply pick the next data point uniformly at random without storing specific probabilities; if the ground truth suggests that the data points are positively correlated, this approach might yield a performance gain; 2) when it is known that two points are positively correlated, one could enhance variance reduction by strategically encouraging transition from one point to another.

5 Convergence Analysis

In this section, we present convergence results for our generalized TD algorithm (Algorithm 1) under both the expected updating rule and the sample-based updating rule. Detailed proofs are provided in Appendix A.

We show that the expected updating rule of our generalized TD algorithm can be viewed as a generalized Bellman operator (25) that is contractive. Based on this, we further prove finite-time convergence when applying TD(0) updates with linear function approximation. We primarily follow the convergence framework presented in Bhandari et al. [6], making non-trivial adaptations due to the presence of the inverse link function. We start with the assumptions.

Assumption 5.1 (Feature regularity). $\forall s \in \mathcal{S}, \|\mathbf{x}(s)\|_2 \leq 1$ and the steady-state covariance matrix $\Sigma \stackrel{\text{def}}{=} D(s)\mathbf{x}(s)\mathbf{x}(s)^\top$ has full rank.

Assumption 5.1 is a typical assumption necessary for the existence of the fixed point when there is no transform function [49].

Assumption 5.2. The inverse link function $f : \mathbb{R} \mapsto \mathcal{Y}$ is continuous, invertible and strictly increasing. Moreover, it has a bounded derivative f' on any bounded domain.

Remark. Assumption 5.2 is satisfied for those inverse link functions commonly used in GLMs, including but not limited to identity function (linear regression), exponential function (Poisson regression), and sigmoid function (logistic regression) [31, Table 2.1].

For theoretical analysis, we consider training in a convex and compact set $\mathcal{W} \subset \mathbb{R}^d$ such that the parameters are projected back to it after each update (see Equations (25) and (32)). Without this, one may need to consider

certain special cases where the parameters $\mathbf{w}$ may diverge to infinity, akin to the case of (unconstrained) logistic regression. With this in mind, the following lemma holds.

Lemma 5.3. *Under Assumption 5.1, 5.2, for $\mathbf{w} \in \mathcal{W}$, there exists $L \geq 1$ such that $\forall s_1, s_2 \in \mathcal{S}$ with $z_1 = \mathbf{x}(s_1)^\top \mathbf{w}$, $z_2 = \mathbf{x}(s_2)^\top \mathbf{w}$,*

$$\frac{1}{L}|z_1 - z_2| \leq |f(z_1) - f(z_2)| \leq L|z_1 - z_2|. \tag{24}$$

This lemma is important for relating the bounds between the logit space and the output/label space. Let $z(s) = \mathbf{x}(s)^\top \mathbf{w}$, or $z = \mathbf{x}^\top \mathbf{w}$ for conciseness in the following. The next assumption is necessary later to ensure that the step size is positive.

Assumption 5.4 (Bounded discount)**.** The discount factor satisfies $\gamma < \frac{1}{L^2}$ for the L in Lemma 5.3.

Given these assumptions, Section 5.1 shows that the projected expected update admits a unique fixed point $\mathbf{w}^* \in \mathcal{W}$, Section 5.2 proves finite-time convergence bounds for the projected expected update, and finally, Section 5.3 proves the convergence bound of sample-based update.

5.1 Contraction of Expected Update

The expected update followed by a projection onto $\mathcal{W}$ is given by

$$\mathbf{w}_{t+1} = \mathcal{P}(\mathbf{w}_t + \alpha \overline{g}(\mathbf{w}_t)) \tag{25}$$

where $\overline{g}(\mathbf{w}_t) \stackrel{\text{def}}{=} \mathbb{E}[(y_{td} - y)\mathbf{x}]$ is the expected update, $\mathcal{P}(\mathbf{w}) \stackrel{\text{def}}{=} \operatorname{argmin}_{\mathbf{u} \in \mathcal{W}} \|\mathbf{w} - \mathbf{u}\|_2^2$ is the projection, $y \stackrel{\text{def}}{=} f(z), y_{td} \stackrel{\text{def}}{=} f(r + \gamma \mathbf{x}'^\top \mathbf{w}_t)$ and $\mathbf{x}' \stackrel{\text{def}}{=} \mathbf{x}(s')$. This updating rule – the mapping from $\mathbf{w}_t$ to $\mathbf{w}_{t+1}$ – can be viewed as a type of *generalized Bellman operator*. The term "generalized" refers to its ability to incorporate different types of f functions, enabling it to accommodate various learning tasks, such as regression and classification.

We show the existence of a fixed point $\mathbf{w}^* \in \mathcal{W}$ for our update using the Banach fixed-point theorem. Given two iterates $\mathbf{w}_t^A, \mathbf{w}_t^B \in \mathcal{W}$ and their updates

$$\mathbf{w}_{t+1}^A = \mathcal{P}(\mathbf{w}_t^A + \alpha \overline{g}(\mathbf{w}_t^A)) \qquad \mathbf{w}_{t+1}^B = \mathcal{P}(\mathbf{w}_t^B + \alpha \overline{g}(\mathbf{w}_t^B)) \tag{26}$$

we will show that any two parameters $\mathbf{w}^A, \mathbf{w}^B \in \mathcal{W}$ will get closer after each iteration.

Note that

$$\|\mathbf{w}_{t+1}^A - \mathbf{w}_{t+1}^B\|_2^2 \leq \|\mathbf{w}_t^A + \alpha \overline{g}(\mathbf{w}_t^A) - (\mathbf{w}_t^B + \alpha \overline{g}(\mathbf{w}_t^B))\|_2^2 \tag{27}$$

since projection $\mathcal{P}$ onto a convex set is contracting. Furthermore, we can expand it as

$$\begin{aligned}\|\mathbf{w}_{t+1}^A - \mathbf{w}_{t+1}^B\|_2^2 \leq \|\mathbf{w}_t^A - \mathbf{w}_t^B\|_2^2 - 2\alpha(\mathbf{w}_t^A - \mathbf{w}_t^B)^\top(\overline{g}(\mathbf{w}_t^B) - \overline{g}(\mathbf{w}_t^A)) \\ + \alpha^2 \|\overline{g}(\mathbf{w}_t^A) - \overline{g}(\mathbf{w}_t^B)\|_2^2\end{aligned} \tag{28}$$

The common strategy is to ensure that the second term can outweigh the third term so that the distance decreases after each iteration. To simplify notation, denote their predictions $z^A = \mathbf{x}^\top \mathbf{w}^A, z^B = \mathbf{x}^\top \mathbf{w}^B$, transformed predictions $y^A = f(z^A), y^B = f(z^B)$ and bootstrap targets $y_{td}^A = f(r + \mathbf{x}'^\top \mathbf{w}^A), y_{td}^B = f(r + \mathbf{x}'^\top \mathbf{w}^B)$. The following two lemmas bound the two terms respectively.

Lemma 5.5. *For $\mathbf{w}^A, \mathbf{w}^B \in \mathcal{W}$, $(\mathbf{w}^A - \mathbf{w}^B)^\top(\overline{g}(\mathbf{w}^B) - \overline{g}(\mathbf{w}^A)) \geq \left(\frac{1}{L} - \gamma L\right) \cdot \mathbb{E}\left[(z^A - z^B)^2\right]$.*

Lemma 5.6. *For $\mathbf{w}^A, \mathbf{w}^B \in \mathcal{W}$, $\|\overline{g}(\mathbf{w}^A) - \overline{g}(\mathbf{w}^B)\|_2 \leq 2L\sqrt{\mathbb{E}\left[(z^A - z^B)^2\right]}$*

With these lemmas, we can prove the following contraction theorem.

Theorem 5.7. *[Contraction of Expected Update] Under Assumption 5.1-5.4, by choosing* $\alpha = \frac{1-\gamma L^2}{4L^3} > 0$*, the projected update (25) is a contraction w.r.t.* $\|\cdot\|_2$ *with a unique fixed point* $\mathbf{w}^* \in \mathcal{W}$*, and* $\mathbf{w}_t$ *converges to it.*

With this well-defined fixed point, we will analyze the finite-time convergence rate next.

5.2 Convergence Rate of Expected Update

Now we prove the convergence rate. The distance from $\mathbf{w}_{t+1}$ to $\mathbf{w}^*$ can be expanded as

$$\|\mathbf{w}^* - \mathbf{w}_{t+1}\|_2^2 = \|\mathbf{w}^* - \mathbf{w}_t\|_2^2 - 2\alpha(\mathbf{w}^* - \mathbf{w}_t)^\top \overline{g}(\mathbf{w}_t) + \alpha^2 \|\overline{g}(\mathbf{w}_t)\|_2^2 \tag{29}$$

Similarly to before, we want to ensure that the second term can outweigh the third term in the RHS so that $\mathbf{w}_{t+1}$ can get closer to $\mathbf{w}^*$ than $\mathbf{w}_t$ in each iteration. This is achieved again by applying Lemma 5.5 and Lemma 5.6, which leads to the following result:

Theorem 5.8. *[Convergence Rate of Expected Update] Under Assumption 5.1-5.4, consider the sequence* $(\mathbf{w}_0, \mathbf{w}_1, \cdots)$ *satisfying Equation (25). Let* $\overline{\mathbf{w}}_T \overset{def}{=} \frac{1}{T}\sum_{t=0}^{T-1} \mathbf{w}_t$*,* $\overline{z}_T = \mathbf{x}^\top \overline{\mathbf{w}}_T$ *and* $z^* = \mathbf{x}^\top \mathbf{w}^*$*. By choosing* $\alpha = \frac{1-\gamma L^2}{4L^3} > 0$*, we have*

$$\mathbb{E}\left[(z^* - \overline{z}_T)^2\right] \le \left(\frac{2L^2}{1-\gamma L^2}\right)^2 \frac{\|\mathbf{w}^* - \mathbf{w}_0\|_2^2}{T} \tag{30}$$

$$\|\mathbf{w}^* - \mathbf{w}_T\|_2^2 \le \exp\left(-T\omega \left(\frac{1-\gamma L^2}{2L^2}\right)^2\right) \|\mathbf{w}^* - \mathbf{w}_0\|_2^2 \tag{31}$$

Equation (30) shows that in expectation, the average prediction converges to the true value in the z-space, while Equation (31) shows that the last iterate converges to the fixed point exponentially fast when using expected update. In practice, a sample-based update is preferred and we discuss its convergence next.

5.3 Convergence Rate of Sample-based Update

Here we show the convergence under i.i.d. sample setting. Suppose s_t is sampled from the stationary distribution $D(s)$ and $s_{t+1} \sim P(\cdot|s_t)$. For conciseness, let $\mathbf{x}_t \overset{\text{def}}{=} \mathbf{x}(s_t)$ and $\mathbf{x}_{t+1} \overset{\text{def}}{=} \mathbf{x}(s_{t+1})$. The sample-based updating rule is

$$\mathbf{w}_{t+1} = \mathcal{P}(\mathbf{w}_t + \alpha_t g_t(\mathbf{w}_t)) \quad \text{with} \quad g_t(\mathbf{w}_t) \overset{\text{def}}{=} (y_{t,td} - y_t)\mathbf{x}_t \tag{32}$$

where $y_t \overset{\text{def}}{=} f(\mathbf{x}_t^\top \mathbf{w}_t)$, $y_{t,td} \overset{\text{def}}{=} f(z_{t,td}) = f(r_{t+1} + \gamma \mathbf{x}_{t+1}^\top \mathbf{w}_t)$. The t in g_t is for the sample $(\mathbf{x}_t, y_t)$, while the $\mathbf{w}_t$ in $g_t(\mathbf{w}_t)$ is for calculating y_t and $y_{t,td}$.

To account for randomness, let $\sigma^2 \overset{\text{def}}{=} \mathbb{E}[\|g_t(\mathbf{w}^*)\|_2^2]$, the variance of the TD update at the stationary point $\mathbf{w}^*$ under the stationary distribution. We need two supporting lemmas to prove the convergence. The first shows that there is an optimality condition for $\mathbf{w}^*$ similar to the first-order condition in constrained convex optimization (Lemma 5.9), and the second bounds the expected norm of the update (Lemma 5.10).

For the first lemma, note that the iterative update (25) does not apply gradient descent on any fixed objective, so one cannot simply apply the first-order optimality condition from constrained optimization [8, Sec.4.2.3]. Nevertheless, we can still prove the condition due to the projection step:

Lemma 5.9. *The fixed point* $\mathbf{w}^* \in \mathcal{W}$ *in Theorem 5.7 satisfies*

$$(\mathbf{w}^* - \mathbf{w})^\top \overline{g}(\mathbf{w}^*) \ge 0 \quad \forall \mathbf{w} \in \mathcal{W}. \tag{33}$$

The next lemma is similar to Lemma 5.6 but for the sample-based update:

Lemma 5.10. *For* $\mathbf{w} \in \mathcal{W}$*,* $\mathbb{E}[\|g_t(\mathbf{w})\|_2^2] \le 2\sigma^2 + 8L^2\mathbb{E}[(z_t^* - z_t)^2]$ *where* $\sigma^2 = \mathbb{E}[\|g_t(\mathbf{w}^*)\|_2^2]$*.*

Now we are ready to present the convergence when using i.i.d. sample for the update:

Theorem 5.11. *[Convergence Rate of Sampled-based Update] Under Assumption 5.1-5.4, with sample-based update Equation (32), let* $\sigma^2 = \mathbb{E}[\|g_t(\mathbf{w}^*)\|_2^2]$, $\overline{\mathbf{w}}_T \stackrel{def}{=} \frac{1}{T}\sum_{t=0}^{T-1}\mathbf{w}_t$, $\overline{z}_T = \mathbf{x}^\top\overline{\mathbf{w}}_T$ *and* $z^* = \mathbf{x}^\top\mathbf{w}^*$. *For* $T \geq \frac{64L^6}{(1-\gamma L^2)^2}$ *and a constant step size* $\alpha_t = 1/\sqrt{T}, \forall t$, *we have*

$$\mathbb{E}\left[(z^* - \overline{z}_T)^2\right] \leq \frac{L\left(\|\mathbf{w}^* - \mathbf{w}_0\|_2^2 + 2\sigma^2\right)}{\sqrt{T}(1-\gamma L^2)}. \tag{34}$$

This shows that the generalized TD update converges even when using the sample-based update. Not surprisingly, it is slower than using the expected update (30).

6 Empirical Studies

This section has two primary objectives: first, to validate whether the theoretical results of variance analysis are applicable to real-world datasets in both linear and neural network (NN) settings; second, to investigate the practical utility of our algorithm within a standard learning setting. To accomplish these goals, we have structured the following content into four subsections, respectively: 1) In the linear setting, we examine synthetic noise added to a real-world dataset. 2) In a NN setting, we explore the impact of synthetic noise on a real-world dataset. 3) In a NN setting, we assess our algorithm's performance using a dataset where the training targets may intuitively be correlated. 4) We evaluate our algorithm on commonly used real-world datasets from regression to image classification.

Our conclusions are as follows: 1) TD learning performs in line with our theoretical expectations in either linear or neural network settings with correlated noise; 2) in a standard SL setting, TD demonstrates performance on par with traditional SL methods. Results on additional data and any missing details are in Appendix C.

Setup overview. For regression with real targets, we adopt the popular dataset execution time [40]. For image classification, we use the popular MNIST [24], Fashion-MNIST [54], Cifar10 and Cifar100 [22] datasets. Additional results on other popular datasets (e.g. house price, bike rental, weather, insurance) are in Appendix C. In TD algorithms, unless otherwise specified, we use a fixed transition probability matrix with all entries set to $1/n$, which is memory and computation efficient, and simplifies sampling processes.

Baselines and naming rules. TDReg: our TD approach, with its direct competitor being Reg (conventional l_2 regression). Reg-WP: Utilizes the same probability transition matrix as TDReg but does not employ bootstrap targets. This baseline can be used to assess the effect of bootstrap and transition probability matrix.

6.1 Linear Setting with Correlated Noise

As outlined in Section 4.2, we anticipate that in scenarios where the target noise exhibits positive correlation, TD learning should leverage the advantages of using a transition matrix. When this matrix transitions between points with correlated noise, the TD target counterbalances the noise, thereby reducing variance. As depicted in Figure 2, we observe that with an increasing correlation coefficient (indicative of progressively positively correlated noise), TD consistently shows improved generalization performance towards the underlying best baseline.

In these figures, alongside TD and OLS, we include two baselines known for their efficacy in handling correlated noise: generalized least squares (GLS) and feasible GLS (FGLS) [1]. It is noteworthy that GLS operates under the assumption that the covariance of noise generation is fully known, aiming to approximate the underlying optimal estimator $\mathbf{w}^*$. FGLS has two procedures: estimate the covariance matrix followed by applying this estimation to resolve the linear system. Please refer to Appendix C.2 for details.

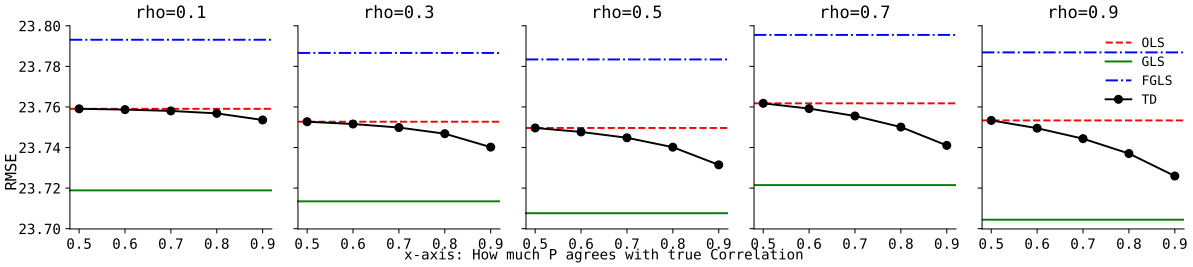


Fig. 2. Test root mean squared error (RMSE) versus different Ps on exec time dataset. From left to right, the noise correlation coefficient increases ($\rho \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$). Each plot's x-axis represents the degree of alignment between the transition matrix P and the true covariance matrix that generates the noise. As P aligns better – implying a higher likelihood of data points with positively correlated noise transitioning from one to another – the solution of TD increasingly approximates the $\mathbf{w}^*$. Furthermore, as the correlation among the data intensifies, one can see larger gap between TD and OLS/FGLS.

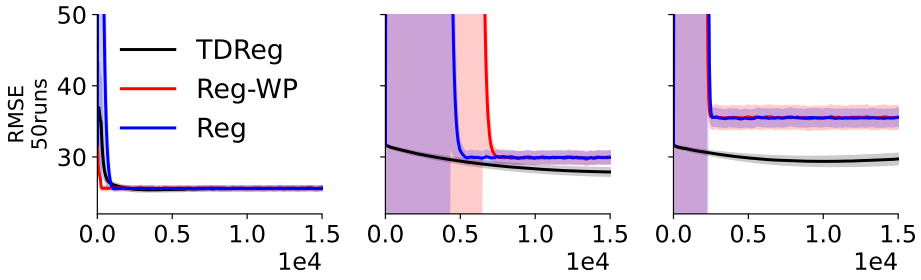


Fig. 3. Learning curves on execute time dataset. From left to right, the noise defined as $c \times \epsilon$ where ϵ is noise generated by GP process and $c \in \{10, 20, 30\}$. Due to the addition of large-scale noise, Reg-WP and Reg exhibit high instability during the early learning stages and ultimately converge to a suboptimal solution compared to our TD (in black line) approach.

6.2 Deep learning: Regression with Synthetic Correlated Noise

Similar to the previous experiment, we introduced noise to the original training target using a GP and opted for a uniform transition matrix. The results, depicted in Figure 3, reveal two key observations: 1) As the noise level increases, the performance advantage of TD over the baselines becomes more pronounced; 2) The baseline Reg-WP performs just as poorly as conventional Reg, underscoring the pivotal role of the TD target in achieving performance gains. Additionally, it was observed that all algorithms select the same optimal learning rate even when smaller learning rates are available, and TD consistently chooses a large $\gamma = 0.95$. This implies that TD's stable performance is attributed NOT to the choice of a smaller learning rate, but rather to the bootstrap target. Additional results are in Figure 7.

6.3 Deep Learning without Synthetic Noise

In order to test the practical utility of our algorithm, we evaluate it using various transition matrices on an air quality dataset [52], without introducing any synthetic noise to alter the original data. This dataset's task is to predict CO concentration. The main objective here is to demonstrate that real-world problems might also feature positively correlated data, where our method could outperform SGD.

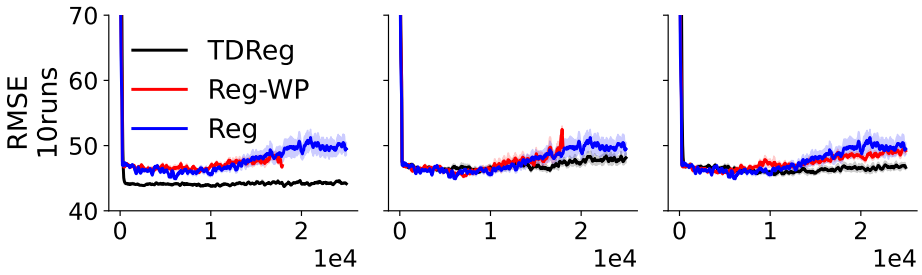


Fig. 4. We show the test RMSE with error vs. the number of training steps when using three types of transition matrices. From left to right, the first type of transition assigns a high probability of transitioning to similar data points, the second assigns a high probability of transitioning to distant points, and the third one uses an uniform constant transition matrix as described before. The results are averaged over 10 random seeds.

Although it is impossible to determine the underlying ground truth of the noise structure precisely, we propose the following intuition for potential positive correlations among data points in this dataset.

If air quality measurements are taken at regular intervals (e.g., hourly or daily) at the same location, temporal dependencies might arise between consecutive measurements due to factors such as diurnal variations, weather patterns, or pollution sources. Consequently, the CO concentration measured at one time point may be positively correlated with that measured at adjacent times. Additionally, spatial dependencies could occur if measurements are taken at nearby locations, particularly in densely populated urban areas with unevenly distributed pollution sources. In such cases, neighboring locations may experience similar air quality conditions, resulting in a positive correlation between their CO concentrations.

Figure 4 shows the learning curves of TD and baselines using different types of transition matrix P. We believe the experiments shed significant light on our approach, based on the following observations:

1. When dealing with potentially correlated data, the conventional SGD algorithm (Reg) may diverge or settle on a suboptimal solution, as it theoretically requires i.i.d. data to converge. In our experiments, SGD did not perform well.

2. Our algorithm (TDReg) demonstrates optimal performance when using a transition matrix P that favors transitions between closer points. It performs less well when this preference is reversed and shows only a modest improvement over SGD with a uniform matrix. This is possibly because the entire dataset is somewhat positively correlated, and the use of a bootstrap target helps mitigate the effect of noise.

3. The comparison with baseline Reg-WP, which differs from Reg only in its use of the same transition matrix as TDReg, aims to isolate the effect of the bootstrap target. If Reg-WP performed as well as TDReg, it would suggest that the benefits of TDReg stem solely from the transition matrix rather than the bootstrap mechanism. However, the experiments show that without the bootstrap, Reg-WP may diverge (as indicated by the disappearance of the red line in the figures), likely because the transition matrix P further increases data point correlation, causing the conventional algorithm to fail.

As an additional note, to exclude the possibility of divergence due to excessively high learning rates in the baseline algorithms, we tested with sufficiently small learning rates. Despite this, the baselines did not perform better, indicating that the divergence was not due to the selection of a larger learning rate of TD.

Table 3. Test root mean squared error (RMSE) with standard error. The results have been smoothed using a 5-point window before averaging over 5 runs.

Alg. / Data	**TD-Reg**	**Reg-WP**	**Reg**
house	3.384 ± 0.21	3.355 ± 0.23	3.319 ± 0.16
exectime	23.763 ± 0.38	23.794 ± 0.36	23.87 ± 0.36
bikeshare	40.656 ± 0.77	40.904 ± 0.36	40.497 ± 0.45

6.4 Deep Learning: Standard Problems

This section aims at investigating how our TD algorithm works on commonly used datasets where there may not be correlation among targets. We found that there is no clear gain/lose with TD's bootstrap target in either regression (Table 3) or image classification tasks (Table 4). For image datasets, we employed a convolutional neural network (CNN) architecture to demonstrate TD's practical utility.

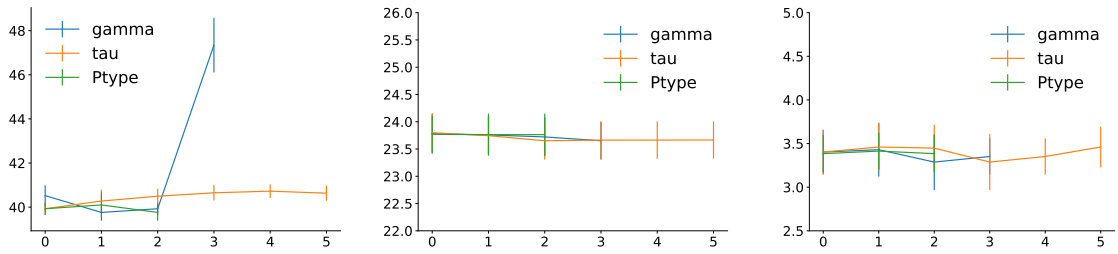
Table 4. Image classification test accuracy. The results are smoothed over 10 evaluations before averaging over 3 random seeds. The standard errors are small with no definitive advantage observed for either algorithm.

Algs / Dataset	**TD-Classify**	**Classify**
mnist	99.06%	99.00%
mnistfashion	89.10%	88.96%
cifar10	67.42%	67.13%
cifar100	31.55%	31.21%

We then further study the hyperparameter sensitivity in Figure 5. With NNs, TD can pose challenges due to the interplay of two additional hyperparameters: γ and the target NN moving rate τ [32]. Optimizing these hyperparameters can often be computationally expensive. Therefore, we investigate their impact on performance by varying γ, τ. We find that selecting appropriate parameters tends to be relatively straightforward. Figure 5 displays the testing performance across various parameter settings. It is evident that performance does not vary significantly across settings, indicating that only one hyperparameter or a very small range of hyperparameters needs to be considered during the hyperparameter sweep. It should be noted that Figure 5 also illustrates the sensitivity analysis when employing three intuitive types of transition matrices that do not require prior knowledge. It appears that there is no clear gain/lose with these choices in a standard task. We refer readers to Appendix C.5 for details.

7 Conclusion

This paper introduces a universal framework that transforms traditional SL problems into RL problems, alongside proposing a generalized TD algorithm. We highlight a specific set of problems for which such algorithms are apt and the potential variance reduction benefits of TD. Additionally, we theoretically formalize the iteration of our generalized TD learning algorithm as a special case of a generalized Bellman operator that is contractive and admits a unique fixed point. Based on this, we further establish the convergence properties of the algorithm. Empirical evidence, encompassing both linear and deep learning contexts, is provided to validate our theoretical findings and to evaluate the algorithm's design and practical applicability. This research represents a foundational step towards bridging classic SL and RL paradigms. Our MRP formulation essentially offers a perspective that links various data points, corresponding to an interconnected worldview.



(a) Bikeshare (b) ExecutionTime (c) House

Fig. 5. Sensitivity to hyperparameter settings. We show test RMSE with error bars while sweeping over discount rate $\gamma \in \{0.1, 0.2, 0.4, 0.9\}$, target NN moving rate $\tau \in \{0.001, 0.01, 0.1, 0.2, 0.4, 0.9\}$, and three types of transition probability. γ hurts when it goes to the largest value on bikeshare, and it is likely because the exponential term in Poisson regression needs a unusually small learning rate. When generating curves for τ and γ, we maintain P_{type} as a simple uniform constant.

Future work and limitations. Our current analysis of covariance does not account for the potential advantages in scenarios where the matrix $\mathbf{S}$ is non-symmetric. This gap is due to both the absence of relevant literature and the lack of interpretable closed-form expressions, as the non-symmetric matrix can no longer be interpreted as a covariance matrix. It is plausible that specific types of transition matrix designs could be advantageous in certain applications – particularly when they align with the underlying data assumptions. Additionally, our discussion does not encompass the potential benefits of incorporating more recent TD algorithms such as emphatic TD [47], gradient TD [29], or quasi second-order accelerated TD [55, 38, 36], which have been shown to enhance stability or convergence. Furthermore, it would be more intriguing to explore the utility of the probability transition matrix in a broader context, such as transfer learning, domain adaptation, or continual learning settings. Lastly, extending our approach to test its effectiveness with more modern NNs, such as transformers, could be of significant interest to a broader community.

Acknowledgments

Yangchen Pan would like to acknowledge the support from Tuing World leading fellow. Junfeng Wen would like to acknowledge the support from NSERC. Philip Torr is supported by the UKRI grant: Turing AI Fellowship. Philip Torr would also like to thank the Royal Academy of Engineering and FiveAI.

References

[1] A. C. Aitken. 1936. Iv.—on least squares and linear combination of observations. *Proceedings of the Royal Society of Edinburgh*, 55, 42–48.
[2] B. H. Baltagi. 2008. *Econometrics. Springer Books*. Springer.
[3] A. Banerjee, S. Merugu, I. S. Dhillon, J. Ghosh, and J. Lafferty. 2005. Clustering with bregman divergences. *Journal of machine learning research*, 6, 10.
[4] A. M. Barreto, D. Precup, and J. Pineau. 2016. Practical kernel-based reinforcement learning. *Journal of Machine Learning Research*.
[5] A. G. Barto and T. G. Dietterich. 2004. Reinforcement learning and its relationship to supervised learning. *Handbook of Learning and Approximate Dynamic Programming*, 45–63.
[6] J. Bhandari, D. Russo, and R. Singal. 2018. A finite time analysis of temporal difference learning with linear function approximation. In *Conference on learning theory*. PMLR, 1691–1692.
[7] C. M. Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer.
[8] S. P. Boyd and L. Vandenberghe. 2004. *Convex optimization*. Cambridge university press.
[9] S. J. Bradtke and A. G. Barto. 1996. Linear least-squares algorithms for temporal difference learning. *Machine learning*, 22, 1, 33–57.

[10] D. Brandfonbrener, A. Bietti, J. Buckman, R. Laroche, and J. Bruna. 2022. When does return-conditioned supervised learning work for offline reinforcement learning? *Advances in Neural Information Processing Systems*.

[11] T. Company. 2021. Travel insurance data. (2021). https://www.kaggle.com/datasets/tejashvi14/travel-insurance-prediction-data/data.

[12] T. Dietterich and X. Wang. 2001. Batch value function approximation via support vectors. In *Advances in Neural Information Processing Systems*. Vol. 14.

[13] H. Fanaee-T and J. Gama. 2014. Event labeling combining ensemble detectors and background knowledge. *Progress in Artificial Intelligence*, 2, 113–127.

[14] D. Ghosh, A. Gupta, A. Reddy, J. Fu, C. M. Devin, B. Eysenbach, and S. Levine. 2021. Learning to reach goals via iterated supervised learning. In *International Conference on Learning Representations*.

[15] T. N. E. Greville. 1966. Note on the generalized inverse of a matrix product. *Siam Review*, 8, 4, 518–521.

[16] D. Gupta, G. Mihucz, M. Schlegel, J. Kostas, P. S. Thomas, and M. White. 2021. Structural credit assignment in neural networks using reinforcement learning. In *Advances in Neural Information Processing Systems*, 30257–30270.

[17] D. P. Helmbold, J. Kivinen, and M. K. Warmuth. 1999. Relative loss bounds for single neurons. *IEEE Transactions on Neural Networks*, 10, 6, 1291–1304.

[18] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne. 2017. Imitation learning: a survey of learning methods. *ACM Computing Surveys (CSUR)*, 50, 2, 1–35.

[19] A. (Joe Young (Owner). 2020. Rain in australia, copyright commonwealth of australia 2010, bureau of meteorology. (2020).

[20] T. Kariya and H. Kurata. 2004. *Generalized least squares*. John Wiley & Sons.

[21] D. P. Kingma and J. Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR*.

[22] A. Krizhevsky. 2009. Learning multiple layers of features from tiny images. Tech. rep. University of Toronto.

[23] M. Kubat, S. Matwin, et al. 1997. Addressing the curse of imbalanced training sets: one-sided selection. *International Conference on Machine Learning*, 97, 179.

[24] Y. LeCun, C. Cortes, and C. Burges. 2010. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2.

[25] L. Lee, B. Eysenbach, R. Salakhutdinov, S. S. Gu, and C. Finn. 2020. Weakly-supervised reinforcement learning for controllable behavior. In *Advances in Neural Information Processing Systems*.

[26] M. Lichman. 2015. UCI machine learning repository. (2015). http://archive.%20ics.%20uci.%20edu/ml.

[27] E. Lin, Q. Chen, and X. Qi. 2020. Deep reinforcement learning for imbalanced classification. *Applied Intelligence*, 50, 8, 2488–2502.

[28] J. MacGlashan, M. K. Ho, R. Loftin, B. Peng, G. Wang, D. L. Roberts, M. E. Taylor, and M. L. Littman. 2017. Interactive learning from policy-dependent human feedback. In *International Conference on Machine Learning*, 2285–2294.

[29] H. R. Maei. 2011. *Gradient temporal-difference learning algorithms*. Ph.D. Dissertation. University of Alberta.

[30] Martín Abadi, Ashish Agarwal, Paul Barham, and et al. 2015. TensorFlow: large-scale machine learning on heterogeneous systems. Software available from tensorflow.org. (2015).

[31] P. McCullagh and J. A. Nelder. 1989. *Generalized Linear Models*. Vol. 37. CRC Press.

[32] V. Mnih et al. 2015. Human-level control through deep reinforcement learning. *nature*, 518, 7540, 529–533.

[33] J. Nelder. 1974. Log linear models for contingency tables: a generalization of classical least squares. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 23, 3, 323–329.

[34] J. A. Nelder and R. W. Wedderburn. 1972. Generalized linear models. *Journal of the Royal Statistical Society Series A: Statistics in Society*, 135, 3, 370–384.

[35] D. Ormoneit and Ś. Sen. 2002. Kernel-based reinforcement learning. *Machine Learning*, 161–178.

[36] Y. Pan, E. S. Azer, and M. White. 2017. Effective sketching methods for value function approximation. In *Conference on Uncertainty in Artificial Intelligence*.

[37] Y. Pan, E. Imani, A.-m. Farahmand, and M. White. 2020. An implicit function learning approach for parametric modal regression. *Advances in Neural Information Processing Systems*, 33, 11442–11452.

[38] Y. Pan, A. White, and M. White. 2017. Accelerated gradient temporal difference learning. *AAAI Conference on Artificial Intelligence*.

[39] K. Panaganti, A. Wierman, and E. Mazumdar. 2024. Model-free robust φ-divergence reinforcement learning using both offline and online data. *International Conference on Machine Learning*.

[40] E. Paredes and R. Ballester-Ripoll. 2018. SGEMM GPU kernel performance. UCI Machine Learning Repository. (2018).

[41] A. Paszke et al. 2017. Automatic differentiation in pytorch. *NIPS Autodiff Workshop*.

[42] J. Perdomo, T. Zrnic, C. Mendler-Dünner, and M. Hardt. 2020. Performative prediction. In *International Conference on Machine Learning*. PMLR, 7599–7609.

[43] M. L. Puterman. 2014. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

[44] S. Seabold and J. Perktold. 2010. Statsmodels: econometric and statistical modeling with python. In *9th Python in Science Conference*.

[45] Y. Song, Y. Zhou, A. Sekhari, D. Bagnell, A. Krishnamurthy, and W. Sun. 2023. Hybrid RL: using both offline and online data can make RL efficient. *International Conference on Learning Representations*.
[46] R. S. Sutton. 1988. Learning to predict by the methods of temporal differences. *Machine learning*, 3, 9–44.
[47] R. S. Sutton, A. R. Mahmood, and M. White. 2016. An emphatic approach to the problem of off-policy temporal-difference learning. *Journal of Machine Learning Research*, 17, 73, 1–29.
[48] C. Szepesvari. 2010. *Algorithms for Reinforcement Learning*. Morgan & Claypool Publishers.
[49] J. N. Tsitsiklis and B. Van Roy. 1997. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42, 5.
[50] H. van Hasselt, J. Quan, M. Hessel, Z. Xu, D. Borsa, and A. Barreto. 2019. General non-linear bellman equations. *CoRR*, abs/1907.03687. arXiv: 1907.03687.
[51] H. van Seijen, M. Fatemi, and A. Tavakoli. 2019. Using a logarithmic mapping to enable lower discount factors in reinforcement learning. In *Advances in Neural Information Processing Systems*.
[52] S. Vito. 2016. Air Quality. UCI Machine Learning Repository. (2016).
[53] F. Wang, P. Li, and A. C. Konig. 2010. Learning a bi-stochastic data similarity matrix. In *2010 IEEE International Conference on Data Mining*. IEEE, 551–560.
[54] H. Xiao, K. Rasul, and R. Vollgraf. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. (2017). arXiv: 1708.07747 [cs.LG].
[55] H. Yao and Z.-Q. Liu. 2008. Preconditioned temporal difference learning. In *Proceedings of the 25th international conference on Machine learning*, 1208–1215.

The appendix is organized as follows.

(1) Appendix A provides detail proofs for the theorems.
(2) Appendix B provides implementation details on synthetic datasets from Section 4.1.
(3) Appendix C provides implementation details and additional results for those real-world datasets:
   - Linear regression (Appendix C.2),
   - Regression with Neural Networks (NNs) (Appendix C.3),
   - Binary classification with NNs (Appendix C.4),
   - Implementation details of different heuristically designed $\mathbf{P}$ are in Appendix C.5.

A Proofs

A.1 Proof for Proposition 4.1

Proposition 4.1. *[Connection between TD and OLS] When* $\mathbf{D}$ *has full support and* $\mathbf{X}$ *has linearly independent rows, TD and OLS have the same minimum norm solution. Moreover, any solution to the linear system* $\mathbf{Xw} = \mathbf{y}$ *must also be a solution to TD's linear system* $\mathbf{Aw} = \mathbf{b}$ *as defined in Equation* (4).

Proof. In our TD formulation, the reward is $\mathbf{r} = (\mathbf{I} - \gamma\mathbf{P})\mathbf{y}$. With $\mathbf{S} = \mathbf{D}(\mathbf{I} - \gamma\mathbf{P})$, we have $\mathbf{A} = \mathbf{X}^\top\mathbf{S}\mathbf{X}$, and $\mathbf{b} = \mathbf{X}^\top\mathbf{S}\mathbf{y}$. To verify the first claim, define The min-norm solutions found by TD and OLS are respectively $\mathbf{w}_{TD} = \mathbf{A}^\dagger\mathbf{b}$, and $\mathbf{w}_{LS} = \mathbf{X}^\dagger\mathbf{y}$, where

$$\mathbf{w}_{TD} = \mathbf{A}^\dagger\mathbf{b} = (\mathbf{X}^\top\mathbf{S}\mathbf{X})^\dagger \cdot \mathbf{X}^\top\mathbf{S}\mathbf{y}. \tag{35}$$

When $\mathbf{D}$ has full support, $\mathbf{S}$ is invertible (thus has linearly independent rows/columns). Additionally, $\mathbf{X}$ has linearly independent rows so $(\mathbf{X}^\top\mathbf{S}\mathbf{X})^\dagger = \mathbf{X}^\dagger\mathbf{S}^{-1}(\mathbf{X}^\top)^\dagger$ [15, Thm.3] and the TD solution becomes

$$\mathbf{w}_{TD} = \mathbf{A}^\dagger\mathbf{b} = \mathbf{X}^\dagger\mathbf{S}^{-1}(\mathbf{X}^\top)^\dagger\mathbf{X}^\top\mathbf{S}\mathbf{y} \tag{36}$$

Finally, when $\mathbf{X}$ has linearly independent rows, $(\mathbf{X}^\top)^\dagger\mathbf{X}^\top = \mathbf{I}_n$ so $\mathbf{w}_{TD} = \mathbf{X}^\dagger\mathbf{y} = \mathbf{w}_{LS}$.

To verify the second part, the TD's linear system is

$$\mathbf{X}^\top\mathbf{S}\mathbf{X}\mathbf{w} = \mathbf{X}^\top\mathbf{S}\mathbf{y}, \tag{37}$$

which is essentially preconditioning the linear system $\mathbf{Xw} = \mathbf{y}$ by $\mathbf{X}^\top\mathbf{S}$. Hence, any solution to the latter is also a solution to TD. □

A.2 Proof for Theorem 5.7

Lemma 5.3. *Under Assumption 5.1, 5.2, for* $\mathbf{w} \in \mathcal{W}$*, there exists* $L \geq 1$ *such that* $\forall s_1, s_2 \in \mathcal{S}$ *with* $z_1 = \mathbf{x}(s_1)^\top \mathbf{w}, z_2 = \mathbf{x}(s_2)^\top \mathbf{w}$,

$$\frac{1}{L}|z_1 - z_2| \leq |f(z_1) - f(z_2)| \leq L|z_1 - z_2|. \tag{24}$$

Proof. Assumption 5.1 and the compactness of $\mathcal{W}$ ensure that the linear prediction $z = \mathbf{x}(s)^\top \mathbf{w}, \forall s \in \mathcal{S}$ will also be in a compact domain. Given that f is continuous and invertible (Assumption 5.2), effectively the domain and image of will also be compact. Furthermore, since f' is bounded, there exists a constant $L = \max(L_f, L_{f^{-1}}) \geq 1$ such that Equation (24) holds $\forall s_1, s_2 \in \mathcal{S}$ with $z_1 = \mathbf{x}(s_1)^\top \mathbf{w}, z_2 = \mathbf{x}(s_2)^\top \mathbf{w}$, where $L_f, L_{f^{-1}}$ are the Lipschitz constants of f and f^{-1} respectively. □

We show the existence of an optimal solution $\mathbf{w}^* \in \mathcal{W}$ for our update using the Banach fixed-point theorem. In the following, we will show that any two parameters $\mathbf{w}^A, \mathbf{w}^B \in \mathcal{W}$ will get closer after the one update step and a projection step. To achieve this, we will first introduce two helpful lemmas in the following. To simplify notation, denote their predictions $z^A = \mathbf{x}^\top \mathbf{w}^A, z^B = \mathbf{x}^\top \mathbf{w}^B$, transformed predictions $y^A = f(z^A), y^B = f(z^B)$ and bootstrap targets $y^A_{td} = f(r + \mathbf{x}'^\top \mathbf{w}^A), y^B_{td} = f(r + \mathbf{x}'^\top \mathbf{w}^B)$.

Lemma 5.5. *For* $\mathbf{w}^A, \mathbf{w}^B \in \mathcal{W}$, $(\mathbf{w}^A - \mathbf{w}^B)^\top (\overline{g}(\mathbf{w}^B) - \overline{g}(\mathbf{w}^A)) \geq \left(\frac{1}{L} - \gamma L\right) \cdot \mathbb{E}\left[(z^A - z^B)^2\right]$.

Proof. Note that

$$(\mathbf{w}^A - \mathbf{w}^B)^\top [\overline{g}(\mathbf{w}^B) - \overline{g}(\mathbf{w}^A)] = (\mathbf{w}^A - \mathbf{w}^B)^\top \mathbb{E}[\ [(y^B_{td} - y^B) - (y^A_{td} - y^A)] \cdot \mathbf{x}\ ] \tag{38}$$

$$= \mathbb{E}[\ (z^A - z^B) \cdot [(y^A - y^B) - (y^A_{td} - y^B_{td})]\ ] \tag{39}$$

By Lemma 5.3 and using the assumption that f is strictly increasing, we have $(z^A - z^B)(y^A - y^B) \geq \frac{1}{L}(z^A - z^B)^2$. Moreover, the function $\mathbf{z} \mapsto f(\mathbf{r} + \gamma \mathbf{P}\mathbf{z})$ is (γL)-Lipschitz so

$$\mathbb{E}\left[\ (z^A - z^B)(y^A_{td} - y^B_{td})\ \right] \leq \gamma L \cdot \mathbb{E}[(z^A - z^B)^2]. \tag{40}$$

Plugging these two into Equation (39) completes the proof. □

Lemma 5.6. *For* $\mathbf{w}^A, \mathbf{w}^B \in \mathcal{W}$, $\|\overline{g}(\mathbf{w}^A) - \overline{g}(\mathbf{w}^B)\|_2 \leq 2L\sqrt{\mathbb{E}\left[(z^A - z^B)^2\right]}$

Proof. To start

$$\|\overline{g}(\mathbf{w}^A) - \overline{g}(\mathbf{w}^B)\|_2 = \|\mathbb{E}[\ [(y^A_{td} - y^A) - (y^B_{td} - y^B)] \cdot \mathbf{x}\ ]\|_2 \tag{41}$$

$$\leq \sqrt{\mathbb{E}[\|\mathbf{x}\|_2^2]}\sqrt{\mathbb{E}\left[[(y^A_{td} - y^A) - (y^B_{td} - y^B)]^2\right]} \qquad \text{Cauchy-Schwartz} \tag{42}$$

$$\leq \sqrt{\mathbb{E}\left[[(y^B - y^A) - (y^B_{td} - y^A_{td})]^2\right]} \qquad \text{Assumption 5.1} \tag{43}$$

Let $\delta \overset{\text{def}}{=} y^B - y^A$ and $\delta_{td} \overset{\text{def}}{=} y^B_{td} - y^A_{td}$, then

$$\mathbb{E}\left[(\delta - \delta_{td})^2\right] = \mathbb{E}[\delta^2] + \mathbb{E}[\delta^2_{td}] - 2\mathbb{E}[\delta\delta_{td}] \tag{44}$$

$$\leq \mathbb{E}[\delta^2] + \mathbb{E}[\delta^2_{td}] + 2|\mathbb{E}[\delta\delta_{td}]| \tag{45}$$

$$\leq \mathbb{E}[\delta^2] + \mathbb{E}[\delta^2_{td}] + 2\sqrt{\mathbb{E}[\delta^2]\mathbb{E}[\delta^2_{td}]} \qquad \text{Cauchy-Schwartz} \tag{46}$$

$$= \left(\sqrt{\mathbb{E}[\delta^2]} + \sqrt{\mathbb{E}[\delta^2_{td}]}\right)^2 \tag{47}$$

As a result,

$$\|\overline{g}(\mathbf{w}^A) - \overline{g}(\mathbf{w}^B)\|_2 \leq \sqrt{\mathbb{E}\left[(y^B - y^A)^2\right]} + \sqrt{\mathbb{E}\left[(y^B_{td} - y^A_{td})^2\right]} \tag{48}$$

$$\leq L\left\{\sqrt{\mathbb{E}\left[(z^B - z^A)^2\right]} + \sqrt{\mathbb{E}\left[(z^B_{td} - z^A_{td})^2\right]}\right\} \tag{49}$$

Finally, note that

$$\mathbb{E}\left[(z^B_{td} - z^A_{td})^2\right] = \mathbb{E}\left[\ [(r + \gamma \mathbf{x}'^\top \mathbf{w}^B) - (r + \gamma \mathbf{x}'^\top \mathbf{w}^A)]^2\ \right] \tag{50}$$

$$= \gamma^2 \mathbb{E}\left[\ (\mathbf{x}'^\top \mathbf{w}^B - \mathbf{x}'^\top \mathbf{w}^A)^2\ \right] \tag{51}$$

$$= \gamma^2 \mathbb{E}\left[\ (z^B - z^A)^2\ \right] \tag{52}$$

where the last line is because both s, s' are assumed to be from the stationary distribution. Plugging this to Equation (49) and using the fact that $\gamma \leq 1$ complete the proof. □

Now we are ready to prove the contraction. Given two iterates $\mathbf{w}^A_t, \mathbf{w}^B_t \in \mathcal{W}$ and their updates

$$\mathbf{w}^A_{t+1} = \mathcal{P}(\mathbf{w}^A_t + \alpha \overline{g}(\mathbf{w}^A_t)) \qquad \mathbf{w}^B_{t+1} = \mathcal{P}(\mathbf{w}^B_t + \alpha \overline{g}(\mathbf{w}^B_t)), \tag{53}$$

the following theorem shows that the projected update is a contraction mapping.

Theorem 5.7. *[Contraction of Expected Update] Under Assumption 5.1-5.4, by choosing $\alpha = \frac{1-\gamma L^2}{4L^3} > 0$, the projected update (25) is a contraction w.r.t. $\|\cdot\|_2$ with a unique fixed point $\mathbf{w}^* \in \mathcal{W}$, and $\mathbf{w}_t$ converges to it.*

Proof. For $\mathbf{w}^A_t, \mathbf{w}^B_t \in \mathcal{W}$, with probability one, for any $t \in \mathbb{N}_0$

$$\|\mathbf{w}^A_{t+1} - \mathbf{w}^B_{t+1}\|_2^2 \leq \|\mathbf{w}^A_t + \alpha \overline{g}(\mathbf{w}^A_t) - (\mathbf{w}^B_t + \alpha \overline{g}(\mathbf{w}^B_t))\|_2^2 \tag{54}$$

since projection onto a convex set is contracting. Then we can further expand it as

$$\|\mathbf{w}^A_{t+1} - \mathbf{w}^B_{t+1}\|_2^2 \leq \|\mathbf{w}^A_t - \mathbf{w}^B_t\|_2^2 - 2\alpha(\mathbf{w}^A_t - \mathbf{w}^B_t)^\top(\overline{g}(\mathbf{w}^B_t) - \overline{g}(\mathbf{w}^A_t)) + \alpha^2 \|\overline{g}(\mathbf{w}^A_t) - \overline{g}(\mathbf{w}^B_t)\|_2^2 \tag{55}$$

where we can bound the second and third terms using Lemma 5.5 and Lemma 5.6 respectively so

$$\|\mathbf{w}^A_{t+1} - \mathbf{w}^B_{t+1}\|_2^2 \leq \|\mathbf{w}^A_t - \mathbf{w}^B_t\|_2^2 - \left(2\alpha\left(\frac{1}{L} - \gamma L\right) - 4L^2\alpha^2\right)\mathbb{E}\left[(z^A_t - z^B_t)^2\right] \tag{56}$$

By choosing $\alpha = \frac{1-\gamma L^2}{4L^3} > 0$

$$\|\mathbf{w}^A_{t+1} - \mathbf{w}^B_{t+1}\|_2^2 \leq \|\mathbf{w}^A_t - \mathbf{w}^B_t\|_2^2 - \left(\frac{1-\gamma L^2}{2L^2}\right)^2 \mathbb{E}\left[(z^A_t - z^B_t)^2\right] \tag{57}$$

$$= \|\mathbf{w}^A_t - \mathbf{w}^B_t\|_2^2 - \left(\frac{1-\gamma L^2}{2L^2}\right)^2 \mathbb{E}\left[(\mathbf{x}^\top(\mathbf{w}^A_t - \mathbf{w}^B_t))^2\right] \tag{58}$$

$$\leq \|\mathbf{w}^A_t - \mathbf{w}^B_t\|_2^2 - \left(\frac{1-\gamma L^2}{2L^2}\right)^2 \mathbb{E}\left[\|\mathbf{x}\|_2^2\right] \|\mathbf{w}^A_t - \mathbf{w}^B_t\|_2^2 \qquad \text{Cauchy-Schwartz} \tag{59}$$

$$\leq \|\mathbf{w}^A_t - \mathbf{w}^B_t\|_2^2 - \left(\frac{1-\gamma L^2}{2L^2}\right)^2 \|\mathbf{w}^A_t - \mathbf{w}^B_t\|_2^2 \qquad \text{Assumption 5.1} \tag{60}$$

Finally, due to Assumption 5.4 and Lemma 5.3, we have $1 - \left(\frac{1-\gamma L^2}{2L^2}\right)^2 \in [0, 1)$ so it is a contraction mapping w.r.t. $\|\cdot\|_2$. By the Banach fixed-point theorem, the projected update admits a unique fixed point $\mathbf{w}^* \in \mathcal{W}$ and our iterate converges to it. □

A.3 Proof for Theorem 5.8

Once we know that the algorithm is convergent from Theorem 5.7, we are now ready to prove the convergence rate.

Theorem 5.8. *[Convergence Rate of Expected Update] Under Assumption 5.1-5.4, consider the sequence* $(\mathbf{w}_0, \mathbf{w}_1, \cdots)$ *satisfying Equation (25). Let* $\overline{\mathbf{w}}_T \overset{def}{=} \frac{1}{T}\sum_{t=0}^{T-1} \mathbf{w}_t$, $\overline{z}_T = \mathbf{x}^\top \overline{\mathbf{w}}_T$ *and* $z^* = \mathbf{x}^\top \mathbf{w}^*$. *By choosing* $\alpha = \frac{1-\gamma L^2}{4L^3} > 0$, *we have*

$$\mathbb{E}\left[(z^* - \overline{z}_T)^2\right] \leq \left(\frac{2L^2}{1-\gamma L^2}\right)^2 \frac{\|\mathbf{w}^* - \mathbf{w}_0\|_2^2}{T} \tag{30}$$

$$\|\mathbf{w}^* - \mathbf{w}_T\|_2^2 \leq \exp\left(-T\omega\left(\frac{1-\gamma L^2}{2L^2}\right)^2\right)\|\mathbf{w}^* - \mathbf{w}_0\|_2^2 \tag{31}$$

Proof. With probability 1, for any $t \in \mathbb{N}_0$

$$\|\mathbf{w}^* - \mathbf{w}_{t+1}\|_2^2 = \|\mathbf{w}^* - \mathbf{w}_t\|_2^2 - 2\alpha(\mathbf{w}^* - \mathbf{w}_t)^\top(\overline{g}(\mathbf{w}_t) - \overline{g}(\mathbf{w}^*)) + \alpha^2\|\overline{g}(\mathbf{w}_t) - \overline{g}(\mathbf{w}^*)\|_2^2 \tag{61}$$

$$\leq \|\mathbf{w}^* - \mathbf{w}_t\|_2^2 - \left(2\alpha\left(\frac{1}{L} - \gamma L\right) - 4L^2\alpha^2\right)\mathbb{E}\left[(z^* - z_t)^2\right] \tag{62}$$

where $z_t \overset{\text{def}}{=} \mathbf{x}^\top \mathbf{w}_t$. The last inequality uses Lemma 5.5 and Lemma 5.6 with $\mathbf{w}^A = \mathbf{w}_t$ and $\mathbf{w}_B = \mathbf{w}^*$. Using $\alpha = \frac{1-\gamma L^2}{4L^3} > 0$

$$\|\mathbf{w}^* - \mathbf{w}_{t+1}\|_2^2 \leq \|\mathbf{w}^* - \mathbf{w}_t\|_2^2 - \left(\frac{1-\gamma L^2}{2L^2}\right)^2 \mathbb{E}\left[(z^* - z_t)^2\right] \tag{63}$$

Telescoping sum gives

$$\left(\frac{1-\gamma L^2}{2L^2}\right)^2 \times \sum_{t=0}^{T-1} \mathbb{E}\left[(z^* - z_t)^2\right] \leq \sum_{t=0}^{T-1}(\|\mathbf{w}^* - \mathbf{w}_t\|_2^2 - \|\mathbf{w}^* - \mathbf{w}_{t+1}\|_2^2) \leq \|\mathbf{w}^* - \mathbf{w}_0\|_2^2 \tag{64}$$

By Jensen's inequality

$$\mathbb{E}\left[(z^* - \overline{z}_T)^2\right] \leq \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\left[(z^* - z_t)^2\right] \leq \left(\frac{2L^2}{1-\gamma L^2}\right)^2 \frac{\|\mathbf{w}^* - \mathbf{w}_0\|_2^2}{T} \tag{65}$$

Finally, since we assume that $\|\mathbf{x}(s)\|_2^2 \leq 1, \forall s$, we have $\mathbb{E}\left[(z^* - z_t)^2\right] \geq \omega\|\mathbf{w}^* - \mathbf{w}_t\|_2^2$ where ω is the maximum eigenvalue of the steady-state feature covariance matrix $\Sigma = \mathbf{X}^\top \mathbf{D}\mathbf{X} = \sum_s D(s)\mathbf{x}(s)\mathbf{x}(s)^\top$. Therefore, Equation (63) leads to

$$\|\mathbf{w}^* - \mathbf{w}_{t+1}\|_2^2 \leq \left(1 - \omega\left(\frac{1-\gamma L^2}{2L^2}\right)^2\right)\|\mathbf{w}^* - \mathbf{w}_t\|_2^2 \tag{66}$$

$$\leq \exp\left(-\omega\left(\frac{1-\gamma L^2}{2L^2}\right)^2\right)\|\mathbf{w}^* - \mathbf{w}_t\|_2^2 \qquad \forall x \in \mathbb{R}, 1 - x \leq e^{-x} \tag{67}$$

Repeatedly applying this bound gives Equation (31). □

A.4 Proof of Theorem 5.11

Here, we analyze the sample-based update (32). To account for randomness, let $\sigma^2 \stackrel{\text{def}}{=} \mathbb{E}[\|g_t(\mathbf{w}^*)\|_2^2]$, the variance of the TD update at the stationary point $\mathbf{w}^*$ under the stationary distribution.

Lemma 5.9. *The fixed point $\mathbf{w}^* \in \mathcal{W}$ in Theorem 5.7 satisfies*

$$(\mathbf{w}^* - \mathbf{w})^\top \overline{g}(\mathbf{w}^*) \geq 0 \quad \forall \mathbf{w} \in \mathcal{W}. \tag{33}$$

Proof. Since $\mathbf{w}^*$ is the fixed point of the iterative update (25), it satisfies

$$\mathbf{w}^* = \operatorname*{argmin}_{\mathbf{w} \in \mathcal{W}} \|\mathbf{w} - (\mathbf{w}^* + \alpha \overline{g}(\mathbf{w}^*))\|_2^2. \tag{68}$$

This is a constrained optimization with a well-defined objective (L_2 distance), which means we can apply the first-order optimality condition [8, Sec.4.2.3] and $\mathbf{w}^*$ satisfies

$$[\mathbf{w}^* - (\mathbf{w}^* + \alpha \overline{g}(\mathbf{w}^*))]^\top [\mathbf{w} - \mathbf{w}^*] \geq 0 \quad \forall \mathbf{w} \in \mathcal{W}. \tag{69}$$

Intuitively, this condition means that $\mathbf{w}^* - (\mathbf{w}^* + \alpha \overline{g}(\mathbf{w}^*))$ makes a non-obtuse angle with any feasible direction $\mathbf{w} - \mathbf{w}^*$. Simplifying this equation gives the desired result (33). □

Lemma 5.10. *For $\mathbf{w} \in \mathcal{W}$, $\mathbb{E}[\|g_t(\mathbf{w})\|_2^2] \leq 2\sigma^2 + 8L^2\mathbb{E}[(z_t^* - z_t)^2]$ where $\sigma^2 = \mathbb{E}[\|g_t(\mathbf{w}^*)\|_2^2]$.*

Proof. To start

$$\begin{aligned}
\mathbb{E}[\|g_t(\mathbf{w})\|_2^2] &= \mathbb{E}[\|g_t(\mathbf{w}) - g_t(\mathbf{w}^*) + g_t(\mathbf{w}^*)\|_2^2] && && (70)\\
&\leq \mathbb{E}[(\|g_t(\mathbf{w}^*)\|_2 + \|g_t(\mathbf{w}) - g_t(\mathbf{w}^*)\|_2)^2] && \text{Triangle inequality} && (71)\\
&\leq 2\mathbb{E}[(\|g_t(\mathbf{w}^*)\|_2^2] + 2\mathbb{E}[\|g_t(\mathbf{w}) - g_t(\mathbf{w}^*)\|_2^2] && (a+b)^2 \leq 2a^2 + 2b^2 && (72)\\
&\leq 2\sigma^2 + 2\mathbb{E}\left[\ \|[(y_{t,td} - y_t) - (y_{t,td}^* - y_t^*)]\mathbf{x}_t\|_2^2\ \right] && && (73)\\
&\leq 2\sigma^2 + 2\mathbb{E}\left[\ ((y_{t,td} - y_t) - (y_{t,td}^* - y_t^*))^2\ \right] && \text{Assumption 5.1} && (74)\\
&= 2\sigma^2 + 2\mathbb{E}\left[\ ((y_t^* - y_t) - (y_{t,td}^* - y_{t,td}))^2\ \right] && && (75)\\
&\leq 2\sigma^2 + 4\left(\mathbb{E}[(y_t^* - y_t)^2] + \mathbb{E}[(y_{t,td}^* - y_{t,td})^2]\right) && (a-b)^2 \leq 2a^2 + 2b^2 && (76)
\end{aligned}$$

where $y_t^* \stackrel{\text{def}}{=} f(z_t^*) = f(\mathbf{x}_t^\top \mathbf{w}^*)$ and $y_{t,td}^* \stackrel{\text{def}}{=} f(z_{t,td}^*) = f(r_t + \mathbf{x}_{t+1}^\top \mathbf{w}^*)$. Note that by Lemma 5.3

$$\mathbb{E}[(y_t^* - y_t)^2] + \mathbb{E}[(y_{t,td}^* - y_{t,td})^2] \leq L^2 \left\{\mathbb{E}\left[(z_t^* - z_t)^2\right] + \mathbb{E}\left[(z_{t,td}^* - z_{t,td})^2\right]\right\}. \tag{77}$$

Finally,

$$\begin{aligned}
\mathbb{E}\left[(z_{t,td}^* - z_{t,td})^2\right] &= \mathbb{E}\left[\ [(r_t + \gamma \mathbf{x}_{t+1}^\top \mathbf{w}^*) - (r_t + \gamma \mathbf{x}_{t+1}^\top \mathbf{w}_t)]^2\ \right] && (78)\\
&= \gamma^2 \mathbb{E}\left[\ (\mathbf{x}_{t+1}^\top \mathbf{w}^* - \mathbf{x}_{t+1}^\top \mathbf{w}_t)^2\ \right] && (79)\\
&= \gamma^2 \mathbb{E}\left[\ (z_t^* - z_t)^2\ \right] && (80)
\end{aligned}$$

where the last line is because both s_t, s_{t+1} are from the stationary distribution. Combining these with Equation (76) gives

$$\begin{aligned}
\mathbb{E}[\|g_t(\mathbf{w})\|_2^2] &\leq 2\sigma^2 + 4L^2(1+\gamma^2)\mathbb{E}\left[(z_t^* - z_t)^2\right] && && (81)\\
&\leq 2\sigma^2 + 8L^2\mathbb{E}\left[(z_t^* - z_t)^2\right]. && 0 \leq \gamma \leq 1 && (82)
\end{aligned}$$

□

Now we are ready to present the convergence when using i.i.d. sample for the update:

Theorem 5.11. *[Convergence Rate of Sampled-based Update] Under Assumption 5.1-5.4, with sample-based update Equation (32), let* $\sigma^2 = \mathbb{E}[\|g_t(\mathbf{w}^*)\|_2^2]$, $\overline{\mathbf{w}}_T \overset{def}{=} \frac{1}{T}\sum_{t=0}^{T-1}\mathbf{w}_t$, $\overline{z}_T = \mathbf{x}^\top \overline{\mathbf{w}}_T$ *and* $z^* = \mathbf{x}^\top \mathbf{w}^*$. *For* $T \geq \frac{64L^6}{(1-\gamma L^2)^2}$ *and a constant step size* $\alpha_t = 1/\sqrt{T}, \forall t$, *we have*

$$\mathbb{E}\left[(z^* - \overline{z}_T)^2\right] \leq \frac{L\left(\|\mathbf{w}^* - \mathbf{w}_0\|_2^2 + 2\sigma^2\right)}{\sqrt{T}(1-\gamma L^2)}. \tag{34}$$

Proof. First note that for any $t \in \mathbb{N}_0$

$$\|\mathbf{w}^* - \mathbf{w}_{t+1}\|_2^2 = \|\mathcal{P}(\mathbf{w}^*) - \mathcal{P}(\mathbf{w}_t + \alpha_t g_t(\mathbf{w}_t))\|_2^2 \tag{83}$$
$$\leq \|\mathbf{w}^* - (\mathbf{w}_t + \alpha_t g_t(\mathbf{w}_t))\|_2^2 \tag{84}$$

since $\mathbf{w}^* \in \mathcal{W}$ and projection onto a convex set is contracting. Therefore,

$$\mathbb{E}[\|\mathbf{w}^* - \mathbf{w}_{t+1}\|_2^2] \leq \mathbb{E}[\|\mathbf{w}^* - \mathbf{w}_t - \alpha_t g_t(\mathbf{w}_t))\|_2^2] \tag{85}$$
$$= \mathbb{E}[\|\mathbf{w}^* - \mathbf{w}_t\|_2^2] - 2\alpha_t \mathbb{E}[(\mathbf{w}^* - \mathbf{w}_t)^\top g_t(\mathbf{w}_t)] + \alpha_t^2 \mathbb{E}[\|g_t(\mathbf{w}_t)\|_2^2] \tag{86}$$

Let's take a look at the second term more closely.

$$\mathbb{E}[(\mathbf{w}^* - \mathbf{w}_t)^\top g_t(\mathbf{w}_t)] = \mathbb{E}[(\mathbf{w}^* - \mathbf{w}_t)^\top (g_t(\mathbf{w}_t) - g_t(\mathbf{w}^*))] + \mathbb{E}[(\mathbf{w}^* - \mathbf{w}_t)^\top g_t(\mathbf{w}^*)] \tag{87}$$

and for the last term here we have

$$\mathbb{E}[(\mathbf{w}^* - \mathbf{w}_t)^\top g_t(\mathbf{w}^*)] = \mathbb{E}[\mathbb{E}[(\mathbf{w}^* - \mathbf{w}_t)^\top g_t(\mathbf{w}^*)|\mathbf{w}_t]] \tag{88}$$
$$= \mathbb{E}[(\mathbf{w}^* - \mathbf{w}_t)^\top \overline{g}(\mathbf{w}^*)] \geq 0 \tag{89}$$

which is due to Lemma 5.9. As a result, Equation (86) becomes

$$\mathbb{E}[\|\mathbf{w}^* - \mathbf{w}_{t+1}\|_2^2] \leq \mathbb{E}[\|\mathbf{w}^* - \mathbf{w}_t\|_2^2] - 2\alpha_t \mathbb{E}[(\mathbf{w}^* - \mathbf{w}_t)^\top (g_t(\mathbf{w}_t) - g_t(\mathbf{w}^*))] + \alpha_t^2 \mathbb{E}[\|g_t(\mathbf{w}_t)\|_2^2] \tag{90}$$

Note that Lemma 5.5 holds for any $\mathbf{w}^A, \mathbf{w}^B \in \mathcal{W}$ and the expectation in $\overline{g}(\mathbf{w}) = \mathbb{E}[g_t(\mathbf{w})]$ is based on the sample (s_t, r_t, s_{t+1}), *regardless of the choice of* $\mathbf{w}$. Thus, one can choose *any* $\mathbf{w}$ and then $\mathbb{E}[g_t(\mathbf{w})|\mathbf{w}] = \overline{g}(\mathbf{w})$. As a result, both Lemma 5.5 and Lemma 5.10 can be applied to $\mathbb{E}[(\mathbf{w}^* - \mathbf{w}_t)^\top (g_t(\mathbf{w}_t) - g_t(\mathbf{w}^*))|\mathbf{w}_t]$ and $\mathbb{E}[\|g_t(\mathbf{w}_t)\|_2^2|\mathbf{w}_t]$, respectively, in the following. Thus Equation (90) becomes

$$\mathbb{E}[\|\mathbf{w}^* - \mathbf{w}_{t+1}\|_2^2] \leq \mathbb{E}[\|\mathbf{w}^* - \mathbf{w}_t\|_2^2] - 2\alpha_t \mathbb{E}[\mathbb{E}[(\mathbf{w}^* - \mathbf{w}_t)^\top (g_t(\mathbf{w}_t) - g_t(\mathbf{w}^*))|\mathbf{w}_t]] + \alpha_t^2 \mathbb{E}[\mathbb{E}[\|g_t(\mathbf{w}_t)\|_2^2|\mathbf{w}_t]] \tag{91}$$
$$\leq \mathbb{E}[\|\mathbf{w}^* - \mathbf{w}_t\|_2^2] - \left(2\alpha_t\left(\frac{1}{L} - \gamma L\right) - 8L^2\alpha_t^2\right)\mathbb{E}\left[(z^* - z_t)^2\right] + 2\alpha_t^2\sigma^2 \tag{92}$$
$$\leq \mathbb{E}[\|\mathbf{w}^* - \mathbf{w}_t\|_2^2] - \alpha_t\left(\frac{1}{L} - \gamma L\right)\mathbb{E}\left[(z^* - z_t)^2\right] + 2\alpha_t^2\sigma^2 \tag{93}$$

where the last inequality is due to $\alpha_t = \frac{1}{\sqrt{T}} \leq \frac{1-\gamma L^2}{8L^3}$. Then telescoping sum gives

$$\frac{1}{\sqrt{T}}\left(\frac{1}{L} - \gamma L\right)\sum_{t=0}^{T-1}\mathbb{E}\left[(z^* - z_t)^2\right] \leq \|\mathbf{w}^* - \mathbf{w}_0\|_2^2 + 2\sigma^2 \tag{94}$$

$$\iff \sum_{t=0}^{T-1}\mathbb{E}\left[(z^* - z_t)^2\right] \leq \frac{\sqrt{T}L}{1-\gamma L^2}\left(\|\mathbf{w}^* - \mathbf{w}_0\|_2^2 + 2\sigma^2\right). \tag{95}$$

Finally, Jensen's inequality completes the proof

$$\mathbb{E}\left[(z^* - \bar{z}_T)^2\right] \leq \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[(z^* - z_t)^2\right] \leq \frac{L\left(\|\mathbf{w}^* - \mathbf{w}_0\|_2^2 + 2\sigma^2\right)}{\sqrt{T}(1-\gamma L^2)}. \tag{96}$$

□

B Experiment Details: Synthetic Data

This subsection provides details of the empirical verification in Section 4.1.

Each element of the input matrix $\mathbf{X}$ is drawn from the standard normal distribution $\mathcal{N}(0,1)$. This (almost surely) guarantees that $\mathbf{X}$ has linearly independent rows in the overparametrization regime (i.e., when $n < d$). The true model $\mathbf{w}^*$ is set to be a vector of all ones. Each label y_i is generated by $y_i = \mathbf{x}_i^\top \mathbf{w}^* + \epsilon_i$ with noise $\epsilon_i \sim \mathcal{N}(0, 0.1^2)$.

We test various transition matrices $\mathbf{P}$ as shown in Table 2. For `Random`, each element of $\mathbf{P}$ is drawn from the uniform distribution $U(0,1)$ and then normalized so that each row sums to one. The `Deficient` variant is exact the same as `Random`, except that the last column is set to all zeros before row normalization. This ensures that the last state is never visited from any state, thus not having full support in its stationary distribution. `Uniform` simply means every element of $\mathbf{P}$ is set to $1/n$ where $n = 100$ is the number of training points. `Distance (Close)` assigns higher transition probability to points closer to the current point, where the element in the ith row and the jth column is first set to $\exp(-(y_i - y_j)^2/2)$ then the whole matrix is row-normalized. Finally, `Distance (Far)` uses $1 - \exp(-(y_i - y_j)^2/2)$ before normalization. The last two variants are used to see if similarity between points can play a role in the transition when using our TD algorithm.

As shown in Table 2, the min-norm solution $\mathbf{w}_{TD}$ is very close to the min-norm solution of OLS as long as $n < d$ and $\mathbf{D}$ has full support (non-deficient $\mathbf{P}$). The choice of $\mathbf{P}$ only has little effect in such cases. This synthetic experiment verifies our analysis in the main text.

C Experiment Details: Real-world Data

C.1 Implementation Details

Deep learning experiments are based on tensorflow [30], version 2.11.0, except that the ResNN18 experiments are using pytorch [41]. Code is available at https://github.com/yannickycpan/reproduceSL.git. Below introduce common setup; different settings will be specified when mentioned.

Datasets. We use three popular datasets house price [26], execution time [40] and Bikeshare [13] as benchmark datasets. We have performed one-hot encoding for all categorical variables and removed irrelevant features such as date and year as done by [37]. This preprocessing results in 114 features. The Bikeshare dataset, which uses count numbers as its target variable, is popularly used for testing Poisson regressions. The air quality dataset [52] is loaded by using package *ucimlrepo* by `from ucimlrepo import fetch_ucirepo`.

For image datasets, we employ CNN consisting of three convolution layers with the number of kernels 32, 64, and 64, each with a filter size of 2×2. This was followed by two fully connected hidden layers with 256 and 128 units, respectively, before the final output layer. On all image datasets, Adam optimizer is used and the learning

rate sweeps over $\{0.003, 0.001, 0.0003\}$, $\gamma \in \{0.01, 0.1, 0.2\}$, $\tau \in \{0.01, 0.1\}$. The neural network is trained with mini-batch size 128. For ResNN18, we use learning rate 0.001 and $\tau = 0.01$.

Hyperparameter settings. For regression and binary classification tasks, we employ neural networks with two hidden layers of size 256×256 and ReLU activation functions. These networks are trained with a mini-batch size of 128 using the Adam optimizer [21]. In our TD algorithm, we perform hyperparameter sweeps for $\gamma \in \{0.1, 0.9\}$, target network moving rate $\tau \in \{0.01, 0.1\}$. For all algorithms we sweep learning rate $\alpha \in \{0.0003, 0.001, 0.003, 0.01\}$, except for cases where divergence occurs, such as in Poisson regression on the Bikeshare dataset, where we additionally sweep $\{0.00003, 0.0001\}$. Training iterations are set to 15k for the house data, 25k for Bikeshare, and 30k for other datasets. We perform random splits of each dataset into 60% for training and 40% for testing for each random seed or independent run. Hyperparameters are optimized over the final 25% of evaluations to avoid divergent settings. The reported results in the tables represent the average of the final two evaluations after smoothing window.

Naming rules. For convenience, we repeat naming rules from the main body here. TDReg: our TD approach, with its direct competitor being Reg (conventional l_2 regression). Reg-WP: Utilizes the same probability transition matrix as TDReg but does not employ bootstrap targets. This baseline can be used to assess the effect of bootstrap and transition probability matrix. On Bikesharedata, TDReg uses an exponential link function designed for handling counting data, and the baseline becomes Poisson regression correspondingly.

C.2 Additional Results on Linear Regression

As complementary results to the execute time dataset presented in Section 6, we provide results on two other regression datasets below (see Figure 6). We consistently observe that the TD algorithm performs more closely to the underlying best estimator. Notably, the performance gain tends to increase as the correlation strengthens or as the transition probability matrix better aligns with the data correlation.

When implementing FGLS, we initially run OLS to obtain the residuals. Subsequently, an algorithm is employed to fit these residuals for estimating the noise covariance matrix. This matrix is then utilized to compute the closed-form solution. The implementation is done by API from Seabold and Perktold [44].

In this set of experiments, to speed up multiple matrix inversion and noise sampling, we randomly take 500 subset of the original datasets for training and testing.

C.3 Additional Results: Regression with Correlated Noise with NNs

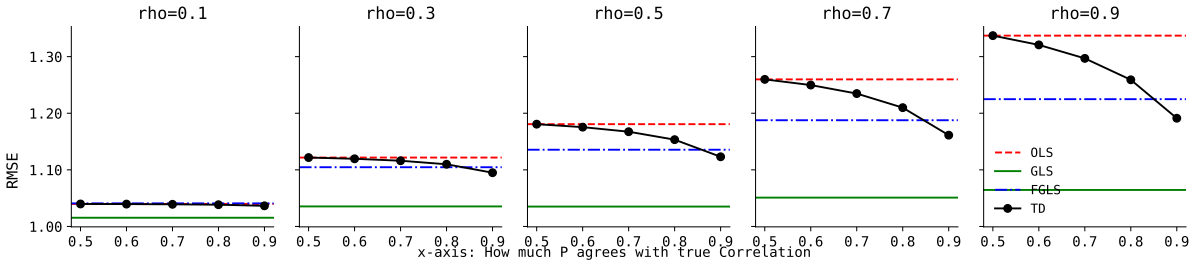
Similar to that has been shown in Figure 3, we show learning curves on increasingly strong correlated noise in Figure 7. In this set of experiments, to speed up noise sampling, we randomly take 1k subset of the original datasets for training and testing. The covariance matrix used to generate correlated noise is specified in Appendix B.

C.4 Additional Results: Binary Classification with NNs

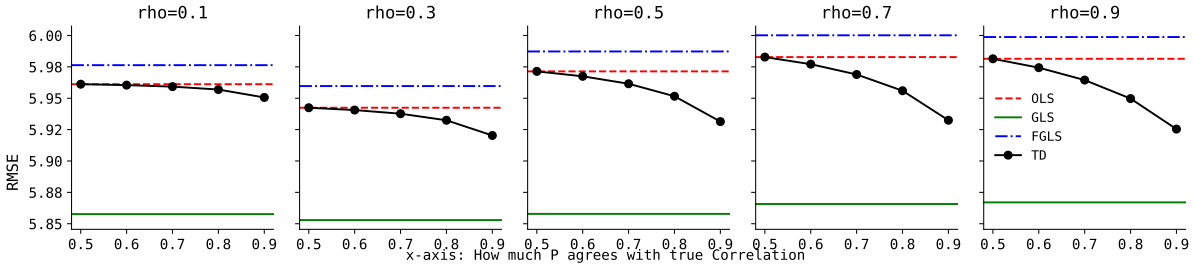
For binary classification, we utilize datasets from Australian weather [19], and travel insurance [11]. The aim of this series of experiments is to examine: 1) the impact of utilizing a link function; 2) with a specially designed transition probability matrix, the potential unique benefits of the TD algorithm in an imbalanced classification setting. Our findings indicate that: 1) the link function significantly influences performance; and 2) while the transition matrix proves beneficial in addressing class imbalance, this advantage seems to stem primarily from up/down-sampling rather than from the TD bootstrap targets.

Recall that we employ three intuitive types of transition matrices: $P(\mathbf{x}'|\mathbf{x})$ is larger when the two points $\mathbf{x}, \mathbf{x}'$ are 1) similar (denoted as P_s); 2) far apart (P_f); 3) $P(\mathbf{x}'|\mathbf{x}) = 1/n, \forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}$ (P_c).

Since our results in Section 6.4 and Figure 5 indicate that $\mathbf{P}$ does not significantly impact regular regression, we conducted experiments on binary classification tasks and observed their particular utility when dealing with

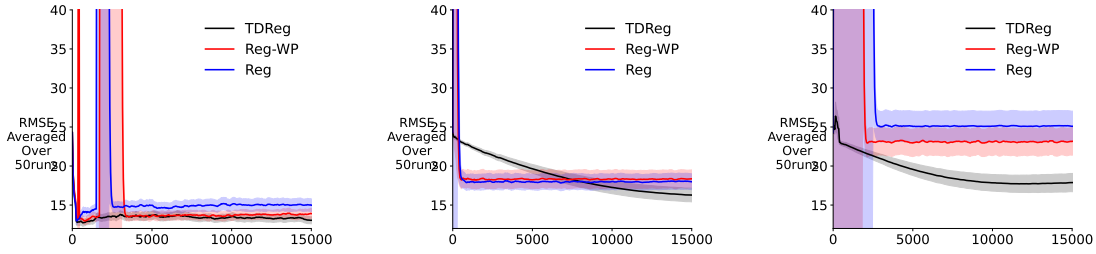


(a) bikeshare dataset



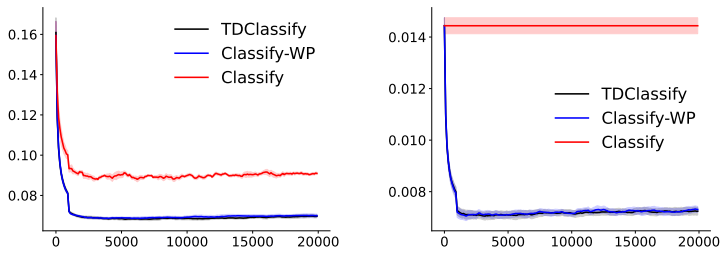
(b) house price dataset

Fig. 6. Test Root Mean Squared Error (RMSE) versus different values of P on bikeshare and execution time dataset. From left to right, the noise correlation coefficient increases ($\rho \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$). Each plot's x-axis represents the degree of alignment between the transition matrix P and the true covariance matrix that generates the noise. Consistent with our expectations, as P approaches the true correlation – implying a higher likelihood of data points with positively correlated noise transitioning from one to another – the solution derived from TD increasingly approximates the optimal one. Furthermore, as the correlation among the data intensifies, TD's solution is closer to optimum and one can see larger gap between TD and OLS/FGLS. The results are averaged over 30 runs.



(a) House, $\sigma = 10$ (b) House, $\sigma = 20$ (c) House, $\sigma = 30$

Fig. 7. Learning curves on house dataset. From left to right, the noise variance increases, with $\sigma \in 10, 20, 30$. The results are averaged over 50 runs. Due to the addition of large-scale noise, the two baseline algorithms, Reg-WP and Reg, exhibit high instability during the early learning stages and ultimately converge to a suboptimal solution compared to our TD approach.



(a) weather (b) insurance

Fig. 8. Learning curves of binary classification with imbalance: balanced testing error v.s. training steps. The results have been smoothed using a 5-point window before averaging over 5 runs.

imbalanced labels. We define P_s by defining the probability of transitioning to the same class as 0.9 and to the other class as 0.1. Table 5 presents the reweighted balanced results for three binary classification datasets with class imbalance. It is worth noting that in such cases, Classify-WP serves as both 1) no bootstrap baseline and 2) the upsampling techniques for addressing class imbalance in the literature [23].

Observing that TD-Classify and Classify-WP yield nearly identical results and Classify (without using TD's sampling) is significantly worse, suggesting that the benefit of TD arises from the sampling distribution rather than the bootstrap estimate in the imbalanced case. Furthermore, P_f, P_s yield almost the same results in this scenario since they provide the same stationary distribution (equal weight to each class), so here Classify-WP represents both. We also conducted tests using P_c, which yielded results that are almost the same as Classify, and have been omitted from the table. In conclusion, the performance difference of TD in the imbalanced case arises from the transition probability matrix rather than the bootstrap target. The transition matrix's impact is due to the implied difference in the stationary distribution.

Table 5. Binary classification with imbalance. 0.0073 means 0.73% misclassification rate. The results are smoothed over 5 evaluations before averaging over 5 random seeds.

Dataset \ Algs	**TD-Classify**	**Classify-WP**	**Classify**	**TD-WOF**
Insurance	0.0073 ± 0.0001	0.0073 ± 0.0001	0.0144 ± 0.0003	0.4994 ± 0.0005
Weather	0.0695 ± 0.0008	0.0701 ± 0.0008	0.0913 ± 0.0010	0.4965 ± 0.0031

The usage of inverse link function. The results of TD on classification without using a transformation/link function are presented in Table 5 and are marked by the suffix 'WOF.' These results are not surprising, as the bootstrap estimate can potentially disrupt the TD target entirely. Consider a simple example where a training example $\mathbf{x}$ has a label of one and transitions to another example, also labeled one. Then the reward ($r = y - \gamma y'$) will be $1 - \gamma$. If the bootstrap estimate is negative, the TD target might become close to zero or even negative, contradicting the original training label of one significantly.

On binary classification dataset, weather and insurance, the imbalance ratios (proportion of zeros) are around 72.45%, 88.86% respectively. We set number of iterations to 20k for both and it is sufficient to see convergence. Additionally, in our TD algorithm, to prevent issues with inverse link functions, we add or subtract 1×10^{-15} when applying them to values of 0 or 1 in classification tasks. It should be noted that this parameter can result in invalid values depending on concrete loss implementation, usually setting $< 10^{-7}$ should be generally good.

On those class-imbalanced datasets, when computing the reweighted testing error, we use the function from https://scikit-learn.org/stable/modules/generated/sklearn.metrics.balanced_accuracy_score.html.

C.5 On the Implementation of P

As we mentioned in Section 6, to investigate the effect of transition matrix, we implemented three types of transition matrices: $\mathbf{P}(\mathbf{x}'|\mathbf{x})$ is larger when the two points x, x' 1) are similar; 2) when they are far apart; 3) $\mathbf{P}(\mathbf{x}'|\mathbf{x}) = 1/n, \forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}$. To expedite computations, P_s, P_f are computed based on the training targets instead of the features. The rationale for choosing these options is as follows: the first two may lead to a reduction in the variance of the bootstrap estimate if two consecutive points are positively or negatively correlated.

The resulting matrix may not be a valid stochastic matrix, we use DSM projection [53] to turn it into a valid one.

We now describe the implementation details. For first choice, given two points $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2)$, the formulae to calculate the similarity is:

$$k(y_1, y_2) = \exp(-(y_1 - y_2)^2/v) + 0.1 \tag{97}$$

where v is the variance of all training targets divided by training set size. The second choice is simply $1 - \exp(-(y_1 - y_2)^2/v)$.

Note that the constructed matrix may not be a valid probability transition matrix. To turn it into a valid stochastic matrix, we want: $\mathbf{P}$ itself must be row-normalized (i.e., $\mathbf{P}\mathbf{1} = \mathbf{1}$). To ensure fair comparison, we want equiprobable visitations for all nodes/points, that is, the stationary distribution is assumed to be uniform: $\boldsymbol{\pi} = \frac{1}{n}\mathbf{1}$.

The following proposition shows the necessary and sufficient conditions of the uniform stationary distribution property:

Proposition C.1. *$\boldsymbol{\pi} = \frac{1}{n}\mathbf{1}$ is the stationary distribution of an ergodic $\mathbf{P}$ if and only if $\mathbf{P}$ is a doubly stochastic matrix (DSM).*

Proof. **If**: Note that

$$\pi_j = \sum_i \pi_i p_{ij} \quad \text{and} \quad 1 = \sum_i p_{ij} \tag{98}$$

Subtracting these two gives

$$1 - \pi_j = \sum_i (1 - \pi_i) p_{ij} \quad \text{and} \quad \frac{1 - \pi_j}{n - 1} = \sum_i \frac{1 - \pi_i}{n - 1} p_{ij}.$$

This last equation indicates that $\left(\frac{1-\pi_1}{n-1}, \frac{1-\pi_2}{n-1}, \cdots, \frac{1-\pi_n}{n-1}\right)^\top$ is also the stationary distribution of $\mathbf{P}$ (note that it is non-negative and sum to one). Due to the uniqueness of the stationary distribution, we must have

$$\frac{1 - \pi_i}{n - 1} = \pi_i$$

and thus $\pi_i = \frac{1}{n}, \forall i$.

Only if: Since $\boldsymbol{\pi} = \frac{1}{n}\mathbf{1}$ is the stationary distribution of $\mathbf{P}$, we have

$$\frac{1}{n}\mathbf{1}^\top \mathbf{P} = \frac{1}{n}\mathbf{1}^\top$$

and thus $\mathbf{1}^\top \mathbf{P} = \mathbf{1}^\top$ indicating that $\mathbf{P}$ is column-normalized. Since $\mathbf{P}$ is row-normalized by definition, it is doubly stochastic. □

With linear function approximation,

$$\mathbf{A} = \mathbf{X}^\top \mathbf{D}(\mathbf{I} - \gamma\lambda\mathbf{P})^{-1}(\mathbf{I} - \gamma\mathbf{P})\mathbf{X} \tag{99}$$

$$\mathbf{b} = \mathbf{X}^\top \mathbf{D}(\mathbf{I} - \gamma\lambda\mathbf{P})^{-1}(\mathbf{I} - \gamma\mathbf{P})\mathbf{y} \tag{100}$$

where $\mathbf{X} \in \mathbb{R}^{n\times d}$ is the feature matrix, and $\mathbf{D}$ is uniform when $\mathbf{P}$ is a DSM. As a result, we can apply a DSM [53] projection method to our similarity matrix.

Received 15 May 2025; accepted 14 July 2025