Off-policy learning based on weighted importance sampling with linear computational complexity

A. Rupam Mahmood **Richard S. Sutton**
Reinforcement Learning and Artificial Intelligence Laboratory
Department of Computing Science, University of Alberta, Edmonton, AB T6G 2E8 Canada

Abstract

Importance sampling is an essential component of model-free off-policy learning algorithms. Weighted importance sampling (WIS) is generally considered superior to ordinary importance sampling but, when combined with function approximation, it has hitherto required computational complexity that is $O(n^2)$ or more in the number of features. In this paper we introduce new off-policy learning algorithms that obtain the benefits of WIS with $O(n)$ computational complexity. Our algorithms maintain for each component of the parameter vector a measure of the extent to which that component has been used in previous examples. This measure is used to determine component-wise step sizes, merging the ideas of stochastic gradient descent and sample averages. We present our main WIS-based algorithm first in an intuitive acausal form (the forward view) and then derive a causal algorithm using eligibility traces that is equivalent but more efficient (the backward view). In three small experiments, our algorithms performed significantly better than prior $O(n)$ algorithms for off-policy policy evaluation. We also show that our adaptive step-size technique alone can improve the performance of *on-policy* algorithms such as TD(λ) and true online TD(λ).

1 Weighted importance sampling for off-policy Monte Carlo estimation

In off-policy learning problems, an agent learns about a policy while its experience is generated by following a different policy. A Monte Carlo technique known as *importance sampling* (Kahn & Marshall 1953, Rubinstein 1981) is often used to resolve this mismatch in policies (Sutton & Barto 1998, Dann, Neumann & Peters 2014, Geist & Scherrer 2014). One of the most effective variants of importance sampling is *weighted importance sampling* (WIS), which often gives much lower variance than the ordinary form of importance sampling and is generally preferred due to its superior empirical performance (Hesterberg 1988, Precup, Sutton & Singh 2000, Shelton 2001, Liu 2001, Robert & Casella 2004, Koller & Friedman 2009).

Parametric function approximation is a widely used and viewed as essential for large-scale reinforcement learning applications. However, only recently has WIS been extended to this more general setting. Mahmood, van Hasselt and Sutton (2014) have recently developed WIS-LSTD(λ), which extends WIS from tabular off-policy learning to linear function approximation and eligibility traces. However, WIS-LSTD(λ) is a least-squares algorithm that involves a matrix inversion in its update, the computational complexity of which scales $O(n^3)$ in the number of features. In large-scale applications, $O(n)$ algorithms, such as stochastic gradient descent, temporal-difference (TD) learning, and its gradient-based variants, are often preferred.

WIS has not yet been extended to $O(n)$ algorithms. Modern off-policy algorithms with $O(n)$ computational complexity, such as GTD(λ) (Maei 2011), GQ(λ) (Maei & Sutton 2010), and true online GTD(λ) (van Hasselt, Mahmood & Sutton 2014), all use the ordinary variant of importance sampling and can suffer severely due to the problem of large variance (Defazio & Graepel 2014).

In this work, we take several steps to bridge the gap between the Monte Carlo estimator WIS and $O(n)$ off-policy algorithms with function approximation. The key to this endeavor is to relate stochastic gradient descent (SGD) to simple Monte Carlo estimators such as the sample average. We realize that there is a missing link in that, when the function approximation setting reduces to the tabular setting, SGD does not mimic the sample average estimator. This is not unique to off-policy learning, and the relationship is missing even in the on-policy setting. Our first key step is to develop a new SGD that bridges this gap in an on-policy supervised-learning setting. Using insights from this step, we subsequently develop $O(n)$ off-policy algorithms based on WIS in a general reinforcement learning setting.

2 Sample averages and stochastic gradient descent

In this section, we investigate the relationship between the sample average estimator and SGD. We first show that SGD does not reduce to the sample average estimator in the fully-representable case also known as the *tabular* representation. Then we propose a modification to SGD and show that it achieves the sample average estimator when the feature representation is tabular.

The sample average is one of the simplest Monte Carlo estimators. In order to introduce it, consider that data arrives as a sequence of samples $Y_k \in \mathbb{R}$ drawn from a fixed distribution. The goal of the learner is to estimate the expected value of the samples, $v \doteq \mathbb{E}[Y_k]$. The sample average estimator $\hat{V}_{t+1}$ for data given up to time t can be defined and incrementally updated in the following way:

$$\hat{V}_{t+1} \doteq \frac{\sum_{k=1}^{t} Y_k}{t} = \hat{V}_t + \frac{1}{t}\left(Y_t - \hat{V}_t\right);\ \hat{V}_1 \doteq 0. \quad (1)$$

In the incremental update, $\frac{1}{t}$ can be viewed as a form of step size, modulating the amount of change made to the current estimate, which decreases with time in this case. In the parametric function approximation case, we have to go beyond sample average and use stochastic approximation methods such as SGD.

To introduce SGD, we use a supervised-learning setting with linear function approximation. In this setting, data arrives as a sequence of input-output pairs (X_k, Y_k), where X_k takes values from a finite set $\mathcal{X}$ and $Y_k \in \mathbb{R}$. The learner observes a feature representation of the inputs, where each input is mapped to a feature vector $\boldsymbol{\phi}_k \doteq \boldsymbol{\phi}(X_k) \in \mathbb{R}^n$. The goal of the learner is to estimate the conditional expectation of Y_k for each unique input $x \in \mathcal{X}$ as a linear function of the features: $\boldsymbol{\theta}^\top \boldsymbol{\phi}(x) \approx v(x) \doteq \mathbb{E}[Y_k | X_k = x]$. SGD incrementally updates the parameter vector $\boldsymbol{\theta} \in \mathbb{R}^n$ at each time step t in the following way:

$$\boldsymbol{\theta}_{t+1} \doteq \boldsymbol{\theta}_t + \alpha_t \left(Y_t - \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_t\right) \boldsymbol{\phi}_t, \quad (2)$$

where $\alpha_t > 0$ is a scalar step-size parameter, which is often set to a small constant. The per-update time and memory complexity of SGD is $O(n)$.

Linear function approximation includes tabular representations as a special case. For example, if the feature vectors are $|\mathcal{X}|$-dimensional standard basis vectors, then each feature uniquely represents an input, and the feature representation becomes tabular.

We are interested in finding whether SGD degenerates to sample average when the linear function approximation setting reduces to the tabular setting. Both incremental updates are in a form where the previous estimate is incremented with a product of an error and a step size. In the SGD update, the product also has the feature vector as a factor, but in the tabular setting it simply selects the input for which an update is made.

A major difference between SGD and sample average is the ability of SGD to track under non-stationarity through the use of a constant step size. Typically, the step size of SGD is set to a constant value or decreased with time, where the latter does not work well under non-stationarity but is similar to how sample average works. While we attempt to accommodate sample average estimation more closely within SGD, it is also desirable to retain the tracking ability of SGD.

SGD clearly cannot achieve sample average with a constant step size. On the other hand, if we set the step-size parameter in the SGD update as $\alpha_t = \frac{1}{t}$, the SGD update still does not subsume the sample average. This is because, in the SGD update (2), time t is the total number of samples seen so far, whereas, in the sample average update (1), it is the number of samples seen so far only for one specific input.

We take two important steps to bridge the gap between the sample average update and the SGD update. First, we extend the sample average estimator to incorporate tracking through recency weighting, where the amount of weight assigned to the recent samples is modulated by a scalar recency-weighting constant. This new *recency-weighted average* estimator subsumes sample average as a special case and hence unifies both tracking and sample averaging. Second, we propose a variant of SGD that reduces to recency-weighted average in the tabular setting and still uses only $O(n)$ per-update memory and computation.

Our proposed recency-weighted average estimator can be derived by minimizing an empirical mean squared objective with recency weighting:

$$\tilde{V}_{t+1} \doteq \arg\min_{v} \frac{1}{t} \sum_{k=1}^{t} (1-\eta)^{t-k} \left(Y_k - v\right)^2; 0 \leq \eta < 2.$$

Here, the recency-weighting constant η exponentially weights the past observations down and thus gives more weight to the recent samples. When $\eta = 0$, all samples are weighted equally. The recency-weighted average can be defined and incrementally updated as follows:

$$\tilde{V}_{t+1} = \frac{\sum_{k=1}^{t}(1-\eta)^{t-k} Y_k}{\sum_{k=1}^{t}(1-\eta)^{t-k}} = \tilde{V}_t + \frac{1}{\tilde{U}_{t+1}}\left(Y_k - \tilde{V}_t\right), \quad (3)$$

$$\tilde{U}_{t+1} \doteq (1-\eta)\tilde{U}_t + 1; \qquad \tilde{U}_1 \doteq 0, \quad \tilde{V}_1 \doteq 0. \quad (4)$$

It is easy to see that the recency-weighted average is an unbiased estimator of v. Moreover, when $\eta = 0$, it reduces to the sample average estimator.

Now, we propose a modified SGD in the supervised-learning setting that for tabular representation reduces to the recency-weighted average. The updates are as follows:

$$\mathbf{u}_{t+1} \doteq (\mathbf{1} - \eta \boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t) \circ \mathbf{u}_t + \boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t, \quad (5)$$

$$\boldsymbol{\alpha}_{t+1} \doteq \mathbf{1} \oslash \mathbf{u}_{t+1}, \tag{6}$$

$$\boldsymbol{\theta}_{t+1} \doteq \boldsymbol{\theta}_t + \boldsymbol{\alpha}_{t+1} \circ \left(Y_t - \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_t\right) \boldsymbol{\phi}_t, \tag{7}$$

where $\eta \geq 0$ is the recency-weighting factor, $\circ$ is component-wise vector multiplication, $\oslash$ is component-wise vector division where a division by zero results in zero, and $\mathbf{1} \in \mathbb{R}^n$ is a vector of all ones. Here, $\boldsymbol{\alpha}_{t+1} \in \mathbb{R}^n$ is a vector step-size parameter, set as the vector division of $\mathbf{1}$ by $\mathbf{u}_{t+1} \in \mathbb{R}^n$, which parallels $\tilde{U}$ of the recency-weighted average. We call $\mathbf{u}$ the *usage vector*, as it can be seen as an estimate of how much each feature is "used" over time by the update. We call this algorithm the *usage-based SGD* (U-SGD). Replacing a division by zero with zero in the step-size vector amounts to having no updates for the corresponding component. This makes sense because a zero in any component of $\mathbf{u}$ can occur only at the beginning when $\mathbf{u}$ is initialized to zero and the corresponding feature has not been activated yet. Once a feature is nonzero, the corresponding component of the step-size vector becomes positive and, with sufficiently small positive η, it always remains so.

In the following theorem, we show that U-SGD reduces to recency-weighted average in the tabular setting and hence is a generalization of the sample average estimator as well.

Theorem 1 (Backward consistency of U-SGD with sample average)**.** *If the feature representation is tabular, the vectors* $\mathbf{u}$ *and* $\boldsymbol{\theta}$ *are initially set to zero, and* $0 \leq \eta < 2$*, then U-SGD defined by (5)-(7) degenerates to the recency-weighted average estimator defined by (3) and (4), in the sense that each component of the parameter vector* $\boldsymbol{\theta}_{t+1}$ *of U-SGD becomes the recency-weighted average estimator of the corresponding input.*

(Proved in Appendix A.1).

3 WIS and off-policy SGD

In this section, we carry over weighted importance sampling (WIS) to off-policy SGD, drawing from the ideas developed in the previous section. We introduce two new off-policy SGD algorithms based on WIS. The first one subsumes WIS fully but does not lead to an $O(n)$ implementation, whereas the other algorithm is more amenable to an efficient implementation.

First we introduce both the ordinary importance sampling (OIS) and WIS. Importance sampling is a technique for estimating an expectation under one distribution using samples drawn from a different distribution. OIS estimates the expectation by forming a special kind of sample average. Consider that samples $Y_k \in \mathbb{R}$ are drawn from a sample distribution l, but the goal of the learner is to estimate the expectation $v_g \doteq \mathbb{E}_g\left[Y_k\right]$ under a different distribution g. OIS estimates v_g by scaling each sample Y_k by the importance-sampling ratio $W_k \doteq \frac{g(Y_k)}{l(Y_k)}$ and forming a sample average estimate of the scaled samples:

$$\tilde{V}_{t+1} \doteq \frac{\sum_{k=1}^t W_k Y_k}{t} = \tilde{V}_t + \frac{1}{t}\left(W_t Y_t - \tilde{V}_t\right); \quad \tilde{V}_1 = 0.$$

WIS, on the other hand, estimates v_g by forming a weighted average estimate of the original samples. Its definition and incremental update are as follows:

$$\hat{V}_{t+1} \doteq \frac{\sum_{k=1}^t W_k Y_k}{\sum_{k=1}^t W_k} = \hat{V}_t + \frac{1}{\hat{U}_{t+1}} W_t \left(Y_k - \hat{V}_t\right),$$

$$\hat{U}_{t+1} \doteq \hat{U}_t + W_t; \qquad \hat{U}_1 = 0, \quad \hat{V}_1 = 0.$$

If there is no discrepancy between the sample and the target distribution, then $W_k = 1, \forall k$, and both OIS and WIS become equivalent to the sample average estimator.

We derive the *recency-weighted WIS* as a solution to a mean squared objective with recency weighting and additionally importance sampling:

$$\bar{V}_{t+1} \doteq \arg\min_v \frac{1}{t}\sum_{k=1}^t (1-\eta)^{t-k} W_k \left(Y_k - v\right)^2; 0 \leq \eta < 2,$$

$$= \frac{\sum_{k=1}^t (1-\eta)^{t-k} W_k Y_k}{\sum_{k=1}^t (1-\eta)^{t-k} W_k}.$$

It is easy to see that, when $\eta = 0$, the recency-weighted WIS estimator reduces to WIS. Recency-weighted WIS can be updated incrementally in the following way:

$$\bar{V}_{t+1} = \bar{V}_t + \frac{1}{\bar{U}_{t+1}} W_t \left(Y_t - \bar{V}_t\right); \quad \bar{V}_1 = 0, \tag{8}$$

$$\bar{U}_{t+1} \doteq (1-\eta)\bar{U}_t + W_t; \qquad \bar{U}_1 = 0. \tag{9}$$

Now we introduce two variants of SGD based on WIS in a more general off-policy reinforcement learning setting with linear function approximation. In this setting, a learning agent interacts with an environment by taking an action $A_k \in \mathcal{A}$ in a state of the environment $S_k \in \mathcal{S}$ on each time step k. Here $\mathcal{A}$ and $\mathcal{S}$ are considered finite. Upon taking an action, the agent receives a scalar reward R_{k+1} and transitions to state S_{k+1}. Instead of observing the states directly, the agent observes a feature representation of the states where each state is mapped to a feature vector $\boldsymbol{\phi}_k \doteq \boldsymbol{\phi}(S_k) \in \mathbb{R}^n$. In an off-policy policy-evaluation problem, the agent takes actions based on a fixed behavior policy $b(\cdot|S_k)$. The goal is to estimate the $v_\pi(s)$ (the expected sum of the future discounted reward when starting in s and following π) as a linear function of the features:

$$\boldsymbol{\theta}^\top \boldsymbol{\phi}(s) \approx v_\pi(s) \doteq \mathbb{E}\left[G_k | S_k = s, A_i \sim \pi(\cdot|S_i), \forall i\right],$$

$$G_k \doteq \sum_{i=k}^{\infty} R_{i+1} \prod_{j=k+1}^{i} \gamma(S_j),$$

where $\boldsymbol{\theta} \in \mathbb{R}^n$ is the parameter vector to learn, and $\gamma_k \doteq \gamma(S_k) \in [0,1]$ denotes a state-dependent discounting (e.g.,

see Sutton et al. 2014). In a general value-function setting, such state-dependent discounting can be used to denote termination in a state s by setting $\gamma(s) = 0$ (Sutton et al. 2011).

In order to learn $\boldsymbol{\theta}$ model-free from samples, we need to use an importance sampling technique since the behavior policy and the target policy can be different. Given a partial trajectory from time k to $t+1$: $S_k, A_k, R_{k+1}, \cdots, S_{t+1}$, the importance-sampling ratio ρ_k^{t+1} is defined as the likelihood ratio of this trajectory starting at S_k under the target policy and the behavior policy:

$$\rho_k^{t+1} \doteq \frac{\prod_{i=k}^{t} p(S_{i+1}|S_i, A_i)\pi(A_i|S_i)}{\prod_{i=k}^{t} p(S_{i+1}|S_i, A_i)b(A_i|S_i)} = \prod_{i=k}^{t} \frac{\pi(A_i|S_i)}{b(A_i|S_i)}$$
$$= \prod_{i=k}^{t} \rho_i; \qquad \rho_i \doteq \rho_i^{i+1},$$

where p is the state-transition probability function. The value $v_\pi(s)$ can be estimated using returns scaled by the corresponding importance-sampling ratios. Consider that data is available up to step $t+1$ and no termination or discounting occurs by that time. A return originating from state S_k can be approximated by using a full discounting at the final step: $\gamma_{t+1} = 0$. Then a flat truncated return can be defined as (Sutton et al. 2014):

$$G_k^{t+1} \doteq \sum_{i=k}^{t} R_{i+1}.$$

When the states are visible and the number of states are small, the value $v_\pi(s)$ for each state s can be estimated using importance sampling such as OIS, WIS or recency-weighted WIS by setting $Y_k \doteq G_k^{t+1}$ and $W_k \doteq \rho_k^{t+1}$.

Now, we propose the first off-policy SGD based on WIS, which we call *WIS-SGD-1*. With $0 \leq k < t+1$ and $\boldsymbol{\theta}_0^t \doteq \boldsymbol{\theta}_0, \forall t$, the following updates define WIS-SGD-1:

$$\mathbf{u}_{k+1}^{t+1} \doteq (\mathbf{1} - \eta\boldsymbol{\phi}_k \circ \boldsymbol{\phi}_k) \circ \mathbf{u}_k^{t+1} + \rho_k^{t+1}\boldsymbol{\phi}_k \circ \boldsymbol{\phi}_k, \quad (10)$$
$$\boldsymbol{\alpha}_{k+1}^{t+1} \doteq \mathbf{1} \oslash \mathbf{u}_{k+1}^{t+1}, \quad (11)$$
$$\boldsymbol{\theta}_{k+1}^{t+1} \doteq \boldsymbol{\theta}_k + \boldsymbol{\alpha}_{k+1}^{t+1} \circ \rho_k^{t+1}\left(G_k^{t+1} - \boldsymbol{\phi}_k^\top \boldsymbol{\theta}_k^{t+1}\right)\boldsymbol{\phi}_k. \quad (12)$$

Similar to U-SGD, WIS-SGD-1 maintains a vector step size through the update of a usage vector, which in this case also includes the importance-sampling ratios. Unlike U-SGD, the parameters of WIS-SGD-1 use two time indices. The time index in the subscript corresponds to the time step of the prediction, and the time index in the superscript stands for the data horizon. In the following, we show that WIS-SGD-1 reduces to recency-weighted WIS, and hence to WIS as well, in the tabular setting.

Theorem 2 (Backward consistency of WIS-SGD-1 with WIS)**.** *If the feature representation is tabular, the vectors* $\mathbf{u}$ *and* $\boldsymbol{\theta}$ *are initially set to zero, and* $0 \leq \eta < 2$*, then WIS-SGD-1 defined by (10)-(12) degenerates to recency-weighted WIS defined by (8) and (9) with* $Y_k \doteq G_k^{t+1}$ *and* $W_k \doteq \rho_k^{t+1}$*, in the sense that each component of the parameter vector* $\boldsymbol{\theta}_{t+1}^{t+1}$ *of WIS-SGD-1 becomes the recency-weighted WIS estimator of the corresponding input.* (Proved in Appendix A.2.)

Now we focus on whether and how WIS-SGD-1 can be implemented efficiently. The updates as defined above cannot be computed in $O(n)$ per time step. An update for step k requires computing an importance-sampling ratio and a flat truncated return that are available only at $t+1 > k$. It can be computed by looking ahead into the future from k, but then the update becomes acausal. It can alternatively be computed by waiting until time $t+1$ and iterating for each k. But then the update made at $t+1$ becomes expensive, scaling linearly with t, that is, $O(tn)$.

Such updates, where samples are available in future from the time step when the update is made, are often known as *forward-view* updates (Sutton & Barto 1998). Forward-view updates are typically expensive, but for some forward-view updates it is possible to derive causal and efficient updates, known as *backward-view* updates, that compute exactly the same estimate at each time step. Classically these equivalences were achieved for offline updating. Van Seijen and Sutton (2014) showed that such equivalences can also be achieved in the online case.

Converting a forward-view update into an efficient backward-view update depends on combining the extra data available at $t+1$ with the current estimate $\boldsymbol{\theta}_t^t$ in an efficient way to give the next estimate $\boldsymbol{\theta}_{t+1}^{t+1}$. For linear recursive updates, it is tantamount to unrolling both $\boldsymbol{\theta}_{t+1}^{t+1}$ and $\boldsymbol{\theta}_t^t$ and finding whether their difference can be written in terms of data available only in the recent-most step. It is often not possible to achieve such efficient backward-view updates, and we believe WIS-SGD-1 is one such case.

To appreciate why an efficient backward-view update of WIS-SGD-1 is not plausible, consider the update of $\boldsymbol{\theta}_t^t$ unrolled back to the beginning of time:

$$\boldsymbol{\theta}_t^t = \left(\mathbf{I} - \rho_{t-1}^t(\boldsymbol{\alpha}_t^t \circ \boldsymbol{\phi}_{t-1})\boldsymbol{\phi}_{t-1}^\top\right)\boldsymbol{\theta}_{t-1}^t + \rho_{t-1}^t G_{t-1}^t \boldsymbol{\phi}_{t-1}$$
$$= \prod_{k=0}^{t-1}\left(\mathbf{I} - \rho_k^t(\boldsymbol{\alpha}_{k+1}^t \circ \boldsymbol{\phi}_k)\boldsymbol{\phi}_k^\top\right)\boldsymbol{\theta}_0^t$$
$$+ \sum_{k=0}^{t-1} \rho_k^t G_k^t \prod_{j=k+1}^{t-1}\left(\mathbf{I} - \rho_j^t(\boldsymbol{\alpha}_{j+1}^t \circ \boldsymbol{\phi}_j)\boldsymbol{\phi}_j^\top\right)\boldsymbol{\phi}_k.$$

In order to obtain $\boldsymbol{\theta}_{t+1}^{t+1}$ by combining the new data $\boldsymbol{\phi}_t$ and R_{t+1} with $\boldsymbol{\theta}_t^t$, it is evident that each of the $\rho_k^t(\boldsymbol{\alpha}_{k+1}^t \circ \boldsymbol{\phi}_k)\boldsymbol{\phi}_k^\top$ terms in the first product needs to be replaced by $\rho_k^{t+1}(\boldsymbol{\alpha}_{k+1}^{t+1} \circ \boldsymbol{\phi}_k)\boldsymbol{\phi}_k^\top$, which is unlikely to be achieved in an inexpensive way. This problem does not appear in previous algorithms with online equivalence such as true online TD(λ) (van Seijen & Sutton 2014) or true online GTD(λ)

(van Hasselt, Mahmood & Sutton 2014), because the terms involved in the product of the unrolled update in those algorithms do not involve *forward-view* terms, that is, they contain ρ_k and α_{k+1} in those products instead of ρ_k^{t+1} and $\boldsymbol{\alpha}_{k+1}^{t+1}$. This specific problem with WIS-SGD-1 is due to the fact that the error of the update in (12) is multiplied by the forward-view terms ρ_k^{t+1} and $\boldsymbol{\alpha}_{k+1}^{t+1}$.

The observation we made in the above leads us to develop a second off-policy SGD. In this algorithm, first we replace the forward-view term $\boldsymbol{\alpha}_{k+1}^{t+1}$ from the update of $\boldsymbol{\theta}$ with $\boldsymbol{\alpha}_{k+1}^{k+1}$. Second, instead of multiplying the terms in the error $G_k^{t+1} - \boldsymbol{\phi}_k^\top \boldsymbol{\theta}_k^{t+1}$ with the same forward-view term ρ_k^{t+1}, we multiply the first term G_k^{t+1} by ρ_k^{t+1} and the second term $\boldsymbol{\phi}_k^\top \boldsymbol{\theta}_k^{t+1}$ by ρ_k. To account for this discrepancy, we add two more terms in the error, and the resultant error of the new update becomes $\rho_k^{t+1} G_k^{t+1} - \rho_k^{t+1} \boldsymbol{\phi}_k^\top \boldsymbol{\theta}_{k-1}^{k-1} + \rho_k \boldsymbol{\phi}_k^\top \boldsymbol{\theta}_{k-1}^{k-1} - \rho_k \boldsymbol{\phi}_k^\top \boldsymbol{\theta}_k^{t+1}$. Here, the first two terms are approximating the WIS-SGD-1 error $\rho_k^{t+1}\left(G_k^{t+1} - \boldsymbol{\phi}_k^\top \boldsymbol{\theta}_k^{t+1}\right)$, whereas the last two terms are adding a bias. Although this new algorithm no longer reduces to WIS in the tabular setting, it is developed based on WIS and still retains the main ideas behind recency-weighted WIS. Hence, we call this algorithm *WIS-SGD-2*. The following updates define WIS-SGD-2, with $0 \leq k < t+1$:

$$\mathbf{u}_{k+1}^{t+1} \doteq (\mathbf{1} - \eta \boldsymbol{\phi}_k \circ \boldsymbol{\phi}_k) \circ \mathbf{u}_k^{t+1} + \rho_k^{t+1} \boldsymbol{\phi}_k \circ \boldsymbol{\phi}_k, \quad (13)$$

$$\boldsymbol{\alpha}_{k+1} \doteq \mathbf{1} \oslash \mathbf{u}_{k+1}^{k+1}, \quad (14)$$

$$\delta_k^{t+1} \doteq \rho_k^{t+1} G_k^{t+1} - \rho_k^{t+1} \boldsymbol{\phi}_k^\top \boldsymbol{\theta}_{k-1} + \rho_k \boldsymbol{\phi}_k^\top \boldsymbol{\theta}_{k-1} - \rho_k \boldsymbol{\phi}_k^\top \boldsymbol{\theta}_k^{t+1}, \quad (15)$$

$$\boldsymbol{\theta}_{k+1}^{t+1} \doteq \boldsymbol{\theta}_k^{t+1} + \boldsymbol{\alpha}_{k+1} \circ \delta_k^{t+1} \boldsymbol{\phi}_k. \quad (16)$$

Here, $\boldsymbol{\theta}_k \doteq \boldsymbol{\theta}_k^k$, and $\boldsymbol{\theta}_{-1} = \mathbf{0}$. It can be easily verified that, in the on-policy case, WIS-SGD-2 degenerates to U-SGD and hence retains the backward consistency with the sample average estimator.

Although this algorithm has much more plausibility of having an efficient backward view due to the careful modifications, it is not yet immediately clear how such a backward-view update can be obtained. Van Hasselt, Mahmood and Sutton (2014) introduced an online equivalence technique from which both true online TD(λ) and true online GTD(λ) can be derived. Their technique requires the target in the error to have a specific recurrence relation. Unfortunately, that specific relation does not hold for the target in WIS-SGD-2. A new technique is needed in order to derive an efficient backward view for WIS-SGD-2.

4 A new online equivalence technique

In this section, we introduce a new technique for deriving efficient backward views from online forward-view updates. We show that this technique subsumes the existing technique for online equivalences (van Hasselt, Mahmood & Sutton 2014). The following theorem describes the new online equivalence technique (Proved in Appendix A.3).

Theorem 3 (Online equivalence technique)**.** *Consider any forward view that updates toward an interim scalar target Y_k^t with*

$$\boldsymbol{\theta}_{k+1}^{t+1} \doteq \mathbf{F}_k \boldsymbol{\theta}_k^{t+1} + Y_k^{t+1} \mathbf{w}_k + \mathbf{x}_k, \quad 0 \leq k < t+1,$$

where $\boldsymbol{\theta}_0^t \doteq \boldsymbol{\theta}_0$ for some initial $\boldsymbol{\theta}_0$, and both $\mathbf{F}_k \in \mathbb{R}^{n \times n}$ and $\mathbf{w}_k \in \mathbb{R}^n$ can be computed using data available at k. Assume that the temporal difference $Y_k^{t+1} - Y_k^t$ at k is related to the temporal difference at $k+1$ as follows:

$$Y_k^{t+1} - Y_k^t = d_{k+1}\left(Y_{k+1}^{t+1} - Y_{k+1}^t\right) + b_t g_k \prod_{j=k+1}^{t-1} c_j, 0 \leq k < t,$$

where b_k, c_k, d_k and g_k are scalars that can be computed using data available at time k. Then the final weight $\boldsymbol{\theta}_{t+1} \doteq \boldsymbol{\theta}_{t+1}^{t+1}$ can be computed through the following backward-view updates, with $\mathbf{e}_{-1} \doteq \mathbf{0}$, $\mathbf{d}_0 \doteq \mathbf{0}$, and $t \geq 0$:

$$\begin{aligned}
\mathbf{e}_t &\doteq \mathbf{w}_t + d_t \mathbf{F}_t \mathbf{e}_{t-1}, \\
\boldsymbol{\theta}_{t+1} &\doteq \mathbf{F}_t \boldsymbol{\theta}_t + (Y_t^{t+1} - Y_t^t)\mathbf{e}_t + Y_t^t \mathbf{w}_t + b_t \mathbf{F}_t \mathbf{d}_t + \mathbf{x}_t, \\
\mathbf{d}_{t+1} &\doteq c_t \mathbf{F}_t \mathbf{d}_t + g_t \mathbf{e}_t.
\end{aligned}$$

This equivalence technique allows producing backward views that contain a *dutch* trace $\mathbf{e}$ (van Hasselt et al. 2014) and an extra set of weights $\mathbf{d}$ known as *provisional* weights (Sutton et al. 2014) at the same time. In previous works (Sutton et al. 2014, Mahmood et al. 2014), the provisional weights appeared only in offline updates. Due to this online equivalence technique, this is the first time the provisional weights have emerged in online updates.

In the following theorem, we show that the new equivalence technique is a generalization of the existing equivalence technique developed by van Hasselt et al. (2014). Hence, it readily follows that the existing algorithms with an online equivalence, such as true online TD(λ) and true online GTD(λ), can be derived using the new equivalence technique. The proof is given in Appendix A.4.

Theorem 4 (Generality of the new equivalence technique)**.** *The online equivalence technique by van Hasselt et al. (2014, Theorem 1) can be retrieved as a special case of the online equivalence technique given in Theorem 3.*

5 A new off-policy TD(λ) based on WIS

In this section, we develop a new off-policy algorithm that generalizes WIS-SGD-2 to partial termination and bootstrapping. Then we use the new online equivalence technique to derive an equivalent $O(n)$ backward-view update. We use a state-dependent bootstrapping parameter $\lambda_k \doteq \lambda(S_k) \in [0, 1]$ in developing the new algorithm.

First, we construct the target, and then we define a new update for the usage vector $\mathbf{u}$ in this more general setting.

Based on the general off-policy forward view by Sutton et al. (2014), we combine truncated returns G_k^{t+1} and truncated corrected returns $G_k^{t+1} + \boldsymbol{\phi}_{t+1}^\top \boldsymbol{\theta}_t$ scaled by corresponding weights due to discounting, bootstrapping and importance sampling to develop an overall return:

$$G_{k,t+1}^{\rho} \doteq \rho_k C_k^t \Big((1-\gamma_{t+1}) G_k^{t+1} + \gamma_{t+1} \left(G_k^{t+1} + \boldsymbol{\phi}_{t+1}^\top \boldsymbol{\theta}_t \right) \Big) + \sum_{i=k+1}^{t} \rho_k C_k^{i-1} \Big((1-\gamma_i) G_k^i + \gamma_i (1-\lambda_i) \left(G_k^i + \boldsymbol{\phi}_i^\top \boldsymbol{\theta}_{i-1} \right) \Big) - \rho_k \left(C_k^t + \sum_{i=k+1}^{t} C_k^{i-1} (1-\gamma_i \lambda_i) - 1 \right) \boldsymbol{\phi}_k^\top \boldsymbol{\theta}_{k-1}, \quad (17)$$

where $C_k^t \doteq \prod_{j=k+1}^{t} \gamma_j \lambda_j \rho_j$, $\boldsymbol{\theta}_k \doteq \boldsymbol{\theta}_k^k$, $0 \leq k < t+1$ and $\boldsymbol{\theta}_{-1} = \mathbf{0}$. It can be readily verified that, when no bootstrapping is used, that is, $\lambda_k = 1, \forall k$ and discounting occurs only at the data horizon $t+1$, that is, $\gamma_0 = \gamma_1 = \cdots = \gamma_t = 1$ and $\gamma_{t+1} = 0$, then $G_{k,t+1}^{\rho} = \rho_k^{t+1} G_k^{t+1} - \rho_k^{t+1} \boldsymbol{\phi}_k^\top \boldsymbol{\theta}_{k-1} + \rho_k \boldsymbol{\phi}_k^\top \boldsymbol{\theta}_{k-1}$. Hence $G_{k,t+1}^{\rho}$ is a strict generalization of the WIS-SGD-2 target to the state-dependent discounting and bootstrapping.

The usage vector $\mathbf{u}$ of the WIS-SGD algorithms rescales the components of the parameter updates in order to clamp down the updates proportionally when they become large due to large importance-sampling ratios. However, when bootstrapping is used, larger trajectories are given smaller weights, and hence their corresponding importance-sampling ratios will have less severe effect on the updates. For example, when full bootstrapping is used, that is, $\lambda_k = 0, \forall k$, the overall return becomes $G_{k,t+1}^{\rho} = \rho_k \left(R_{k+1} + \gamma_{k+1} \boldsymbol{\phi}_{k+1}^\top \boldsymbol{\phi}_k \right)$, with an importance-sampling ratio of a one-transition long trajectory. In such cases, updating $\mathbf{u}$ with the importance-sampling ratio of the full trajectory ρ_k^{t+1} is unnecessary. Hence, the amount of importance weighting in $\mathbf{u}$ at each step should be modulated by the amount of discounting and bootstrapping.

Based on the overall return in (17) and the idea of discounting and bootstrapping-*aware* update of $\mathbf{u}$ discussed above, we propose a new off-policy TD algorithm based on WIS, which we call *WIS-TD(λ)*. It consists of the following forward-view updates:

$$\tilde{\rho}_k^{t+1} \doteq \rho_k \sum_{i=k+1}^{t} C_k^{i-1} (1-\gamma_i \lambda_i) + \rho_k C_k^t; \quad \tilde{\rho}_t^t \doteq 0, \quad (18)$$

$$\mathbf{u}_{k+1}^{t+1} \doteq (\mathbf{1} - \eta \boldsymbol{\phi}_k \circ \boldsymbol{\phi}_k) \circ \mathbf{u}_k^{t+1} + \tilde{\rho}_k^{t+1} \boldsymbol{\phi}_k \circ \boldsymbol{\phi}_k, \quad (19)$$

$$\boldsymbol{\alpha}_{k+1} \doteq \mathbf{1} \oslash \mathbf{u}_{k+1}^{k+1}, \quad (20)$$

$$\boldsymbol{\theta}_{k+1}^{t+1} \doteq \boldsymbol{\theta}_k^{t+1} + \boldsymbol{\alpha}_{k+1} \circ \left(G_{k,t+1}^{\rho} - \rho_k \boldsymbol{\phi}_k^\top \boldsymbol{\theta}_k^{t+1} \right) \boldsymbol{\phi}_k. \quad (21)$$

It can be easily verified that, when no bootstrapping is used, that is, $\lambda_k = 1, \forall k$ and discounting occurs only at the data horizon $t+1$, that is, $\gamma_0 = \gamma_1 = \cdots = \gamma_t = 1$ and $\gamma_{t+1} = 0$, then $\tilde{\rho}_k^{t+1} = \rho_k^{t+1}$, and we already showed that the target of WIS-TD(λ) $G_{k,t+1}^{\rho}$ reduces to the WIS-SGD-2 target in this case. Hence, WIS-TD(λ) subsumes WIS-SGD-2, establishing a direct backward consistency to sample average.

In the following, we apply the new online equivalence technique to the above forward-view update to derive an $O(n)$ backward-view update computing the same parameter vector $\boldsymbol{\theta}_t$ at each t. For that, first we derive an $O(n)$ backward-view update for the step size that computes the same $\boldsymbol{\alpha}_t$ as in the above algorithm at each t.

Theorem 5 (Backward view update for $\boldsymbol{\alpha}_t$ of WIS-TD(λ))**.** *The step-size vector $\boldsymbol{\alpha}_t$ computed by the following backward-view update and the forward-view update defined by (18) – (20) are equal at each step t:*

$$\mathbf{u}_{t+1} \doteq (\mathbf{1} - \eta \boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t) \circ \mathbf{u}_t + \rho_t \boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t + (\rho_t - 1) \gamma_t \lambda_t (\mathbf{1} - \eta \boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t) \circ \mathbf{v}_t, \quad (22)$$

$$\mathbf{v}_{t+1} \doteq \gamma_t \lambda_t \rho_t (\mathbf{1} - \eta \boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t) \circ \mathbf{v}_t + \rho_t \boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t, \quad (23)$$

$$\boldsymbol{\alpha}_{t+1} \doteq \mathbf{1} \oslash \mathbf{u}_{t+1}. \quad (24)$$

(Proved in Appendix A.5.)

Now, we derive an $O(n)$ backward-view update that computes the same $\boldsymbol{\theta}_t^t$ as the above forward view.

Theorem 6 (Backward view update for $\boldsymbol{\theta}_t^t$ of WIS-TD(λ))**.** *The parameter vector $\boldsymbol{\theta}_t$ computed by the following backward-view update and the parameter vector $\boldsymbol{\theta}_t^t$ computed by the forward-view update defined by (17) and (21) are equal at every time step t:*

$$\mathbf{e}_t \doteq \rho_t \boldsymbol{\alpha}_{t+1} \circ \boldsymbol{\phi}_t + \gamma_t \lambda_t \rho_t \left(\mathbf{e}_{t-1} - \rho_t (\boldsymbol{\alpha}_{t+1} \circ \boldsymbol{\phi}_t) \boldsymbol{\phi}_t^\top \mathbf{e}_{t-1} \right), \quad (25)$$

$$\boldsymbol{\theta}_{t+1} \doteq \boldsymbol{\theta}_t + \boldsymbol{\alpha}_{t+1} \circ \rho_t \left(\boldsymbol{\theta}_{t-1}^\top \boldsymbol{\phi}_t - \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_t \right) \boldsymbol{\phi}_t + (R_{t+1} + \gamma_{t+1} \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_{t+1} - \boldsymbol{\theta}_{t-1}^\top \boldsymbol{\phi}_t) \mathbf{e}_t + (\rho_t - 1) \gamma_t \lambda_t \left(\mathbf{d}_t - \rho_t (\boldsymbol{\alpha}_{t+1} \circ \boldsymbol{\phi}_t) \boldsymbol{\phi}_t^\top \mathbf{d}_t \right), \quad (26)$$

$$\mathbf{d}_{t+1} \doteq \gamma_t \lambda_t \rho_t \left(\mathbf{d}_t - \rho_t (\boldsymbol{\alpha}_{t+1} \circ \boldsymbol{\phi}_t) \boldsymbol{\phi}_t^\top \mathbf{d}_t \right) + \left(R_{t+1} + \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_{t+1} - \boldsymbol{\theta}_{t-1}^\top \boldsymbol{\phi}_t \right) \mathbf{e}_t. \quad (27)$$

(Proved in Appendix A.6.)

The overall backward view of WIS-TD(λ) is defined by (22) – (27), and its complete description is given in Appendix A.7. Note that, Theorem 6 does not depend on how $\boldsymbol{\alpha}_{t+1}$ is set. The per-update time and memory complexity of WIS-TD(λ) is $O(n)$. An auxiliary parameter vector might be included in WIS-TD(λ) by making use of the $\mathbf{x}_k$ vector of the online equivalence technique as was done by van Hasselt et al. (2014), but we do not explore this possibility here.

A seemingly related algorithm is fLSTD-SA (Prashanth, Korda & Munos 2014), which is an on-policy stochastic approximation method derived based on the on-policy

LSTD algorithm (Bradtke & Barto 1996) by randomizing the transition samples. One might speculate whether an $O(n)$ off-policy stochastic-approximation algorithm based on WIS can be derived from WIS-LSTD(λ) by applying the techniques used in fLSTD-SA. However, the application of fLSTD-SA in the off-policy case does not appear to achieve any form of WIS. Updating a vector step size based on the usage of the features and importance-sampling weights is a distinctive aspect of our new algorithm and is essential for obtaining the benefits of WIS, which is absent in existing stochastic-approximation methods.

6 Extending existing algorithms based on the new adaptive step size

The vector step-size adaptation based on the update of the usage vector $\mathbf{u}$ is only loosely coupled with WIS-TD(λ) and can be freely combined with existing off-policy algorithms as well as the on-policy ones. When combined with the existing algorithms, this step-size adaptation is expected to yield benefits due to the rescaling it performs according to the magnitude of importance-sampling weights and the frequency of feature activation.

We propose two new off-policy algorithms: *WIS-GTD(λ)* and *WIS-TO-GTD(λ)*, based on GTD(λ) (Maei 2011) and true online GTD(λ) (van Hasselt et al. 2014), respectively. In both algorithms, we propose replacing the scalar step size of the main parameter vector with the vector step size according to (22) – (24). The scalar step-size parameter of the auxiliary parameter vector of GTD(λ) and true online GTD(λ) could also be replaced with the vector step size with a different recency-weighting factor, but we leave it out here. The descriptions of these two algorithms are given in Appendix A.7.

We propose two new on-policy algorithms: *usage-based TD(λ)* (U-TD(λ)) and *usage-based true online TD(λ)* (U-TO-TD(λ)), by combining the vector-step-size adaptation with two existing on-policy algorithms: TD(λ) (Sutton & Barto 1998) and true online TD(λ) (van Seijen & Sutton 2014), respectively. There are interesting interrelationships between these on-policy and off-policy algorithms. For example, WIS-GTD(λ) becomes equivalent to U-TD(λ) in the on-policy case when the second step-size parameter $\beta = 0$. On the other hand, WIS-TD(λ) directly degenerates to U-TO-TD(λ) in the on-policy case, whereas WIS-TO-GTD(λ) reduces to U-TO-TD(λ) in the on-policy case with $\beta = 0$. We provide the description of U-TD(λ) and U-TO-TD(λ) in Appendix A.7.

7 Experimental results

In this section we evaluate the new algorithms using two sets of experiments with off-policy and on-policy policy-evaluation tasks, respectively. Source code for both the off-policy and on-policy experiments are available online[1]. In the first set of experiments, we compared the new off-policy algorithms: WIS-TD(λ), WIS-GTD(λ) and WIS-TO-GTD(λ) with two existing $O(n)$ algorithms: GTD(λ) and true online GTD(λ) (TO-GTD(λ)), and with two least squares algorithms: LSTD-TO(λ), an off-policy algorithm proposed by Dann, Neumann and Peters (2014), and WIS-LSTD(λ), an ideal extension of WIS. For evaluation, we created three off-policy policy-evaluation tasks.

The first task was constructed based on a random-walk Markov chain where the states can be imagined to be laid out on a horizontal line. There were 11 non-terminal states and two terminal states: on the left and the right ends of the chain. From each non-terminal state, there are two actions available: *left*, leads to the state to the left, and *right*, leads to the state to the right. The initial state was always set to the state in the middle of the chain. The reward was sparsely available: 0 for all transitions except for the rightmost transition to the terminal state, where it was $+1$. The behavior policy was uniformly random between the two actions and the target policy chose *right* with 0.99 probability. No discounting was used. The feature vectors were binary representations of state indices. For 11 non-terminal states, each feature vector was of length $\lfloor \log_2(11) \rfloor + 1 = 4$, and these vectors for the states from left to right were $(0,0,0,1)^\top$, $(0,0,1,0)^\top, (0,0,1,1,)^\top, \cdots, (1,0,1,1)^\top$. The features were all zero for the terminal states.

The second and the third tasks were constructed using randomly generated MDPs. We represent a randomly generated MDP as $(N, m, b, \mathbf{\Gamma})$ where N and m stand for the number of states and actions, respectively, and b is a branching factor denoting the number of next states for a given state-action pair. Here, $\mathbf{\Gamma} \in \mathbb{R}^{N \times N}$ is a diagonal matrix where the entries are the state-dependent discounting $\gamma(\cdot)$ for each state. We use such a state-dependent discounting to denote termination under the target policy while experience continues seamlessly under the behavior policy. For each state, the next b states were chosen from total N states randomly without replacement, and the transition probabilities were generated by partitioning the unit interval at $b-1$ cut points which were selected uniformly randomly from $[0, 1]$. The rewards for a transition from a state-action pair to the next state were selected uniformly randomly from $[0, 1]$ and kept deterministic. The behavior policy probabilities for different actions in a particular state were set using uniform random numbers from $[10^{-15}, 1 + 10^{-15}]$ and normalized to sum to one. The target policy is much less stochastic: one of the actions in a particular state is chosen to have probability 0.99 and the rest of the actions are equiprobable.

[1] Source code for off-policy experiments is available at `http://github.com/armahmood/wis-td-experiments` and for on-policy experiments at `http://github.com/armahmood/usage-td-experiments`.

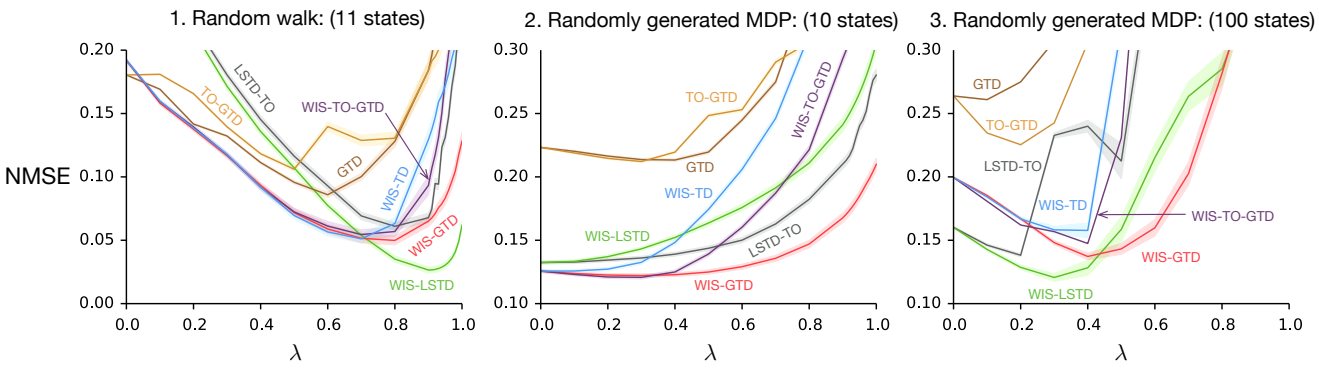

Figure 1: Empirical comparison of the new WIS-based $O(n)$ algorithms with two existing $O(n)$ algorithms and two LSTD algorithms on three off-policy policy-evaluation tasks. Performance is shown in the empirical normalized MSE (NMSE) measured by averaging over 50 independent runs and 100 episodes for the first task, 500 steps for the second, and 5000 steps for the third. The new WIS-based algorithms performed significantly better than both existing $O(n)$ algorithms in all three off-policy tasks and competitively with one of the LSTD algorithms.

We constructed the second task by randomly generating an MDP with parameters $(10, 3, 3, \mathbf{\Gamma})$, where $\gamma(\cdot) = 0$ for 2 randomly chosen states to denote termination under the target policy and $\gamma(\cdot) = 0.99$ for the rest of the 8 states. For the third task, we randomly generated an MDP with parameters $(100, 3, 10, \mathbf{\Gamma})$, where $\gamma(\cdot) = 0$ for 5 randomly chosen states and $\gamma(\cdot) = 0.99$ for the rest. The feature vectors were binary representations of the indices of all states including those for which $\gamma(\cdot) = 0$. In these two tasks, the feature vectors were normalized to have unit length.

We tested all algorithms for different values of constant λ, from 0 to 0.9 in steps of 0.1 and from 0.9 to 1.0 in steps of 0.01. The first step-size parameter α of GTD(λ) and TO-GTD(λ) was varied by powers of 10 with powers chosen from -3 to 0 in steps of 0.25. The second step-size parameter β of both algorithms was varied among values $[0, 0.001, 0.01, 0.1]$. The initial value u_0 of the components of the usage vector $\mathbf{u}$ for WIS-TD(λ), WIS-GTD(λ) and WIS-TO-GTD(λ) was varied by powers of 10 with powers chosen from 0 to 3 in steps of 0.25. The recency-weighting factor η of the same algorithms was set as $\eta = \mu/u_0$, where μ was varied among values $[0, 0.001, 0.01, 0.1, 1]$. The second step-size parameter β for WIS-GTD(λ) and WIS-TO-GTD(λ) was set to zero. The matrix to be inverted in LSTD-TO(λ) and WIS-LSTD(λ) was initialized to $\epsilon\mathbf{I}$, where ϵ was varied by powers of 10 with powers chosen from -3 to $+3$ in steps of 0.2. The initial parameter vector $\boldsymbol{\theta}_0$ was set to $\mathbf{0}$.

Performance was measured as the empirical mean squared error (MSE) between the estimated values of the states and their true values under the target policy projected to the space spanned by the given features. The error was weighted according to the state-visitation distribution under the behavior policy. As the scale of this MSE measure can vary between these tasks, we normalized it by the squared weighted L2 norm of the projected true value, which is equivalent to the MSE under $\boldsymbol{\theta} = \mathbf{0}$. As a result, the initial normalized MSE (NMSE) for each algorithm was 1. For each run, we averaged this error over 100 episodes measured at the end of each episode for the first task, over 500 steps for the second task, and over 5000 steps for the third. We produced the final estimate by further averaging over 50 independent runs.

Figure 1 shows the empirical performance together with standard error on the three off-policy policy-evaluation tasks with respect to different λ and optimized over all other parameters. In all three tasks, the new algorithms significantly outperformed both GTD(λ) and TO-GTD(λ) indicating the effectiveness of the adaptive vector step size in retaining the advantage of WIS. The new algorithms also performed competitively with LSTD-TO(λ) in all tasks. Among the new algorithms, WIS-GTD(λ) performed superiorly with large values of λ.

We also studied the sensitivity of the new algorithms with respect to their parameters. Although these algorithms replace the scalar constant step-size parameter of their base learner with an adaptive vector step size based on feature usage, the estimate of the usage depends on two new parameters: the initial value u_0 and the recency weighting constant η. The initial value u_0 of the usage vector can be interpreted as the inverse of the initial step size, and its tuning can be as extensive as that of the step-size parameter in other algorithms. On the other hand, η can be viewed as the desired final step size. As a result, their product $\mu = u_0\eta$ is unit free and requires less rigorous tuning.

In our final set of experiments, we compared the new $O(n)$ on-policy algorithms: U-TD(λ) and U-TO-TD(λ), with two $O(n)$ on-policy algorithms: TD(λ) with accumulating traces and true online TD(λ), which we call *TO-TD(λ)*.

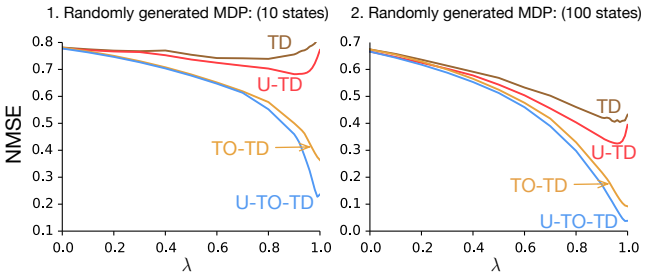

Figure 2: Empirical comparison of the new $O(n)$ usage-based algorithms with two existing $O(n)$ TD algorithms on on-policy policy-evaluation tasks. Performance is measured in empirical normalized MSE (NMSE).

We used randomly generated MDPs to produce two on-policy policy-evaluation tasks. As the TD algorithms here estimate state-value functions, it sufficed to construct Markov Reward Processes (MRPs), which we obtained by choosing the number of actions m to be 1 in both tasks. Our first task used an MDP with 10 states: $(10, 1, 3, 0.99\mathbf{I})$ and the second task used an MDP with 100 states: $(100, 1, 10, 0.99\mathbf{I})$. The feature vectors were binary representations of the state indices as in the off-policy tasks and were normalized to have unit length.

For each task, performance of each algorithm was measured for different values of their parameters. For TD(λ) and TO-TD(λ), the scalar step-size parameter α was varied by powers of 10 with powers chosen from -3 to 1 in steps of 0.25. For U-TD(λ) and U-TO-TD(λ), the parameter u_0 was varied by powers of 10 with powers chosen from -1 to 3 in steps of 0.25. The rest of the parameters for all four algorithms were varied using the same values as in the off-policy tasks. Performance was measured using NMSE as in the off-policy tasks. For each run, we averaged this error over 100 steps for the first task and 1000 steps for the second. The final estimate is produced again by averaging over 50 independent runs.

Figure 3 shows the performance on both tasks for different λ with rest of the parameters optimized. Left plot corresponds to MDP $(10, 1, 3, 0.99\mathbf{I})$ and the right plot corresponds to MDP $(100, 1, 10, 0.99\mathbf{I})$. On both tasks, the new algorithms performed significantly better than their base learning algorithms for higher values of λ and performed equally well for the smaller ones. The standard error in each case was smaller than the width of the curves shown. This set of experiments hints that the step-size adaptation based on the usage of features can be useful in both off-policy and on-policy tasks.

8 Discussion and Conclusions

Weighted importance sampling (WIS), one of the most effective variants of importance sampling, has long been neglected in off-policy learning with parametric function approximation. Recently introduced WIS-LSTD(λ) extends WIS to linear function approximation and eligibility traces but scales $O(n^3)$ in computational complexity in the number of features. In this paper, we took this endeavor one step further and carried over much of the benefits of WIS to $O(n)$ off-policy algorithms. In the process, we developed modifications of stochastic gradient descent that are more closely related to sample averages. This endeavor also resulted in developing a new online equivalence technique for deriving causal efficient updates from acausal intuitive updates, which was deemed essential for achieving $O(n)$ updates for our new algorithms. On three off-policy policy-evaluation experiments, the new algorithms outperformed the existing $O(n)$ off-policy algorithms and performed competitively with LSTD-TO(λ).

An intriguing outcome of our work is an adaptive vector step size that is updated based on the *usage* of features, which has emerged naturally from our goal to incorporate the sample average within SGD. It is distinct from the existing adaptive step-size algorithms but not the only possible one that can achieve a Monte Carlo equivalence. An interesting direction for future work would be to explore the other possible variants. Although beyond the scope of this work, it is interesting to investigate how the insights from the new step-size adaptation idea can be incorporated in existing step-size adaptation algorithms such as Autostep (Mahmood et al. 2012), vSGD (Schaul et al. 2013) or Adam (Kingma et al. 2014). Convergence analysis of the new algorithms is another interesting direction for future work.

Acknowledgements

The authors thank Hado van Hasselt, Joseph Modayil and Marlos C. Machado for fruitful discussions that helped improve the quality of this work. This work was supported by grants from Alberta Innovates – Technology Futures, the National Science and Engineering Research Council of Canada, and the Alberta Innovates Centre for Machine Learning.

References

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

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

Defazio, A., Graepel, T. (2014). A comparison of learning algorithms on the Arcade Learning Environment. *arXiv preprint arXiv:1410.8620*.

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

Hesterberg, T. C. (1988), *Advances in Importance Sampling*, Ph.D. Dissertation, Statistics Department, Stanford University.

Kahn, H., Marshall, A. W. (1953). Methods of reducing sample size in Monte Carlo computations. In *Journal of the Operations Research Society of America, 1*(5):263–278.

Kingma, D. P., Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv*:1412.6980.

Koller, D., Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

Liu, J. S. (2001). *Monte Carlo Strategies in Scientific Computing*. Berlin, Springer-Verlag.

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

Maei, H. R. (2011). *Gradient Temporal-Difference Learning Algorithms*. PhD thesis, University of Alberta.

Mahmood, A. R., Sutton, R. S., Degris, T., Pilarski, P. M. (2012). Tuning-free step-size adaptation. In *Proceedings of the 2012 IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 2121–2124.

Mahmood, A. R., van Hasselt, H., Sutton, R. S. (2014). Weighted importance sampling for off-policy learning with linear function approximation. In *Advances in Neural Information Processing Systems 27*, Montreal, Canada.

Prashanth, L. A., Korda, N., Munos, R. (2014). Fast LSTD using stochastic approximation: Finite time analysis and application to traffic control. In *Machine Learning and Knowledge Discovery in Databases* Springer, Lecture Notes in Computer Science, 8725:66–81.

Precup, D., Sutton, R. S., Singh, S. (2000). Eligibility traces for off-policy policy evaluation. In *Proceedings of the 17th International Conference on Machine Learning*, pp. 759–766. Morgan Kaufmann.

Robert, C. P., and Casella, G., (2004). *Monte Carlo Statistical Methods*, New York, Springer-Verlag.

Rubinstein, R. Y. (1981). *Simulation and the Monte Carlo Method*, New York, Wiley.

Schaul, T., Zhang, S., LeCun, Y. (2013). No more pesky learning rates. In *Proceedings of the 30th International Conference on Machine Learning*.

Shelton, C. R. (2001). *Importance Sampling for Reinforcement Learning with Multiple Objectives*. PhD thesis, Massachusetts Institute of Technology.

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

Sutton, R. S., Modayil, J., Delp, M., Degris, T., Pilarski, P. M., White, A., Precup, D. (2011). Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems*, pp. 761–768.

Sutton, R. S., Mahmood, A. R., Precup, D., van Hasselt, H. (2014). A new Q(λ) with interim forward view and Monte Carlo equivalence. In *Proceedings of the 31st International Conference on Machine Learning*, Beijing, China.

van Hasselt, H., Mahmood, A. R., Sutton, R. S. (2014). Off-policy TD(λ) with a true online equivalence. In *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence*, Quebec City, Canada.

van Seijen, H., & Sutton, R. S. (2014). True online TD(λ). In *Proceedings of the 31st International Conference on Machine Learning*. JMLR W&CP 32(1):692–700.

Appendix

A.1 Proof of Theorem 1

Theorem 1 (Backward consistency of U-SGD with sample average)**.** *If the feature representation is tabular, the vectors* $\mathbf{u}$ *and* $\boldsymbol{\theta}$ *are initially set to zero, and* $0 \leq \eta < 1$*, then U-SGD defined by (5)-(7) degenerates to the recency-weighted average estimator defined by (3) and (4), in the sense that each component of the parameter vector* $\boldsymbol{\theta}_{t+1}$ *of U-SGD becomes the recency-weighted average estimator of the corresponding input.*

Proof. Consider that t samples have been observed and among them t_x samples correspond to input x. Hence, $\sum_{x \in \mathcal{X}} t_x = t$. Let $Y_{x,k}$ denote the kth output corresponding to input x. Then the recency-weighted average estimator of $v(x)$ given overall data up to t can be equivalently redefined in the following way:

$$\begin{aligned}
\tilde{V}_{t_x+1} &\doteq \frac{\sum_{k=1}^{t_x}(1-\eta)^{t_x-k} Y_{x,k}}{\sum_{k=1}^{t_x}(1-\eta)^{t_x-k}} \\
&= \tilde{V}_{t_x} + \frac{1}{\tilde{U}_{t_x+1}}\left(Y_{x,t_x} - \tilde{V}_{t_x}\right); \quad \tilde{V}_1 = 0, \\
\tilde{U}_{t_x+1} &\doteq (1-\eta)\tilde{U}_{t_x} + 1; \qquad\qquad \tilde{U}_1 = 0.
\end{aligned}$$

Consider that the ith feature corresponds to input x. Then it is equivalent to prove that $[\boldsymbol{\theta}_{t+1}]_i = \tilde{V}_{t_x+1}$, where $[\cdot]_i$ denotes the ith component of a vector.

We prove by induction. First we show that $[\mathbf{u}_{t+1}]_i = \tilde{U}_{t_x+1}$. By assumption, $[\mathbf{u}_1]_i = \tilde{U}_1 = 0$. Now, consider that $[\mathbf{u}_t]_i = \tilde{U}_{(t-1)_x+1}$. Then the ith component of $\mathbf{u}_{t+1}$ can be written as

$$[\mathbf{u}_{t+1}]_i = (1 - \eta[\boldsymbol{\phi}_t]_i^2)[\mathbf{u}_t]_i + [\boldsymbol{\phi}_t]_i^2.$$

If the tth input is not x, then $t_x = (t-1)_x$ and $[\boldsymbol{\phi}_t]_i = 0$. Hence

$$[\mathbf{u}_{t+1}]_i = (1-0)\tilde{U}_{(t-1)_x+1} + 0 = \tilde{U}_{(t-1)_x+1} = \tilde{U}_{t_x+1}.$$

On the other hand, if the tth input is x, then $t_x = (t-1)_x + 1$ and $[\boldsymbol{\phi}_t]_i = 1$. Hence,

$$[\mathbf{u}_{t+1}]_i = (1-\eta)\tilde{U}_{(t-1)_x+1} + 1 = (1-\eta)\tilde{U}_{t_x} + 1 = \tilde{U}_{t_x+1}.$$

Hence, $[\boldsymbol{\alpha}_{t+1}]_i = \frac{1}{\tilde{U}_{t_x+1}}$, if $t_x > 0$, or $[\boldsymbol{\alpha}_{t+1}]_i = 0$, otherwise.

Now, by assumption, $[\boldsymbol{\theta}_1]_i = \tilde{V}_1 = 0$. Consider $[\boldsymbol{\theta}_t]_i = \tilde{V}_{(t-1)_x+1}$ and $t_x > 0$. Then the ith component of $\boldsymbol{\theta}_{t+1}$ can be written as

$$\begin{aligned}
[\boldsymbol{\theta}_{t+1}]_i &= [\boldsymbol{\theta}_t]_i + [\boldsymbol{\alpha}_{t+1}]_i \left(Y_t - \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_t\right)[\boldsymbol{\phi}_t]_i \\
&= \tilde{V}_{(t-1)_x+1} + \frac{1}{\tilde{U}_{t_x+1}}\left(Y_t - \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_t\right)[\boldsymbol{\phi}_t]_i.
\end{aligned}$$

If the tth input is not x, then $[\boldsymbol{\theta}_{t+1}]_i = \tilde{V}_{(t-1)_x+1} + 0 = \tilde{V}_{t_x+1}$.

On the other hand, if the tth input is x, then $Y_t = Y_{x,t_x}$ and

$$\begin{aligned}
[\boldsymbol{\theta}_{t+1}]_i &= \tilde{V}_{(t-1)_x+1} + \frac{1}{\tilde{U}_{t_x+1}}\left(Y_{x,t_x} - \tilde{V}_{(t-1)_x+1}\right) \\
&= \tilde{V}_{t_x} + \frac{1}{\tilde{U}_{t_x+1}}\left(Y_{x,t_x} - \tilde{V}_{t_x}\right) = \tilde{V}_{t_x+1}.
\end{aligned}$$

The only case that is left is when $t_x = 0$. In this case, the tth input cannot be x, and $\tilde{V}_{t_x+1} = \tilde{V}_{(t-1)_x+1} = \cdots = \tilde{V}_1 = 0$. Then

$$\begin{aligned}
[\boldsymbol{\theta}_{t+1}]_i &= [\boldsymbol{\theta}_t]_i + [\boldsymbol{\alpha}_{t+1}]_i \left(Y_t - \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_t\right)[\boldsymbol{\phi}_t]_i \\
&= \tilde{V}_{(t-1)_x+1} + 0 \cdot \left(Y_t - \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_t\right) \cdot 0 \\
&= 0 = \tilde{V}_{t_x+1}. \quad \square
\end{aligned}$$

A.2 Proof of Theorem 2

Theorem 2 (Backward consistency of WIS-SGD-1 with WIS)**.** *If the feature representation is tabular, the vectors* $\mathbf{u}$ *and* $\boldsymbol{\theta}$ *are initially set to zero, and* $0 \leq \eta < 1$*, then WIS-SGD-1 defined by (10)-(12) degenerates to recency-weighted WIS defined by (8) and (9) with* $Y_k \doteq G_k^{t+1}$ *and* $W_k \doteq \rho_k^{t+1}$*, in the sense that each component of the parameter vector of WIS-SGD-1* $\boldsymbol{\theta}_{t+1}^{t+1}$ *becomes the recency-weighted WIS estimator of the corresponding input.*

Proof. The proof is similar to that of Theorem 1.

Consider that data is available up to time $t+1$, among which state s was visited on t_s steps. Let $G_{s,k}^{t+1}$ denote the kth flat truncated return originated from state s and $\rho_{s,k}^{t+1}$ its corresponding importance-sampling ratio. Then the recency-weighted WIS estimator of $v(s)$ given overall data up to $t+1$ can be equivalently redefined in the following way:

$$\bar{V}_{t_s+1}^{t+1} \doteq \bar{V}_{t_s}^{t+1} + \frac{\rho_{s,t_s}^{t+1}}{\bar{U}_{t_s+1}^{t+1}} \left(G_{s,t_s}^{t+1} - \bar{V}_{t_s}^{t+1}\right); \quad \bar{V}_0^{t+1} = 0,$$
$$\bar{U}_{t_s+1}^{t+1} \doteq (1-\eta)\bar{U}_{t_s}^{t+1} + \rho_{s,t_s}^{t+1}; \qquad \bar{U}_0^{t+1} = 0.$$

Consider that the ith feature corresponds to input s. Then it is equivalent to prove that $\left[\boldsymbol{\theta}_{t+1}^{t+1}\right]_i = \bar{V}_{t_s+1}^{t+1}$, where $[\cdot]_i$ denotes the ith component of a vector. By abuse of notation, we drop all the $t+1$ from superscripts, as it is redundant in this proof.

We prove by induction. First we show that $[\mathbf{u}_{t+1}]_i = \bar{U}_{t_s+1}$. By assumption, $[\mathbf{u}_0]_i = \bar{U}_0 = 0$. Considering $[\mathbf{u}_t]_i = \bar{U}_{(t-1)_s+1}$. Then the ith component of $\mathbf{u}_{t+1}$ can be written as

$$[\mathbf{u}_{t+1}]_i = (1 - \eta[\boldsymbol{\phi}_t]_i^2)[\mathbf{u}_t]_i + \rho_t[\boldsymbol{\phi}_t]_i^2.$$

If the state at time t is not s, then $t_s = (t-1)_s$ and $[\boldsymbol{\phi}_t]_i = 0$. Hence

$$[\mathbf{u}_{t+1}]_i = (1-0)\bar{U}_{(t-1)_s+1} + 0 = \bar{U}_{(t-1)_s+1} = \bar{U}_{t_s+1}.$$

On the other hand, if the state at time t is s, then $t_s = (t-1)_s + 1$, $[\boldsymbol{\phi}_t]_i = 1$ and $\rho_t = \rho_{s,t_s}^{t+1}$. Hence,

$$\begin{aligned}[\mathbf{u}_{t+1}]_i &= (1-\eta)\bar{U}_{(t-1)_s+1} + \rho_{s,t_s}^{t+1} \\ &= (1-\eta)\bar{U}_{t_s} + \rho_{s,t_s}^{t+1} = \bar{U}_{t_s+1}.\end{aligned}$$

Hence, $[\boldsymbol{\alpha}_{t+1}]_i = \frac{1}{\bar{U}_{t_s+1}}$, if $t_s > 0$, or $[\boldsymbol{\alpha}_{t+1}]_i = 0$, otherwise.

Now, by assumption, $[\boldsymbol{\theta}_0]_i = \bar{V}_0 = 0$. Considering $[\boldsymbol{\theta}_t]_i = \bar{V}_{(t-1)_s+1}$ and $t_s > 0$, the ith component of $\boldsymbol{\theta}_{t+1}$ can be written as

$$\begin{aligned}[\boldsymbol{\theta}_{t+1}]_i &= [\boldsymbol{\theta}_t]_i + [\boldsymbol{\alpha}_{t+1}]_i \rho_t \left(G_t - \boldsymbol{\phi}_t^\top \boldsymbol{\theta}_t\right)[\boldsymbol{\phi}_t]_i \\ &= \bar{V}_{(t-1)_s+1} + \frac{\rho_t}{\bar{U}_{t_s+1}} \left(G_t - \boldsymbol{\phi}_t^\top \boldsymbol{\theta}_t\right)[\boldsymbol{\phi}_t]_i.\end{aligned}$$

If the state at time t is not s, then $[\boldsymbol{\theta}_{t+1}]_i = \bar{V}_{(t-1)_s+1} + 0 = \bar{V}_{t_s+1}$.

If the state at time t is not s, then $\rho_t = \rho_{s,t_s}$, $G_t = G_{s,t_s}$ and

$$\begin{aligned}[\boldsymbol{\theta}_{t+1}]_i &= \bar{V}_{(t-1)_s+1} + \frac{\rho_{s,t_s}}{\bar{U}_{t_s+1}} \left(G_{s,t_s} - \bar{V}_{(t-1)_s+1}\right) \\ &= \bar{V}_{t_s} + \frac{\rho_{s,t_s}}{\bar{U}_{t_s+1}} \left(G_{s,t_s} - \bar{V}_{t_s}\right) = \bar{V}_{t_s+1}.\end{aligned}$$

The only case that is left is when $t_s = 0$. In this case, the the state at time t cannot be s, and $\bar{V}_{t_s+1} = \bar{V}_{(t-1)_s+1} = \cdots = \bar{V}_0 = 0$. Then

$$\begin{aligned}[\boldsymbol{\theta}_{t+1}]_i &= [\boldsymbol{\theta}_t]_i + [\boldsymbol{\alpha}_{t+1}]_i \rho_t \left(G_t - \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_t\right)[\boldsymbol{\phi}_t]_i \\ &= \bar{V}_{(t-1)_s+1} + 0 \cdot \rho_t \left(G_t - \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_t\right) \cdot 0 \\ &= 0 = \bar{V}_{t_s+1}. \quad \square\end{aligned}$$

A.3 Proof of Theorem 3

Theorem 3 (Online equivalence technique)**.** *Consider any forward view that updates toward an interim scalar target Y_k^t with*

$$\boldsymbol{\theta}_{k+1}^{t+1} \doteq \mathbf{F}_k \boldsymbol{\theta}_k^{t+1} + Y_k^{t+1}\mathbf{w}_k + \mathbf{x}_k, \quad 0 \le k < t+1,$$

where $\boldsymbol{\theta}_0^t \doteq \boldsymbol{\theta}_0$ for some initial $\boldsymbol{\theta}_0$, and both $\mathbf{F}_k \in \mathbb{R}^{n\times n}$ and $\mathbf{w}_k \in \mathbb{R}^n$ can be computed using data available at k. Assume that the temporal difference $Y_k^{t+1} - Y_k^t$ at k is related to the temporal difference at $k+1$ as follows:

$$Y_k^{t+1} - Y_k^t = d_{k+1}\left(Y_{k+1}^{t+1} - Y_{k+1}^t\right) + b_t g_k \prod_{j=k+1}^{t-1} c_j, 0 \le k < t,$$

where b_k, c_k, d_k and g_k are scalars that can be computed using data available at time k. Then the final weight $\boldsymbol{\theta}_{t+1} \doteq \boldsymbol{\theta}_{t+1}^{t+1}$ can be computed through the following backward-view update, with $\mathbf{e}_{-1} \doteq \mathbf{0}$, $\mathbf{d}_0 \doteq \mathbf{0}$, and $t \ge 0$:

$$\begin{aligned}
\mathbf{e}_t &\doteq \mathbf{w}_t + d_t \mathbf{F}_t \mathbf{e}_{t-1}, \\
\boldsymbol{\theta}_{t+1} &\doteq \mathbf{F}_t \boldsymbol{\theta}_t + (Y_t^{t+1} - Y_t^t)\mathbf{e}_t + Y_t^t \mathbf{w}_t + b_t \mathbf{F}_t \mathbf{d}_t + \mathbf{x}_t, \\
\mathbf{d}_{t+1} &\doteq c_t \mathbf{F}_t \mathbf{d}_t + g_t \mathbf{e}_t.
\end{aligned}$$

Proof. We can write the difference between two consecutive estimates as

$$\begin{aligned}
\boldsymbol{\theta}_{t+1}^{t+1} - \boldsymbol{\theta}_t^t &= \mathbf{F}_t \boldsymbol{\theta}_t^{t+1} - \boldsymbol{\theta}_t^t + Y_t^{t+1}\mathbf{w}_k + \mathbf{x}_t \\
&= \mathbf{F}_t\left(\boldsymbol{\theta}_t^{t+1} - \boldsymbol{\theta}_t^t\right) + Y_t^{t+1}\mathbf{w}_k + (\mathbf{F}_t - \mathbf{I})\boldsymbol{\theta}_t^t + \mathbf{x}_t.
\end{aligned}$$

Now let us expand $\boldsymbol{\theta}_t^{t+1} - \boldsymbol{\theta}_t^t$:

$$\begin{aligned}
\boldsymbol{\theta}_t^{t+1} - \boldsymbol{\theta}_t^t &= \mathbf{F}_{t-1}\boldsymbol{\theta}_{t-1}^{t+1} + Y_{t-1}^{t+1}\mathbf{w}_{t-1} + \mathbf{x}_{t-1} \\
&\quad - \mathbf{F}_{t-1}\boldsymbol{\theta}_{t-1}^t - Y_{t-1}^t \mathbf{w}_{t-1} - \mathbf{x}_{t-1} \\
&= \mathbf{F}_{t-1}\left(\boldsymbol{\theta}_{t-1}^{t+1} - \boldsymbol{\theta}_{t-1}^t\right) + \left(Y_{t-1}^{t+1} - Y_{t-1}^t\right)\mathbf{w}_{t-1} \\
&= \mathbf{F}_{t-1}\cdots\mathbf{F}_0(\boldsymbol{\theta}_0^{t+1} - \boldsymbol{\theta}_0^t) + \sum_{k=0}^{t-1} \mathbf{F}_{t-1}\cdots\mathbf{F}_{k+1}(Y_k^{t+1} - Y_k^t)\mathbf{w}_k \\
&= \sum_{k=0}^{t-1} \mathbf{F}_{t-1}\cdots\mathbf{F}_{k+1}(Y_k^{t+1} - Y_k^t)\mathbf{w}_k \\
&= \sum_{k=0}^{t-1} \mathbf{F}_{t-1}\cdots\mathbf{F}_{k+1}\left(d_{k+1}(Y_{k+1}^{t+1} - Y_{k+1}^t) + b_t g_k \prod_{j=k+1}^{t-1} c_j\right)\mathbf{w}_k \\
&= \sum_{k=0}^{t-1} \mathbf{F}_{t-1}\cdots\mathbf{F}_{k+1}\Bigg(d_{k+1}\Bigg(d_{k+2}(Y_{k+2}^{t+1} - Y_{k+2}^t) \\
&\quad + b_t g_{k+1}\prod_{j=k+2}^{t-1} c_j\Bigg) + b_t g_k \prod_{j=k+1}^{t-1} c_j\Bigg)\mathbf{w}_k \\
&= \sum_{k=0}^{t-1} \mathbf{F}_{t-1}\cdots\mathbf{F}_{k+1}\Bigg(d_{k+1}d_{k+2}(Y_{k+2}^{t+1} - Y_{k+2}^t) \\
&\quad + b_t g_{k+1} d_{k+1}\prod_{j=k+2}^{t-1} c_j + b_t g_k \prod_{j=k+1}^{t-1} c_j\Bigg)\mathbf{w}_k \\
&= \sum_{k=0}^{t-1} \mathbf{F}_{t-1}\cdots\mathbf{F}_{k+1}\Bigg(\prod_{j=k+1}^{t} d_j (Y_t^{t+1} - Y_t^t)
\end{aligned}$$

$$
\begin{aligned}
&\quad + b_t \sum_{n=k}^{t-1} g_n \prod_{i=k+1}^{n} d_i \prod_{j=n+1}^{t-1} c_j \Bigg) \mathbf{w}_k \\
&= d_t (Y_t^{t+1} - Y_t^t) \underbrace{\sum_{k=0}^{t-1} \mathbf{F}_{t-1} \cdots \mathbf{F}_{k+1} \prod_{j=k+1}^{t-1} d_j \mathbf{w}_k}_{\mathbf{e}_{t-1}} \\
&\quad + b_t \underbrace{\sum_{k=0}^{t-1} \mathbf{F}_{t-1} \cdots \mathbf{F}_{k+1} \sum_{n=k}^{t-1} g_n \prod_{i=k+1}^{n} d_i \prod_{j=n+1}^{t-1} c_j \mathbf{w}_k}_{\mathbf{d}_t} \\
&= (Y_t^{t+1} - Y_t^t) d_t \mathbf{e}_{t-1} + b_t \mathbf{d}_t.
\end{aligned}
$$

The vectors $\mathbf{e}_t$ and $\mathbf{d}_t$ can be incrementally updated as follows:

$$
\begin{aligned}
\mathbf{e}_t &= \sum_{k=0}^{t} \mathbf{F}_t \cdots \mathbf{F}_{k+1} \prod_{j=k+1}^{t} d_j \mathbf{w}_k \\
&= \mathbf{w}_t + d_t \mathbf{F}_t \sum_{k=0}^{t-1} \mathbf{F}_{t-1} \cdots \mathbf{F}_{k+1} \prod_{j=k+1}^{t-1} d_j \mathbf{w}_k \\
&= \mathbf{w}_t + d_t \mathbf{F}_t \mathbf{e}_{t-1},
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{d}_t &= \sum_{k=0}^{t-1} \mathbf{F}_{t-1} \cdots \mathbf{F}_{k+1} \sum_{n=k}^{t-1} g_n \prod_{i=k+1}^{n} d_i \prod_{j=n+1}^{t-1} c_j \mathbf{w}_k \\
&= \sum_{k=0}^{t-1} \mathbf{F}_{t-1} \cdots \mathbf{F}_{k+1} \left(\sum_{n=k}^{t-2} g_n \prod_{i=k+1}^{n} d_i \prod_{j=n+1}^{t-1} c_j \mathbf{w}_k + g_{t-1} \prod_{j=k+1}^{t-1} d_j \mathbf{w}_k \right) \\
&= \sum_{k=0}^{t-1} \mathbf{F}_{t-1} \cdots \mathbf{F}_{k+1} \sum_{n=k}^{t-2} g_n \prod_{i=k+1}^{n} d_i \prod_{j=n+1}^{t-1} c_j \mathbf{w}_k + g_{t-1} \sum_{k=0}^{t-1} \mathbf{F}_{t-1} \cdots \mathbf{F}_{k+1} \prod_{j=k+1}^{t-1} d_j \mathbf{w}_k \\
&= c_{t-1} \mathbf{F}_{t-1} \sum_{k=0}^{t-2} \mathbf{F}_{t-1} \cdots \mathbf{F}_{k+1} \sum_{n=k}^{t-2} g_n \prod_{i=k+1}^{n} d_i \prod_{j=n+1}^{t-2} c_j \mathbf{w}_k + g_{t-1} \mathbf{e}_{t-1} \\
&= c_{t-1} \mathbf{F}_{t-1} \mathbf{d}_{t-1} + g_{t-1} \mathbf{e}_{t-1}.
\end{aligned}
$$

Then plugging back in

$$
\begin{aligned}
\boldsymbol{\theta}_{t+1}^{t+1} &= \boldsymbol{\theta}_t^t + \mathbf{F}_t \left(\boldsymbol{\theta}_t^{t+1} - \boldsymbol{\theta}_t^t \right) + Y_t^{t+1} \mathbf{w}_t + (\mathbf{F}_t - \mathbf{I}) \boldsymbol{\theta}_t^t + \mathbf{x}_t \\
&= \boldsymbol{\theta}_t^t + d_t \mathbf{F}_t \mathbf{e}_{t-1} (Y_t^{t+1} - Y_t^t) + b_t \mathbf{F}_t \mathbf{d}_t + Y_t^{t+1} \mathbf{w}_t + (\mathbf{F}_t - \mathbf{I}) \boldsymbol{\theta}_t^t + \mathbf{x}_t \\
&= \mathbf{F}_t \boldsymbol{\theta}_t^t + (\mathbf{e}_t - \mathbf{w}_t)(Y_t^{t+1} - Y_t^t) + Y_t^{t+1} \mathbf{w}_t + b_t \mathbf{F}_t \mathbf{d}_t + \mathbf{x}_t \\
&= \mathbf{F}_t \boldsymbol{\theta}_t^t + (Y_t^{t+1} - Y_t^t) \mathbf{e}_t + Y_t^t \mathbf{w}_t + b_t \mathbf{F}_t \mathbf{d}_t + \mathbf{x}_t. \quad \square
\end{aligned}
$$

A.4 Proof of Theorem 4

Theorem 4 (Generality of the new equivalence technique)**.** *The online equivalence technique by van Hasselt, Mahmood and Sutton (2014, Theorem 1) can be retrieved as a special case from the online equivalence technique given in Theorem 3.*

Proof. We describe the online equivalence technique by van Hasselt et al. (2014) in the following.

Consider any forward view that updates toward an interim scalar target Y_k^t with

$$\boldsymbol{\theta}_{k+1}^{t+1} = \boldsymbol{\theta}_k^{t+1} + \mu_k \left(Y_k^{t+1} - \boldsymbol{\phi}_k^\top \boldsymbol{\theta}_k^{t+1}\right) \boldsymbol{\phi}_k + \mathbf{x}_k, 0 \leq k < t,$$

where $\boldsymbol{\theta}_0^t = \boldsymbol{\theta}_0$ for some initial $\boldsymbol{\theta}_0$. Assume that the temporal difference $Y_k^{t+1} - Y_k^t$ at k is related to the temporal difference at $k+1$ as follows:

$$Y_k^{t+1} - Y_k^t = d_{k+1}(Y_{k+1}^{t+1} - Y_{k+1}^t), 0 \leq k < t,$$

where d_k is a scalar that can be computed using data available at time k. Then the final weight $\boldsymbol{\theta}_{t+1} \doteq \boldsymbol{\theta}_{t+1}^{t+1}$ can be computed through the following backward-view update, with $\mathbf{e}_{-1} = \mathbf{0}$ and $t \geq 0$:

$$\begin{aligned}
\mathbf{e}_t &= \mu_t \boldsymbol{\phi}_t + d_t(\mathbf{I} - \mu_t \boldsymbol{\phi}_t \boldsymbol{\phi}_t^\top)\mathbf{e}_{t-1}, \\
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + (Y_t^{t+1} - Y_t^t)\mathbf{e}_t + \mu_t(Y_t^t - \boldsymbol{\phi}_t^\top \boldsymbol{\theta}_t)\boldsymbol{\phi}_t + \mathbf{x}_t.
\end{aligned}$$

The above equivalence technique can be obtained from Theorem 3 as a special case by substituting $\mathbf{F}_k = \mathbf{I} - \mu_k \boldsymbol{\phi}_k \boldsymbol{\phi}_k^\top$, $\mathbf{w}_k = \mu_k \boldsymbol{\phi}_k$ and $b_k = 0$. □

A.5 Proof of Theorem 5

Theorem 5 (Backward view update for $\boldsymbol{\alpha}_t$ of WIS-TD(λ))**.** *The step-size vector $\boldsymbol{\alpha}_t$ computed by the following backward-view update and the forward-view update defined by (18) – (20) are equal at each step t:*

$$\mathbf{u}_{t+1} \doteq (\mathbf{1} - \eta \boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t) \circ \mathbf{u}_t + \rho_t \boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t + (\rho_t - 1)\gamma_t \lambda_t (\mathbf{1} - \eta \boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t) \circ \mathbf{v}_t, \tag{22}$$

$$\mathbf{v}_{t+1} \doteq \gamma_t \lambda_t \rho_t (\mathbf{1} - \eta \boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t) \circ \mathbf{v}_t + \rho_t \boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t, \tag{23}$$

$$\boldsymbol{\alpha}_{t+1} \doteq \mathbf{1} \oslash \mathbf{u}_{t+1}. \tag{24}$$

Proof. First, note that the component-wise vector multiplication in (19) can be written equivalently as a matrix-vector multiplication in the following way:

$$(\mathbf{1} - \eta \boldsymbol{\phi}_k \circ \boldsymbol{\phi}_k) \circ \mathbf{u}_k^{t+1} = (\mathbf{I} - \eta \text{Diag}\,(\boldsymbol{\phi}_k \circ \boldsymbol{\phi}_k))\, \mathbf{u}_k^{t+1},$$

where $\text{Diag}(\mathbf{v}) \in \mathbb{R}^{|\mathbf{v}| \times |\mathbf{v}|}$ is a diagonal matrix with the components of $\mathbf{v}$ in its diagonal.

In Theorem 3, we substitute $\boldsymbol{\theta}_k^{t+1} = \mathbf{u}_k^{t+1}$, $\mathbf{F}_k = (\mathbf{I} - \eta \text{Diag}\,(\boldsymbol{\phi}_k \circ \boldsymbol{\phi}_k))$, $\mathbf{x}_k = \mathbf{0}$, $\mathbf{w}_k = \boldsymbol{\phi}_k \circ \boldsymbol{\phi}_k$ and $Y_k^{t+1} = \tilde{\rho}_k^{t+1}$.

Now, $\tilde{\rho}_k^{t+1}$ can be recursively in t written as follows

$$\begin{aligned}
\tilde{\rho}_k^{t+1} &= \rho_k \sum_{i=k+1}^{t} C_k^{i-1}(1 - \gamma_i \lambda_i) + \rho_k C_k^t \\
&= \rho_k \sum_{i=k+1}^{t-1} C_k^{i-1}(1 - \gamma_i \lambda_i) + \rho_k C_k^{t-1}(1 - \gamma_t \lambda_t) + \rho_k C_k^t \\
&= \rho_k \sum_{i=k+1}^{t-1} C_k^{i-1}(1 - \gamma_i \lambda_i) + \rho_k C_k^{t-1} + \rho_k C_k^{t-1} \rho_t \gamma_t \lambda_t - \rho_k C_k^{t-1} \gamma_t \lambda_t \\
&= \tilde{\rho}_k^t + (\rho_t - 1)\gamma_t \lambda_t \rho_k C_k^{t-1}.
\end{aligned}$$

Hence, it proves that

$$Y_k^{t+1} - Y_k^t = d_{k+1}\left(Y_{k+1}^{t+1} - Y_{k+1}^t\right) + b_t g_k \prod_{j=k+1}^{t-1} c_j, 0 \leq k < t,$$

with $d_i = 0$, $b_i = (\rho_i - 1)\gamma_i\lambda_i$, $g_i = \rho_i$ and $c_i = \gamma_i\lambda_i\rho_i, \forall i$.

Inserting these substitutes in Theorem 3 yields us the backward-view defined by (22) – (24). □

A.6 Proof of Theorem 6

Theorem 6 (Backward view update for $\boldsymbol{\theta}_t^t$ of WIS-TD(λ))**.** *The parameter vector $\boldsymbol{\theta}_t$ computed by the following backward-view update and the parameter vector $\boldsymbol{\theta}_t^t$ computed by the forward-view update defined by (17) and (21) are equal at every time step t:*

$$\mathbf{e}_t \doteq \rho_t \boldsymbol{\alpha}_{t+1} \circ \boldsymbol{\phi}_t + \gamma_t\lambda_t\rho_t\left(\mathbf{e}_{t-1} - \rho_t\left(\boldsymbol{\alpha}_{t+1} \circ \boldsymbol{\phi}_t\right)\boldsymbol{\phi}_t^\top \mathbf{e}_{t-1}\right), \tag{25}$$

$$\begin{aligned}\boldsymbol{\theta}_{t+1} \doteq \boldsymbol{\theta}_t &+ \boldsymbol{\alpha}_{t+1} \circ \rho_t\left(\boldsymbol{\theta}_{t-1}^\top \boldsymbol{\phi}_t - \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_t\right)\boldsymbol{\phi}_t + (R_{t+1} + \gamma_{t+1}\boldsymbol{\theta}_t^\top \boldsymbol{\phi}_{t+1} - \boldsymbol{\theta}_{t-1}^\top \boldsymbol{\phi}_t)\mathbf{e}_t \\ &+ (\rho_t - 1)\gamma_t\lambda_t\left(\mathbf{d}_t - \rho_t\left(\boldsymbol{\alpha}_{t+1} \circ \boldsymbol{\phi}_t\right)\boldsymbol{\phi}_t^\top \mathbf{d}_t\right),\end{aligned} \tag{26}$$

$$\mathbf{d}_{t+1} \doteq \gamma_t\lambda_t\rho_t\left(\mathbf{d}_t - \rho_t\left(\boldsymbol{\alpha}_{t+1} \circ \boldsymbol{\phi}_t\right)\boldsymbol{\phi}_t^\top \mathbf{d}_t\right) + \left(R_{t+1} + \boldsymbol{\theta}_t^\top \boldsymbol{\phi}_{t+1} - \boldsymbol{\theta}_{t-1}^\top \boldsymbol{\phi}_t\right)\mathbf{e}_t. \tag{27}$$

Proof. First, we redefine (21) for convenience:

$$\boldsymbol{\theta}_{k+1}^{t+1} \doteq \boldsymbol{\theta}_k^{t+1} + \boldsymbol{\alpha}_{k+1} \circ \rho_k\left(\zeta_{k,t+1}^\rho - \boldsymbol{\phi}_k^\top \boldsymbol{\theta}_k^{t+1}\right)\boldsymbol{\phi}_k, \tag{28}$$

where $G_{k,t+1}^\rho = \rho_k \zeta_{k,t+1}^\rho$. Hence, $\zeta_{k,t+1}^\rho$ can be given by:

$$\begin{aligned}\zeta_{k,t+1}^\rho \doteq C_k^t\Big((1-\gamma_{t+1})G_k^{t+1} + \gamma_{t+1}\left(G_k^{t+1} + \boldsymbol{\phi}_{t+1}^\top \boldsymbol{\theta}_t\right)\Big) + \sum_{i=k+1}^{t} C_k^{i-1}\Big((1-\gamma_i)G_k^i + \gamma_i(1-\lambda_i)\left(G_k^i + \boldsymbol{\phi}_i^\top \boldsymbol{\theta}_{i-1}\right)\Big) \\ - \left(C_k^t + \sum_{i=k+1}^{t} C_k^{i-1}(1-\gamma_i\lambda_i) - 1\right)\boldsymbol{\phi}_k^\top \boldsymbol{\theta}_{k-1}.\end{aligned}$$

In Theorem 3, we substitute $\mathbf{F}_k = \mathbf{I} - \rho_k(\boldsymbol{\alpha}_{k+1} \circ \boldsymbol{\phi}_k)\boldsymbol{\phi}_k^\top$, $\mathbf{w}_k = \rho_k \boldsymbol{\alpha}_{k+1} \circ \boldsymbol{\phi}_k$, $Y_k^{t+1} = \zeta_{k,t+1}^\rho$ and $\mathbf{x}_k = 0, \forall k$, to get (28). Now, the next step is to establish a recursive relation for ζ^ρ both in k and t. For that, we use the following identities:

$$\begin{aligned}G_k^{k+1} &= R_{k+1}, \\ G_k^{t+1} &= \sum_{i=k}^{t} R_{i+1} = R_{k+1} + G_{k+1}^{t+1}.\end{aligned}$$

First we establish the recurrence relation in k:

$$\begin{aligned}\zeta_{k,t+1}^\rho &= C_k^t\Big((1-\gamma_{t+1})G_k^{t+1} + \gamma_{t+1}\left(G_k^{t+1} + \boldsymbol{\phi}_{t+1}^\top \boldsymbol{\theta}_t\right)\Big) + \sum_{i=k+1}^{t} C_k^{i-1}\Big((1-\gamma_i)G_k^i + \gamma_i(1-\lambda_i)\left(G_k^i + \boldsymbol{\phi}_i^\top \boldsymbol{\theta}_{i-1}\right)\Big) \\ &\quad - \left(C_k^t + \sum_{i=k+1}^{t} C_k^{i-1}(1-\gamma_i\lambda_i) - 1\right)\boldsymbol{\phi}_k^\top \boldsymbol{\theta}_{k-1} \\ &= C_k^t\Big((1-\gamma_{t+1})\left(R_{k+1} + G_{k+1}^{t+1}\right) + \gamma_{t+1}\left(R_{k+1} + G_{k+1}^{t+1} + \boldsymbol{\phi}_{t+1}^\top \boldsymbol{\theta}_t\right)\Big) \\ &\quad + \Big((1-\gamma_{k+1})G_k^{k+1} + \gamma_{k+1}(1-\lambda_{k+1})\left(G_k^{k+1} + \boldsymbol{\phi}_{k+1}^\top \boldsymbol{\theta}_k\right)\Big) \\ &\quad + \sum_{i=k+2}^{t} C_k^{i-1}\Big((1-\gamma_i)\left(R_{k+1} + G_{k+1}^i\right) + \gamma_i(1-\lambda_i)\left(R_{k+1} + G_{k+1}^i + \boldsymbol{\phi}_i^\top \boldsymbol{\theta}_{i-1}\right)\Big)\end{aligned}$$

$$
\begin{aligned}
&- \left(C_k^t + \sum_{i=k+1}^{t} C_k^{i-1}(1-\gamma_i\lambda_i) - 1\right)\boldsymbol{\phi}_k^\top\boldsymbol{\theta}_{k-1} \\
&= \rho_{k+1}\gamma_{k+1}\lambda_{k+1}C_{k+1}^t\left((1-\gamma_{t+1})G_{k+1}^{t+1} + \gamma_{t+1}(G_{k+1}^{t+1} + \boldsymbol{\phi}_{t+1}^\top\boldsymbol{\theta}_t)\right) \\
&\quad + \rho_{k+1}\gamma_{k+1}\lambda_{k+1}\sum_{i=k+2}^{t} C_{k+1}^{i-1}\Big((1-\gamma_i)G_{k+1}^i + \gamma_i(1-\lambda_i)\left(G_{k+1}^i + \boldsymbol{\phi}_i^\top\boldsymbol{\theta}_{i-1}\right)\Big) \\
&\quad - \rho_{k+1}\gamma_{k+1}\lambda_{k+1}\left(C_{k+1}^t + \sum_{i=k+2}^{t} C_{k+1}^{i-1}(1-\gamma_i\lambda_i) - 1\right)\boldsymbol{\phi}_{k+1}^\top\boldsymbol{\theta}_k \\
&\quad + \left(C_k^t + \sum_{i=k+2}^{t} C_k^{i-1}(1-\gamma_i\lambda_i) - \rho_{k+1}\gamma_{k+1}\lambda_{k+1}\right)\boldsymbol{\phi}_{k+1}^\top\boldsymbol{\theta}_k \\
&\quad + C_k^t R_{k+1} + (1-\gamma_{k+1}\lambda_{k+1})R_{k+1} + \gamma_{k+1}(1-\lambda_{k+1})\boldsymbol{\phi}_{k+1}^\top\boldsymbol{\theta}_k \\
&\quad + R_{k+1}\sum_{i=k+2}^{t} C_k^{i-1}(1-\gamma_i\lambda_i) \\
&\quad - \left(C_k^t + \sum_{i=k+1}^{t} C_k^{i-1}(1-\gamma_i\lambda_i) - 1\right)\boldsymbol{\phi}_k^\top\boldsymbol{\theta}_{k-1} \\
&= \rho_{k+1}\gamma_{k+1}\lambda_{k+1}\zeta_{k+1,t+1}^\rho + \left(C_k^t + \sum_{i=k+1}^{t} C_k^{i-1}(1-\gamma_i\lambda_i) - 1\right)\left(R_{k+1} + \boldsymbol{\phi}_{k+1}^\top\boldsymbol{\theta}_k - \boldsymbol{\phi}_k^\top\boldsymbol{\theta}_{k-1}\right) \\
&\quad + R_{k+1} + \boldsymbol{\phi}_{k+1}^\top\boldsymbol{\theta}_k - \rho_{k+1}\gamma_{k+1}\lambda_{k+1}\boldsymbol{\phi}_{k+1}^\top\boldsymbol{\theta}_k + \gamma_{k+1}(1-\lambda_{k+1})\boldsymbol{\phi}_{k+1}^\top\boldsymbol{\theta}_k - (1-\gamma_{k+1}\lambda_{k+1})\boldsymbol{\phi}_{k+1}^\top\boldsymbol{\theta}_k \\
&= \rho_{k+1}\gamma_{k+1}\lambda_{k+1}\zeta_{k+1,t+1}^\rho + \left(C_k^t + \sum_{i=k+1}^{t} C_k^{i-1}(1-\gamma_i\lambda_i) - 1\right)\left(R_{k+1} + \boldsymbol{\phi}_{k+1}^\top\boldsymbol{\theta}_k - \boldsymbol{\phi}_k^\top\boldsymbol{\theta}_{k-1}\right) \\
&\quad + R_{k+1} + \gamma_{k+1}\left(1-\rho_{k+1}\lambda_{k+1}\right)\boldsymbol{\phi}_{k+1}^\top\boldsymbol{\theta}_k.
\end{aligned}
$$

Then the recurrence in t can be established by subtracting $\zeta_{k,t}^\rho$ from $\zeta_{k,t+1}^\rho$:

$$
\begin{aligned}
\zeta_{k,t+1}^\rho - \zeta_{k,t}^\rho &\doteq \rho_{k+1}\gamma_{k+1}\lambda_{k+1}\zeta_{k+1,t+1}^\rho + \left(C_k^t + \sum_{i=k+1}^{t} C_k^{i-1}(1-\gamma_i\lambda_i) - 1\right)\left(R_{k+1} + \boldsymbol{\phi}_{k+1}^\top\boldsymbol{\theta}_k - \boldsymbol{\phi}_k^\top\boldsymbol{\theta}_{k-1}\right) \\
&\quad + R_{k+1} + \gamma_{k+1}\left(1-\rho_{k+1}\lambda_{k+1}\right)\boldsymbol{\phi}_{k+1}^\top\boldsymbol{\theta}_k \\
&\quad - \rho_{k+1}\gamma_{k+1}\lambda_{k+1}\zeta_{k+1,t}^\rho - \left(C_k^{t-1} + \sum_{i=k+1}^{t-1} C_k^{i-1}(1-\gamma_i\lambda_i) - 1\right)\left(R_{k+1} + \boldsymbol{\phi}_{k+1}^\top\boldsymbol{\theta}_k - \boldsymbol{\phi}_k^\top\boldsymbol{\theta}_{k-1}\right) \\
&\quad - R_{k+1} + \gamma_{k+1}\left(1-\rho_{k+1}\lambda_{k+1}\right)\boldsymbol{\phi}_{k+1}^\top\boldsymbol{\theta}_k \\
&= \rho_{k+1}\gamma_{k+1}\lambda_{k+1}\left(\zeta_{k+1,t+1}^\rho - \zeta_{k+1,t}^\rho\right) \\
&\quad + \left(C_k^t - C_k^{t-1} + C_k^{t-1}(1-\gamma_t\lambda_t)\right)\left(R_{k+1} + \boldsymbol{\phi}_{k+1}^\top\boldsymbol{\theta}_k - \boldsymbol{\phi}_k^\top\boldsymbol{\theta}_{k-1}\right) \\
&= \rho_{k+1}\gamma_{k+1}\lambda_{k+1}\left(\zeta_{k+1,t+1}^\rho - \zeta_{k+1,t}^\rho\right) + (\rho_t - 1)\gamma_t\lambda_t C_k^{t-1}\left(R_{k+1} + \boldsymbol{\phi}_{k+1}^\top\boldsymbol{\theta}_k - \boldsymbol{\phi}_k^\top\boldsymbol{\theta}_{k-1}\right).
\end{aligned}
$$

The above recurrence relation establishes

$$
Y_k^{t+1} - Y_k^t = d_{k+1}\left(Y_{k+1}^{t+1} - Y_{k+1}^t\right) + b_t g_k \prod_{j=k+1}^{t-1} c_j, 0 \le k < t,
$$

with $d_i = \rho_i\gamma_i\lambda_i$, $b_i = (\rho_i - 1)\gamma_i\lambda_i$, $g_i = R_{i+1} + \boldsymbol{\phi}_{i+1}^\top\boldsymbol{\theta}_i - \boldsymbol{\phi}_i^\top\boldsymbol{\theta}_{i-1}$ and $c_i = \gamma_i\lambda_i\rho_i, \forall i$. Inserting these substitutes in Theorem 3 yields us the backward-view defined by (25) – (27). □

A.7 Description of WIS-TD(λ), WIS-GTD(λ), WIS-TO-GTD(λ), U-TD(λ) and U-TO-TD(λ)

Algorithm 1 WIS-TD(λ)

Initialization:
Choose $\boldsymbol{\theta}_0, u_0 \geq 0, \eta \geq 0$
Set $\mathbf{u}_0 = u_0\mathbf{1}, \mathbf{v}_0 = \mathbf{0}, \mathbf{e}_{-1} = \mathbf{0}, \mathbf{d}_0 = \mathbf{0}$
for $t = 0, 1, \cdots$ **do**
receive $\boldsymbol{\phi}_t, \rho_t, \gamma_t, \lambda_t, R_{t+1}, \boldsymbol{\phi}_{t+1}, \gamma_{t+1}, \lambda_{t+1}$
$$\begin{aligned}
\mathbf{u}_{t+1} &= (\mathbf{1} - \eta\boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t) \circ \mathbf{u}_t + \rho_t\boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t \\
&\quad +(\rho_t - 1)\gamma_t\lambda_t (\mathbf{1} - \eta\boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t) \circ \mathbf{v}_t \\
\mathbf{v}_{t+1} &= \gamma_t\lambda_t\rho_t (\mathbf{1} - \eta\boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t) \circ \mathbf{v}_t + \rho_t\boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t \\
\boldsymbol{\alpha}_{t+1} &= \mathbf{1} \oslash \mathbf{u}_{t+1} \\
\mathbf{e}_t &= \rho_t\boldsymbol{\alpha}_{t+1} \circ \boldsymbol{\phi}_t \\
&\quad +\gamma_t\lambda_t\rho_t \left(\mathbf{e}_{t-1} - \rho_t (\boldsymbol{\alpha}_{t+1} \circ \boldsymbol{\phi}_t) \boldsymbol{\phi}_t^\top \mathbf{e}_{t-1}\right) \\
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \boldsymbol{\alpha}_{t+1} \circ \rho_t \left(\boldsymbol{\theta}_{t-1}^\top\boldsymbol{\phi}_t - \boldsymbol{\theta}_t^\top\boldsymbol{\phi}_t\right) \boldsymbol{\phi}_t \\
&\quad +(R_{t+1} + \gamma_{t+1}\boldsymbol{\theta}_t^\top\boldsymbol{\phi}_{t+1} - \boldsymbol{\theta}_{t-1}^\top\boldsymbol{\phi}_t)\mathbf{e}_t \\
&\quad +(\rho_t - 1)\gamma_t\lambda_t \left(\mathbf{d}_t - \rho_t (\boldsymbol{\alpha}_{t+1} \circ \boldsymbol{\phi}_t) \boldsymbol{\phi}_t^\top \mathbf{d}_t\right) \\
\mathbf{d}_{t+1} &= \gamma_t\lambda_t\rho_t \left(\mathbf{d}_t - \rho_t (\boldsymbol{\alpha}_{t+1} \circ \boldsymbol{\phi}_t) \boldsymbol{\phi}_t^\top \mathbf{d}_t\right) \\
&\quad + \left(R_{t+1} + \boldsymbol{\theta}_t^\top\boldsymbol{\phi}_{t+1} - \boldsymbol{\theta}_{t-1}^\top\boldsymbol{\phi}_t\right) \mathbf{e}_t
\end{aligned}$$
end for

Algorithm 2 WIS-GTD(λ)

Initialization:
Choose $\boldsymbol{\theta}_0, \mathbf{w}_0, u_0 \geq 0, \eta \geq 0, \beta \geq 0$
Set $\mathbf{u}_0 = u_0\mathbf{1}, \mathbf{v}_0 = \mathbf{0}, \mathbf{e}_{-1} = \mathbf{0}$
for $t = 0, 1, \cdots$ **do**
receive $\boldsymbol{\phi}_t, \rho_t, \gamma_t, \lambda_t, R_{t+1}, \boldsymbol{\phi}_{t+1}, \gamma_{t+1}, \lambda_{t+1}$
$$\begin{aligned}
\mathbf{u}_{t+1} &= (\mathbf{1} - \eta\boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t) \circ \mathbf{u}_t + \rho_t\boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t \\
&\quad +(\rho_t - 1)\gamma_t\lambda_t (\mathbf{1} - \eta\boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t) \circ \mathbf{v}_t \\
\mathbf{v}_{t+1} &= \gamma_t\lambda_t\rho_t (\mathbf{1} - \eta\boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t) \circ \mathbf{v}_t + \rho_t\boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t \\
\boldsymbol{\alpha}_{t+1} &= \mathbf{1} \oslash \mathbf{u}_{t+1} \\
\mathbf{e}_t &= \rho_t (\gamma_t\lambda_t\mathbf{e}_{t-1} + \boldsymbol{\phi}_t) \\
\delta_t &= R_{t+1} + \gamma_{t+1}\boldsymbol{\theta}_t^\top\boldsymbol{\phi}_{t+1} - \boldsymbol{\theta}_t^\top\boldsymbol{\phi}_t \\
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \boldsymbol{\alpha}_{t+1} \circ \delta_t\mathbf{e}_t \\
&\quad -\boldsymbol{\alpha}_{t+1} \circ \gamma_{t+1}(1 - \lambda_{t+1})(\mathbf{e}_t^\top\mathbf{w}_t)\boldsymbol{\phi}_{t+1} \\
\mathbf{w}_{t+1} &= \mathbf{w}_t + \beta \left[\delta_t\mathbf{e}_t - (\mathbf{w}_t^\top\boldsymbol{\phi}_t)\boldsymbol{\phi}_t\right]
\end{aligned}$$
end for

Algorithm 3 WIS-TO-GTD(λ)

Initialization:
Choose $\boldsymbol{\theta}_0, \mathbf{w}_0, u_0 \geq 0, \eta \geq 0, \beta \geq 0$
Set $\mathbf{u}_0 = u_0\mathbf{1}, \mathbf{v}_0 = \mathbf{0}, \mathbf{e}_{-1} = \mathbf{e}_{-1}^\nabla = \mathbf{e}_{-1}^\mathbf{w} = \mathbf{0}, \rho' = 0$
for $t = 0, 1, \cdots$ **do**
receive $\boldsymbol{\phi}_t, \rho_t, \gamma_t, \lambda_t, R_{t+1}, \boldsymbol{\phi}_{t+1}, \gamma_{t+1}, \lambda_{t+1}$
$$\begin{aligned}
\mathbf{u}_{t+1} &= (\mathbf{1} - \eta\boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t) \circ \mathbf{u}_t + \rho_t\boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t \\
&\quad +(\rho_t - 1)\gamma_t\lambda_t (\mathbf{1} - \eta\boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t) \circ \mathbf{v}_t \\
\mathbf{v}_{t+1} &= \gamma_t\lambda_t\rho_t (\mathbf{1} - \eta\boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t) \circ \mathbf{v}_t + \rho_t\boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t \\
\boldsymbol{\alpha}_{t+1} &= \mathbf{1} \oslash \mathbf{u}_{t+1} \\
\mathbf{e}_t &= \rho_t\boldsymbol{\alpha}_{t+1} \circ \boldsymbol{\phi}_t \\
&\quad +\gamma_t\lambda_t\rho_t \left(\mathbf{e}_{t-1} - \rho_t (\boldsymbol{\alpha}_{t+1} \circ \boldsymbol{\phi}_t) \boldsymbol{\phi}_t^\top \mathbf{e}_{t-1}\right) \\
\mathbf{e}_t^\nabla &= \rho_t (\gamma_t\lambda_t\mathbf{e}_{t-1} + \boldsymbol{\phi}_t) \\
\mathbf{e}_t^\mathbf{w} &= \gamma_t\lambda_t\rho'\mathbf{e}_{t-1}^\mathbf{w} + \beta \left(1 - \gamma_t\lambda_t\rho'\boldsymbol{\phi}_t^\top \mathbf{e}_{t-1}^\mathbf{w}\right) \boldsymbol{\phi}_t \\
\delta_t &= R_{t+1} + \gamma_{t+1}\boldsymbol{\theta}_t^\top\boldsymbol{\phi}_{t+1} - \boldsymbol{\theta}_t^\top\boldsymbol{\phi}_t \\
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \delta_t\mathbf{e}_t + (\mathbf{e}_t - \boldsymbol{\alpha}_{t+1} \circ \rho_t\boldsymbol{\phi}_t)(\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-1})^\top\boldsymbol{\phi}_t \\
&\quad -\boldsymbol{\alpha}_{t+1} \circ \gamma_{t+1}(1 - \lambda_{t+1})(\mathbf{w}_t^\top\mathbf{e}_t^\nabla)\boldsymbol{\phi}_{t+1} \\
\mathbf{w}_{t+1} &= \mathbf{w}_t + \rho_t\delta_t\mathbf{e}_t^\mathbf{w} - \beta(\mathbf{w}_t^\top\boldsymbol{\phi}_t)\boldsymbol{\phi}_t \\
\rho' &= \rho_t
\end{aligned}$$
end for

Algorithm 4 U-TD(λ)

Initialization:
Choose $\boldsymbol{\theta}_0, u_0 \geq 0, \eta \geq 0$
Set $\mathbf{u}_0 = u_0\mathbf{1}, , \mathbf{e}_{-1} = \mathbf{0}$
for $t = 0, 1, \cdots$ **do**
receive $\boldsymbol{\phi}_t, \gamma_t, \lambda_t, R_{t+1}, \boldsymbol{\phi}_{t+1}, \gamma_{t+1}, \lambda_{t+1}$
$$\begin{aligned}
\mathbf{u}_{t+1} &= (\mathbf{1} - \eta\boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t) \circ \mathbf{u}_t + \boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t \\
\boldsymbol{\alpha}_{t+1} &= \mathbf{1} \oslash \mathbf{u}_{t+1} \\
\mathbf{e}_t &= \gamma_t\lambda_t\mathbf{e}_{t-1} + \boldsymbol{\phi}_t \\
\delta_t &= R_{t+1} + \gamma_{t+1}\boldsymbol{\theta}_t^\top\boldsymbol{\phi}_{t+1} - \boldsymbol{\theta}_t^\top\boldsymbol{\phi}_t \\
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \boldsymbol{\alpha}_{t+1} \circ \delta_t\mathbf{e}_t
\end{aligned}$$
end for

Algorithm 5 U-TO-TD(λ)

Initialization:
Choose $\boldsymbol{\theta}_0, u_0 \geq 0, \eta \geq 0$
Set $\mathbf{u}_0 = u_0\mathbf{1}, \mathbf{e}_{-1} = \mathbf{0}$
for $t = 0, 1, \cdots$ **do**
receive $\boldsymbol{\phi}_t, \gamma_t, \lambda_t, R_{t+1}, \boldsymbol{\phi}_{t+1}, \gamma_{t+1}, \lambda_{t+1}$
$$\begin{aligned}
\mathbf{u}_{t+1} &= (\mathbf{1} - \eta\boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t) \circ \mathbf{u}_t + \boldsymbol{\phi}_t \circ \boldsymbol{\phi}_t \\
\boldsymbol{\alpha}_{t+1} &= \mathbf{1} \oslash \mathbf{u}_{t+1} \\
\mathbf{e}_t &= \boldsymbol{\alpha}_{t+1} \circ \boldsymbol{\phi}_t + \gamma_t\lambda_t \left(\mathbf{e}_{t-1} - (\boldsymbol{\alpha}_{t+1} \circ \boldsymbol{\phi}_t) \boldsymbol{\phi}_t^\top \mathbf{e}_{t-1}\right) \\
\boldsymbol{\theta}_{t+1} &= \boldsymbol{\theta}_t + \boldsymbol{\alpha}_{t+1} \circ \left(\boldsymbol{\theta}_{t-1}^\top\boldsymbol{\phi}_t - \boldsymbol{\theta}_t^\top\boldsymbol{\phi}_t\right) \boldsymbol{\phi}_t \\
&\quad +(R_{t+1} + \gamma_{t+1}\boldsymbol{\theta}_t^\top\boldsymbol{\phi}_{t+1} - \boldsymbol{\theta}_{t-1}^\top\boldsymbol{\phi}_t)\mathbf{e}_t
\end{aligned}$$
end for