# Multi-step Off-policy Learning Without Importance Sampling Ratios

**A. Rupam Mahmood** **Huizhen Yu** **Richard S. Sutton**

*Reinforcement Learning and Artificial Intelligence Laboratory*
*Department of Computing Science, University of Alberta*
*Edmonton, AB T6G 2E8 Canada*

## Abstract

To estimate the value functions of policies from exploratory data, most model-free off-policy algorithms rely on importance sampling, where the use of importance sampling ratios often leads to estimates with severe variance. It is thus desirable to learn off-policy without using the ratios. However, such an algorithm does not exist for multi-step learning with function approximation. In this paper, we introduce the first such algorithm based on temporal-difference (TD) learning updates. We show that an explicit use of importance sampling ratios can be eliminated by varying the amount of bootstrapping in TD updates in an action-dependent manner. Our new algorithm achieves stability using a two-timescale gradient-based TD update. A prior algorithm based on lookup table representation called Tree Backup can also be retrieved using action-dependent bootstrapping, becoming a special case of our algorithm. In two challenging off-policy tasks, we demonstrate that our algorithm is stable, effectively avoids the large variance issue, and can perform substantially better than its state-of-the-art counterpart.

## 1. Introduction

Off-policy learning constitutes an important class of reinforcement learning problems, where the goal is to learn about a designated target policy while behaving according to a different policy. In recent years, off-policy learning has garnered a substantial amount of attention in policy evaluation tasks. When safety and data frugality is paramount, the ability to evaluate a policy without actually following it can be invaluable (cf. Thomas 2015). Learning a large number of off-policy predictions is also considered important for model learning, options learning (Sutton et al. 1999), scalable life-long learning (White, Modayil & Sutton 2012), and knowledge representation (White 2015).

A number of computationally scalable algorithms have been proposed that can learn off-policy without requiring to access or estimate the model of the environment and in that sense are *model-free* (Precup et al. 2000, 2001, Maei & Sutton 2010, Maei 2011, Yu 2012, Geist & Scherrer 2014, Dann et al. 2014, van Hasselt et al. 2014, Mahmood et al. 2014, Yu 2015, 2016, Hallak et al. 2015a, Mahmood et al. 2015, Sutton et al. 2016, White & White 2016a). A core component of these algorithms is a classical Monte Carlo technique called importance sampling (Hammersley & Handscomb 1964, Rubinstein 1981), where samples are scaled by the likelihood ratio of the two policies, also known as the *importance sampling ratio*, so that they appear to be drawn from the target policy. Although importance sampling plays a key role in correcting the discrepancy between the policies, the highly varying nature of importance sampling ratios often results in large estimation variance (Liu 2001, Koller & Friedman 2009, Dann et al. 2014, Mahmood & Sutton 2015, White & White 2016a). Some off-policy algorithms avoid importance sampling using a model of the environment or combine model-based estimation with importance sampling based estimation. Due to performing model estimation in addition to value function estimation, these algorithms tend to be computationally more complex (Dudík et al. 2011, Paduraru 2013, Hallak et al. 2015b, Li et al. 2015, Jiang & Li 2015, Thomas & Brunskill 2016).

Multi-step learning is one of the most important components of modern temporal-difference learning algorithms. Through multi-step learning, temporal-difference learning algorithms can vary smoothly across a large spectrum of algorithms, including both one-step and Monte Carlo methods. The influence of multi-step learning is the greatest when parametric function approximation is used. In this case, solutions produced by multi-step learning can be much superior to those produced by one-step learning (Tsitsiklis & Van Roy 1997). Unfortunately, the problem of large variance with importance sampling is also the most severe in multi-step learning (White 2015, Mahmood & Sutton 2015). Consequently, multi-step off-policy learning remains problematic and largely unfulfilled.

An obvious approach to solve the problem of multi-step off-policy learning would then be to develop an algorithm that avoids using importance sampling ratios. The absence of these ratios will presumably reduce the estimation variance, making long-term multi-step learning tenable. Only a few model-free algorithms have been proposed to learn off-policy without using importance sampling ratios (Precup et al. 2000, van Hasselt 2011, Harutyunyan et al. 2016). However, all these algorithms were introduced either for one-step learning (van Hasselt 2011) or for learning with lookup table representation (Precup et al. 2000, Harutyunyan et al. 2016), that is, without using parametric function approximation. Multi-step learning does not have a lasting influence on performance in this case.

Our key contribution is to develop an algorithmic technique based on modulating the amount to which the estimates of the subsequent states are used, a concept known as *bootstrapping*, in an action-dependent manner. It results in an action-dependent bootstrapping parameter, which is a generalization of the state-dependent bootstrapping parameter used in prior works (Maei & Sutton 2010, Sutton et al. 2014). For action-value estimation, we show that importance sampling ratios can be eliminated by varying the action-dependent bootstrapping parameter for different state-action pairs in a particular way. Using this technique, we introduce a new algorithm called $ABQ$ that can achieve much less estimation variance compared to the state-of-the-art off-policy algorithm. ABQ is the first to effectively achieve multi-step function approximation solutions for off-policy learning without explicitly using importance sampling ratios. A prior algorithm, Tree Backup (Precup et al. 2000), can be retrieved as a special case of our algorithm. Furthermore, we show that another off-policy algorithm, Retrace (Munos et al. 2016), can also be derived and extended to the case of function approximation with stability using the action-dependent bootstrapping technique.

## 2. Problem formulation and notations

In this section, we formulate the problem of multi-step off-policy learning with parametric function approximation and establish notations. Consider an agent in a dynamical environment with a finite state space $\mathcal{S}$ and action space $\mathcal{A}$. At each time $t = 0, 1, \ldots$, if the present state is $s \in \mathcal{S}$ and the agent takes action $a \in \mathcal{A}$, then the next state $S_{t+1}$ is $s' \in \mathcal{S}$ with probability $p(s'|s,a)$, and the agent receives a random reward $R_{t+1}$ with mean $r(s,a)$ and finite variance upon the state transition. A randomized stationary policy $\pi$ specifies the probability $\pi(a|s)$ of taking action $a$ at state $s$. Of our interest is a given policy $\pi$, referred to as the target policy, and the performance of the agent if it follows $\pi$. Specifically, our interest in this paper is to estimate the action-value function of $\pi$, defined as the expected sum of discounted rewards for any initial state-action pair $(s,a)$:

$$q_\pi(s,a) \overset{\text{def}}{=\!=} \mathrm{E}_\pi\left[\sum_{t=1}^{\infty} \gamma^{t-1} R_t \middle| S_0 = s, A_0 = a\right], \tag{1}$$

where $\gamma < 1$ is a discount factor, and $\mathrm{E}_\pi[\cdot]$ signifies that all subsequent actions follow policy $\pi$.

In off-policy learning, the goal is to estimate $q_\pi$ from the observations $\{(S_t, A_t, R_{t+1})\}_{t \geq 0}$ obtained by the agent while it follows a (stationary randomized) policy $\mu \neq \pi$. We call $\mu$ the behavior policy. (The case $\mu = \pi$ is the on-policy case.) We shall assume that $\mu$ induces an irreducible Markov chain on the state-action space, whose unique invariant distribution we denote by $d_\mu$.

Important to multi-step learning are multi-step Bellman equations satisfied by the action-value function $q_\pi$. We review here such equations for the well-known TD($\lambda$), where $\lambda \in [0,1]$ is the bootstrapping parameter. Let $\mathbf{P}_\pi$ be the transition probability matrix of the Markov chain on $\mathcal{S} \times \mathcal{A}$ induced by the target policy $\pi$, and let $\mathbf{r} \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|}$ be the vector of expected rewards for different state-action pairs: $[\mathbf{r}]_{sa} \overset{\text{def}}{=\!=} r(s,a)$.[1] For $\lambda \in [0,1]$, define the multi-step Bellman operator $T_\pi^{(\lambda)}$ by

$$T_\pi^{(\lambda)}\mathbf{q} \overset{\text{def}}{=\!=} (\mathbf{I} - \gamma\lambda\mathbf{P}_\pi)^{-1}[\mathbf{r} + \gamma(1-\lambda)\mathbf{P}_\pi\mathbf{q}]$$

for all $\mathbf{q} \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|}$, where $\mathbf{I}$ is the identity matrix. Then $q_\pi$ satisfies the multi-step Bellman equation $\mathbf{q}_\pi = T_\pi^{(\lambda)}\mathbf{q}_\pi$, where $\mathbf{q}_\pi$ stands for the action-value function in vector notation.

We approximate the action-value function as a linear function of some given features of state-action pairs: $q_\pi(s,a) \approx \mathbf{w}^\top \mathbf{x}(s,a)$, where $\mathbf{w} \in \mathbb{R}^n$ is the parameter vector to be estimated and $\mathbf{x}(s,a) \in \mathbb{R}^n$ is the feature vector for state $s$ and action $a$. In matrix notation we write this approximation as $\mathbf{q}_\pi \approx \mathbf{X}\mathbf{w}$, where $\mathbf{X} \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}| \times n}$ is the feature matrix with the rows being the feature vectors for different state-action pairs: $[\mathbf{X}]_{sa,:} = \mathbf{x}(s,a)^\top$.

The multi-step solution to the off-policy learning problem with function approximation can be found by solving the fixed point equation: $\mathbf{X}\mathbf{w} = \Pi_\mu T_\pi^{(\lambda)}\mathbf{X}\mathbf{w}$ (when it has a solution), where $\Pi_\mu \overset{\text{def}}{=\!=} \mathbf{X}(\mathbf{X}^\top\mathbf{D}_\mu\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{D}_\mu$ is the projection matrix with $\mathbf{D}_\mu \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}| \times |\mathcal{S}|\cdot|\mathcal{A}|}$ being

1. We use brackets with subscripts to denote elements of vectors and matrices. We use small letters for vectors and capital letters for matrices, both boldfaced.

a diagonal matrix with diagonal elements $d_\mu(s,a)$. The Mean Squared Projected Bellman Error (MSPBE) corresponding to this equation is given by:

$$J(\mathbf{w}) = \left\| \Pi_\mu T_\pi^{(\lambda)} \mathbf{X}\mathbf{w} - \mathbf{X}\mathbf{w} \right\|^2_{\mathbf{D}_\mu}, \tag{2}$$

where $\|\mathbf{q}\|^2_{\mathbf{D}_\mu} = \mathbf{q}^\top \mathbf{D}_\mu \mathbf{q}$, for any $\mathbf{q} \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|}$. The multi-step asymptotic TD solution associated with the fixed-point equation and the above MSPBE can be expressed as $\mathbf{w}_\infty \overset{\text{def}}{=} \mathbf{A}^{-1}\mathbf{b}$, when $\mathbf{A}$ is an invertible matrix, and $\mathbf{A}$ and $\mathbf{b}$ are given by

$$\mathbf{A} \overset{\text{def}}{=} \mathbf{X}^\top \mathbf{D}_\mu \left(\mathbf{I} - \gamma\lambda \mathbf{P}_\pi\right)^{-1} \left(\mathbf{I} - \gamma \mathbf{P}_\pi\right) \mathbf{X}, \tag{3}$$

$$\mathbf{b} \overset{\text{def}}{=} \mathbf{X}^\top \mathbf{D}_\mu \left(\mathbf{I} - \gamma\lambda \mathbf{P}_\pi\right)^{-1} \mathbf{r}. \tag{4}$$

## 3. The advantage of multi-step learning

Under the rubric of temporal-difference learning fall a broad spectrum of methods. On one end of the spectrum, we have one-step methods that fully bootstrap using estimates of the next state and use only the immediate rewards as samples. On the other end of the spectrum, we have Monte Carlo methods that do not bootstrap and rather use all future rewards for making updates. Many multi-step learning algorithms incorporate this full spectrum and can vary smoothly between one-step and Monte Carlo updates using the bootstrapping parameter $\lambda$. Here $1-\lambda$ determines the degree to which bootstrapping is used in the algorithm. With $\lambda = 0$, these algorithms achieve one-step TD updates, whereas with $\lambda = 1$, they effectively achieve Monte Carlo updates. To contrast with one-step learning, multi-step learning is generally viewed as learning with $\lambda > 0$ in TD methods.

Multi-step learning impacts the efficiency of estimation in two ways. First, it allows more efficient estimation compared to one-step learning with a finite amount of samples. One-step learning uses the minimal amount of samples, have relatively less variance compared to Monte Carlo updates, but produces biased estimates. Typically, with a finite amount of samples, a value of $\lambda$ between 0 and 1 reduces the estimation error the most (Sutton & Barto 1998).

Second, when function approximation is used and $q_\pi$ does not lie in the approximation subspace, multi-step learning can produce superior asymptotic solutions compared to one-step learning. As $\lambda$ increases, the multi-step Bellman operator approaches the constant operator that maps every $q$ to $q_\pi$. This in general leads to better approximations, as suggested by the monotonically improving error bound of asymptotic solutions (Tsitsiklis & Van Roy 1997) in the on-policy case, and as we demonstrate for the off-policy case in Figure 1.

Although multi-step learning is desirable with function approximation, it is more difficult in the off-policy case where the detrimental effect of importance sampling is most pronounced. For this reason, off-policy learning without importance sampling ratios is a naturally appealing and desirable solution to this problem. Prior works on off-policy learning without the ratios (e.g., Precup et al. 2000, Harutyunyan et al. 2016) are given in the lookup table case where the benefit of multi-step learning does not show up, because regardless of $\lambda$, the asymptotic solution is $q_\pi$. It is in the case of function approximation that multi-step off-policy learning without importance sampling ratios is most needed.

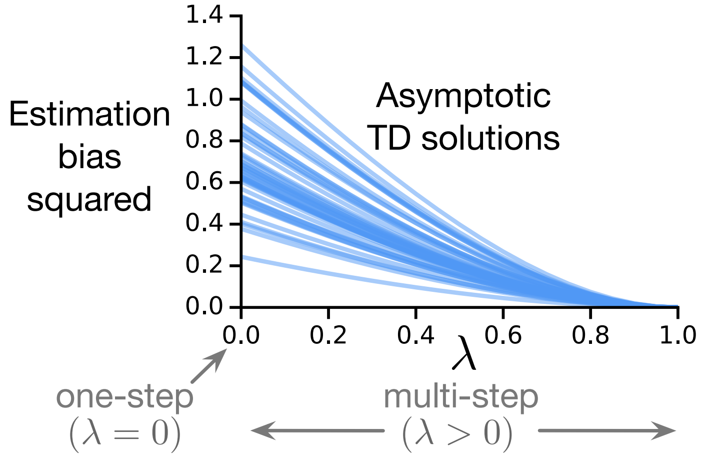

Figure 1: Multi-step solutions are generally superior to one-step solutions, as estimation bias goes to zero often monotonically with increasing $\lambda$, shown here for 50 randomly constructed MDPs. In these MDPs, we used 100 states, 5 actions, and 40 features. The rewards, probabilities, and feature values (binary) were chosen uniformly randomly.

## 4. Multi-step off-policy learning with importance sampling ratios

To set the stage for our work, we describe in this section the canonical multi-step off-policy learning update with importance sampling ratios, and how the ratios introduce variance in off-policy temporal-difference (TD) updates. A TD update is generally constructed based on stochastic approximation methods, where the target of the update is based on returns. Here we consider the off-line update for off-policy TD learning. Although not practical for implementation, off-line updates are useful and develop the foundation for deriving practical and computationally efficient algorithms (Sutton & Barto 1998, Seijen et al. 2016). An off-line TD update for off-policy action-value estimation based on multi-step returns can be defined as:

$$\Delta \mathbf{w}_t = \alpha_t \left( G_t^\lambda - \mathbf{w}^\top \mathbf{x}_t \right) \mathbf{x}_t, \tag{5}$$

where $\alpha > 0$ is the step-size parameter, and $\mathbf{w}$ is a fixed weight vector. Here, $G_t^\lambda$ is the multi-step target, known as $\lambda$-return, defined as the sum of TD errors weighted by powers of $\gamma\lambda$ and products of importance sampling ratios:

$$G_t^\lambda \overset{\text{def}}{=\!=} \sum_{n=t}^{\infty} (\gamma\lambda)^{n-t} \rho_{t+1}^n \delta_n + \mathbf{w}^\top \mathbf{x}_t. \tag{6}$$

The TD error $\delta_t$ is defined as $\delta_t \overset{\text{def}}{=\!=} R_{t+1} + \gamma \mathbf{w}^\top \bar{\mathbf{x}}_{t+1} - \mathbf{w}^\top \mathbf{x}_t$, with $\bar{\mathbf{x}}_t \overset{\text{def}}{=\!=} \sum_a \pi(a|S_t)\mathbf{x}(S_t, a)$. The term $\rho_t^n \overset{\text{def}}{=\!=} \Pi_{i=t}^n \rho_i$ is a product of importance sampling ratios $\rho_t \overset{\text{def}}{=\!=} \frac{\pi(A_t|S_t)}{\mu(A_t|S_t)}$, which accounts for the discrepancy due to using the behavior policy instead of the target policy throughout the trajectory. Note that, the update defined by (5) is a forward-view update, that is, it uses samples that only become available in the future from the time the state of the updated estimate is visited. We call this update *the off-policy* $Q(\lambda)$ *update*. It can be shown that the asymptotic multi-step solution corresponding to off-policy Q($\lambda$) is given by $\mathbf{w}_\infty = \mathbf{A}^{-1}\mathbf{b}$, (3), and (4), when $\mathbf{A}$ is invertible. All existing multi-step off-policy algorithms with importance sampling are of this form or a variant.

When $\lambda = 0$, no importance sampling ratios are involved, and this update reduces to that of off-policy expected Sarsa (Sutton & Barto 1998, van Hasselt 2011, Sutton et al. 2014). This one-step update is also closely related to the one-step Q-learning update, where a greedy nonstationary target policy is used instead of a fixed stationary one.

The importance sampling ratios play a role when $\lambda > 0$, and their influence, including the detrimental impact on variance, is greater with larger $\lambda$. The product $\rho_1^n$ of off-policy Q($\lambda$) in (6) can become as large as $\frac{1}{(\min_{s,a}\mu(a|s))^n}$. Such an exponential growth, when occurred even momentarily, can have large impact on the variance of the estimate. If the value of $\lambda$ is small or very close to zero, the large variance ensuing from the product may be avoided, but it would also be devoid of much of the benefits of multi-step learning.

## 5. Avoiding importance sampling ratios

In this section, we introduce the idea of action-dependent bootstrapping and how it can be used to avoid importance sampling ratios in off-policy estimates. For that, first we introduce an action-dependent bootstrapping parameter $\lambda(s,a) \in [0,1]$, which is allowed to vary between different state-action pairs. A closely related idea is state-dependent bootstrapping used by Sutton and Singh (1994) and Sutton et al. (2014) for state-value estimation, and by Maei and Sutton (2010) for action-value estimation. In those works, the degree of bootstrapping was allowed to vary from one state to another by the state-dependent bootstrapping parameter $\lambda(s) \in [0,1]$ but was not used as a device to reduce the estimation variance.

The variability of the parameter $\lambda(s,a)$ can be utilized algorithmically on a moment-by-moment basis to absorb the detrimental effect of importance sampling and in general to control the impact of importance sampling. Let us use the notational shorthand $\lambda_t \overset{\text{def}}{=} \lambda(S_t, A_t)$, and define a new $\lambda$-return by replacing the constant $\lambda$ in (6) with variable $\lambda(s,a)$:

$$G_t^\lambda = \sum_{n=t}^{\infty} \gamma^{n-t} \lambda_{t+1}^n \rho_{t+1}^n \delta_n + \mathbf{w}^\top \mathbf{x}_t, \tag{7}$$

where $\lambda_t^n \overset{\text{def}}{=} \Pi_{i=t}^n \lambda_i$. Notice that each importance sampling ratio in (7) is factored with a corresponding bootstrapping parameter: $\lambda_t \rho_t$. We can mitigate an explicit use of importance sampling ratios by setting the action-dependent bootstrapping parameter $\lambda(s,a)$ in the following way:

$$\lambda(s,a) = \nu(\psi, s, a)\mu(a|s), \qquad \nu(\psi, s, a) \overset{\text{def}}{=} \min\left(\psi, \frac{1}{\max\left(\mu(a|s), \pi(a|s)\right)}\right) \tag{8}$$

where $\psi \geq 0$ is a constant. Note that $\nu(\psi, s, a)$ is upper-bounded by $\psi_{\max}$, which is defined as follows:

$$\psi_{\max} \stackrel{\text{def}}{=\!=} \frac{1}{\min_{s,a} \max\left(\mu(a|s), \pi(a|s)\right)}. \tag{9}$$

The product $\lambda_t \rho_t$ can then be rewritten as: $\lambda_t \rho_t = \nu(\psi, S_t, A_t)\mu_t \frac{\pi_t}{\mu_t} = \nu(\psi, S_t, A_t)\pi_t$, dispelling an explicit presence of importance sampling ratios from the update. It is easy to see that, under our proposed scheme, the effective bootstrapping parameter is upper bounded by 1: $\lambda_t \leq 1$, and at the same time all the products are also upper bounded by one: $\lambda_t^n \rho_t^n \leq 1$, largely reducing variance.

To understand how $\psi$ influences $\lambda(s, a)$ let us use the following example, where there are only one state and three actions $\{1, 2, 3\}$ available. The behavior policy probabilities are $[0.2, 0.3, 0.5]$ and the target policy probabilities are $[0.2, 0.4, 0.4]$ for the three actions, respectively. Figure 2 (left) shows how the action-dependent bootstrapping parameter $\lambda$ for different actions change as $\psi$ is increased from 0 to $\psi_{\max}$. Initially, the bootstrapping parameter $\lambda$ increased linearly for all actions at a different rate depending on their corresponding behavior policy probabilities. The min in the factor $\nu$ comes into effect with $\psi > \psi_0 \stackrel{\text{def}}{=\!=} \frac{1}{\max_{s,a} \max(\mu(a|s), \pi(a|s))}$, and the largest $\lambda$ at $\psi = \psi_0$ gets capped first. Eventually, with large enough $\psi$, that is, $\psi \geq \psi_{\max}$, all $\lambda$s get capped.

Algorithms with constant $\lambda$ are typically studied in terms of their parameters by varying $\lambda$ between $[0, 1]$, which would not be possible for an algorithm based on the above scheme as $\lambda$ is not a constant any more. For our scheme, the constant tunable parameter is $\psi$, which has three pivotal values: $[0, \psi_0, \psi_{\max}]$. For parameter studies, it would be convenient if $\psi$ is scaled to another tunable parameter between $[0, 1]$. But in that case, we have to make a decision on what value $\psi_0$ should transform to. In the absence of a clear sense of it, a default choice would be to transform $\psi_0$ to 0.5. One such scaling is where we set $\psi$ as a function of another constant $\zeta \geq 0$ in the following way:

$$\psi(\zeta) = 2\zeta\psi_0 + (2\zeta - 1)^+ \times (\psi_{\max} - 2\psi_0), \tag{10}$$

and then vary $\zeta$ between $[0, 1]$ as one would vary $\lambda$. Here $(x)^+ = \max(0, x)$. In this case, $\zeta = 0, 0.5$, and 1 correspond to $\psi(\zeta) = 0, \psi_0$, and $\psi_{\max}$, respectively. Note that $\zeta$ can be written conversely in terms of $\psi$ as follows:

$$\zeta = (\psi - \psi_0)^+ \cdot \frac{2\psi_0 - \psi_{\max}}{2\left(\psi_{\max} - \psi_0\right)\psi_0} + \frac{\psi}{2\psi_0}. \tag{11}$$

The top margin of Figure 2 (left) shows an alternate x-axis in terms of $\zeta$.

To form an update using this proposed modification to $\lambda$, let us use the following notational shorthands,

$$\nu_\zeta(s, a) \stackrel{\text{def}}{=\!=} \nu(\psi(\zeta), s, a), \qquad \nu_{\zeta,t} \stackrel{\text{def}}{=\!=} \nu_\zeta(S_t, A_t), \tag{12}$$

$$\lambda_\zeta(s, a) \stackrel{\text{def}}{=\!=} \nu(\psi(\zeta), s, a)\mu(a|s), \qquad \lambda_{\zeta,t} \stackrel{\text{def}}{=\!=} \lambda_\zeta(S_t, A_t). \tag{13}$$

We use $H_t^\zeta$ to denote the $\lambda$-return defined by (7) with the bootstrapping parameter set according to $\lambda_\zeta$:

$$H_t^\zeta = \sum_{n=t}^{\infty} \gamma^{n-t} \lambda_{\zeta,t+1}^n \rho_{t+1}^n \delta_n + \mathbf{w}^\top \mathbf{x}_t = \sum_{n=t}^{\infty} \gamma^{n-t} \nu_{\zeta,t+1}^n \pi_{t+1}^n \delta_n + \mathbf{w}^\top \mathbf{x}_t, \tag{14}$$

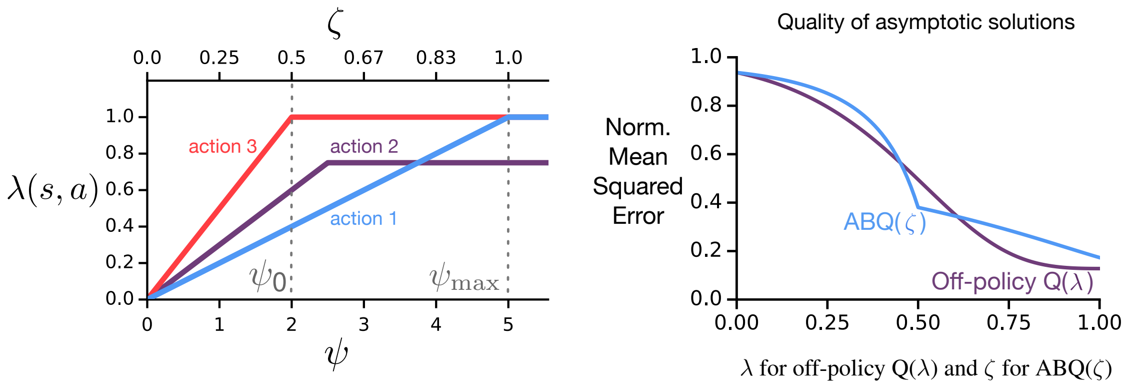

Figure 2: **Left**: The effect of $\psi$ and $\zeta$ on $\lambda(s,a)$ for three different actions under the action-dependent bootstrapping scheme. As $\psi$ is increased, the parameter $\lambda$ for different actions increase at a different rate and get capped for different values of $\psi$. **Right**: The effect of $\lambda$ on GQ($\lambda$) solutions and $\zeta$ on ABQ($\zeta$) solutions. Multi-step ($\lambda > 0$) off-policy Q($\lambda$) solutions are superior to the one-step ($\lambda = 0$) solution. ABQ($\zeta$) solutions can also achieve a similar multi-step advantage with $\zeta > 0$.

where $\lambda^n_{\zeta,t} \overset{\text{def}}{=\!=} \Pi^n_{i=t}\lambda_{\zeta,i}$, $\nu^n_{\zeta,t} \overset{\text{def}}{=\!=} \Pi^n_{i=t}\nu_{\zeta,i}$, and $\pi^n_t \overset{\text{def}}{=\!=} \Pi^n_{i=t}\pi_i$. The off-line forward-view update with $H^\zeta_t$ as the target can be written as:

$$\Delta\mathbf{w}_t = \alpha_t \left(H^\zeta_t - \mathbf{w}^\top\mathbf{x}_t\right)\mathbf{x}_t. \tag{15}$$

The asymptotic solution corresponding to this update, which we call the *ABQ*($\zeta$) solution, is $\mathbf{w}^\zeta_\infty \overset{\text{def}}{=\!=} \mathbf{A}_\zeta{}^{-1}\mathbf{b}_\zeta$ with

$$\mathbf{A}_\zeta \overset{\text{def}}{=\!=} \mathbf{X}^\top\mathbf{D}_\mu\left(\mathbf{I} - \gamma\mathbf{P}_\pi\mathbf{\Lambda}_\zeta\right)^{-1}\left(\mathbf{I} - \gamma\mathbf{P}_\pi\right)\mathbf{X}, \tag{16}$$

$$\mathbf{b}_\zeta \overset{\text{def}}{=\!=} \mathbf{X}^\top\mathbf{D}_\mu\left(\mathbf{I} - \gamma\mathbf{P}_\pi\mathbf{\Lambda}_\zeta\right)^{-1}\mathbf{r}, \tag{17}$$

assuming $\mathbf{A}_\zeta$ is invertible. The derivation is given in Appendix A.1. Here $\mathbf{\Lambda}_\zeta$ is a diagonal matrix with $\lambda_\zeta(s,a)$ being the diagonal elements. This is a multi-step solution when the bootstrapping parameter $\lambda_\zeta(s,a)$ does not uniformly reduce to zero. The drawback of this scheme is that we cannot achieve $\lambda_\zeta(s,a) = 1$ for all state-action pairs, which would produce the off-policy Monte Carlo solution. It is the cost of avoiding importance sampling ratios together with its large variance issue. However, much of the multi-step benefits can still be retained by choosing a large $\zeta$.

To illustrate that ABQ($\zeta$) solutions can retain much of the multi-step benefits, we used a two-state off-policy task similar to the off-policy task by Sutton et al. (2016). In this task, there were two states each with two actions, *left* and *right*, leading to one of the two states deterministically. More specifically, $p(1|1, \mathit{left}) = p(2|1, \mathit{right}) = p(1|2, \mathit{left}) = p(2|2, \mathit{right}) = 1$. There is a deterministic nonzero reward $r(2, \mathit{right}) = +1$; all other transitions have reward zero. The discount factor $\gamma$ was 0.9. The feature vectors were set as $\mathbf{x}(1, \mathit{left}) = \mathbf{x}(1, \mathit{right}) =$

1 and $\mathbf{x}(2, \textit{left}) = \mathbf{x}(2, \textit{right}) = 2$. The behavior policy was chosen as $\mu(\textit{right}|1) = \mu(\textit{left}|2) = 0.9$ to break away from the uniform distribution of the original problem. The target policy was chosen as $\pi(\textit{right}|\cdot) = 0.9$.

We produced different asymptotic solutions defined by (16) and (17), choosing different constant $\zeta$ between $[0, 1]$. We compared ABQ($\zeta$) solutions with off-policy Q($\lambda$) solutions in terms of the Mean Squared Error (MSE) $\|\mathbf{X}\mathbf{w} - \mathbf{q}_\pi\|^2_{\mathbf{D}_\mu}$ normalized by $\|\mathbf{q}_\pi\|^2_{\mathbf{D}_\mu}$. The results are given in Figure 2 (right). Off-policy Q($\lambda$) solutions with $\lambda > 0$ in this task are substantially better than the one-step solution produced with $\lambda = 0$. The ABQ($\zeta$) solutions cannot be as good as the off-policy Q(1) solution, as we already anticipated, but much of the benefits of multi-step off-policy solutions can be attained by choosing a large value of $\zeta$. Although the tunable parameter $\zeta$ of ABQ was set to be a constant, the effective bootstrapping parameter $\lambda_\zeta(s, a)$ was different for different state-action pairs. Therefore, ABQ solutions cannot be obtained simply by rescaling the constant $\lambda$ of off-policy Q($\lambda$).

The algorithm in principle can be used with either $\zeta$ or $\psi$ as the tunable parameter. If we tune with $\zeta \in [0, 1]$, we will have to first use (10) to obtain $\psi(\zeta)$, and then (12) and (13) to obtain $\lambda_t$ on a moment-by-moment basis. Computing $\psi(\zeta)$ from $\zeta$ requires knowing $\psi_0$ and $\psi_{\max}$ which are often known, for example, when the policies are in $\epsilon$-greedy form or fixed and known beforehand. In other cases, tuning with $\psi$ directly can be more convenient. As the scaled parameter $\zeta$ reflects more clearly the qualitative behavior of ABQ, we use it as the tunable parameter in this work.

## 6. The ABQ($\zeta$) algorithm with gradient correction and scalable updates

In this section, we develop a computationally scalable and stable algorithm corresponding to the ABQ($\zeta$) solution. The update (15) given earlier cannot be computed in a scalable manner, because forward-view updates require an increasing amount of computation as time progresses. Moreover, off-policy algorithms with bootstrapping and function approximation may be unstable (Sutton & Barto 1998), unless machinery for ensuring stability is incorporated. Our goal is to develop a stable update corresponding to ABQ($\zeta$) solutions while keeping it free from an explicit use of importance sampling ratios.

First, we produce the equivalent backward view of (15) so that the updates can be implemented without forming explicitly the $\lambda$-return. As we derive in Appendix A.2, these updates are given by

$$\Delta \mathbf{w}_t = \alpha_t \delta_t \mathbf{e}_t, \qquad \mathbf{e}_t = \gamma \nu_{\zeta,t} \pi_t \mathbf{e}_{t-1} + \mathbf{x}_t. \tag{18}$$

Here, $\mathbf{e}_t \in \mathbb{R}^n$ is an accumulating trace vector. The above backward-view update achieves equivalence with the forward-view of (15) only for off-line updating, which is typical for all algorithms with accumulating traces. Equivalence for online updating could also be achieved by following the approach taken by van Seijen et al. (2014, 2016), but we leave that out for simplicity.

We take the approach proposed by Maei (2011) to develop a stable gradient-based TD algorithm. The key step in deriving a gradient-based TD algorithm is to formulate an associated Mean Squared Projected Bellman Error (MSPBE) and produce its gradient, which can then be sampled to produce stochastic updates.

The MSPBE for the update (15) is given by:

$$J(\mathbf{w}) = \left\|\Pi_\mu T_\pi^{(\mathbf{\Lambda}_\zeta)}\mathbf{X}\mathbf{w} - \mathbf{X}\mathbf{w}\right\|^2_{\mathbf{D}_\mu} \tag{19}$$

$$= \left(\mathbf{X}^\top\mathbf{D}_\mu\left(T_\pi^{(\mathbf{\Lambda}_\zeta)}\mathbf{X}\mathbf{w} - \mathbf{X}\mathbf{w}\right)\right)^\top (\mathbf{X}^\top\mathbf{D}_\mu\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{D}_\mu\left(T_\pi^{(\mathbf{\Lambda}_\zeta)}\mathbf{X}\mathbf{w} - \mathbf{X}\mathbf{w}\right). \tag{20}$$

Here, the Bellman operator corresponding to the bootstrapping matrix $\mathbf{\Lambda}_\zeta$ is defined for all $\mathbf{q}\in\mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|}$ as

$$T_\pi^{(\mathbf{\Lambda}_\zeta)}\mathbf{q} \overset{\text{def}}{=\!=} (\mathbf{I}-\gamma\mathbf{P}_\pi\mathbf{\Lambda}_\zeta)^{-1}\left[\mathbf{r}+\gamma\mathbf{P}_\pi(\mathbf{I}-\mathbf{\Lambda}_\zeta)\mathbf{q}\right].$$

Then the gradient can be written as:

$$\nabla J(\mathbf{w}) = -\frac{1}{2}\left(\mathbf{X}^\top\mathbf{D}_\mu(\mathbf{I}-\gamma\mathbf{P}_\pi\mathbf{\Lambda}_\zeta)^{-1}(\mathbf{I}-\gamma\mathbf{P}_\pi)\,\mathbf{X}\right)^\top\mathbf{C}^{-1}\mathbf{g} \tag{21}$$

$$= -\frac{1}{2}\left(\mathbf{g}-\gamma\mathbf{H}^\top\mathbf{C}^{-1}\mathbf{g}\right), \tag{22}$$

where $\mathbf{g}$, $\mathbf{C}$ and $\mathbf{H}$ are defined in the following way:

$$\mathbf{g} \overset{\text{def}}{=\!=} \mathbf{X}^\top\mathbf{D}_\mu\left(T_\pi^{(\mathbf{\Lambda}_\zeta)}\mathbf{X}\mathbf{w}-\mathbf{X}\mathbf{w}\right), \tag{23}$$

$$\mathbf{C} \overset{\text{def}}{=\!=} (\mathbf{X}^\top\mathbf{D}_\mu\mathbf{X}), \tag{24}$$

$$\mathbf{H} \overset{\text{def}}{=\!=} \left(\mathbf{X}^\top\mathbf{D}_\mu(\mathbf{I}-\gamma\mathbf{P}_\pi\mathbf{\Lambda}_\zeta)^{-1}\mathbf{P}_\pi(\mathbf{I}-\mathbf{\Lambda}_\zeta)\mathbf{X}\right). \tag{25}$$

When this gradient is known, the gradient-descent step would be to add the following to the parameter vector:

$$-\frac{1}{2}\alpha_t\nabla J(\mathbf{w}) = \alpha_t\left(\mathbf{g}-\gamma\mathbf{H}^\top\mathbf{C}^{-1}\mathbf{g}\right). \tag{26}$$

In model-free learning, the gradient is not available to the learner. However, we can form a stochastic gradient descent update by sampling from this gradient. For that, we can express $\mathbf{g}$ and $\mathbf{H}$ in an expectation form, and estimate the vector $\mathbf{C}^{-1}\mathbf{g}$ at a faster time scale.

In Appendix A.3, we derive the resulting stochastic gradient corrected algorithm in details, which we call *the ABQ($\zeta$) algorithm*. This algorithm is defined by the following online updates:

$$\delta_t \overset{\text{def}}{=\!=} R_{t+1}+\gamma\mathbf{w}_t^\top\bar{\mathbf{x}}_{t+1}-\mathbf{w}_t^\top\mathbf{x}_t, \tag{27}$$

$$\tilde{\mathbf{x}}_{t+1} \overset{\text{def}}{=\!=} \sum_a \nu_\zeta(S_{t+1},a)\pi(a|S_{t+1})\mathbf{x}(S_{t+1},a), \tag{28}$$

$$\mathbf{e}_t \overset{\text{def}}{=\!=} \gamma\nu_{\zeta,t}\pi_t\mathbf{e}_{t-1}+\mathbf{x}_t, \tag{29}$$

$$\mathbf{w}_{t+1} \overset{\text{def}}{=\!=} \mathbf{w}_t+\alpha_t\Big(\delta_t\mathbf{e}_t-\gamma\mathbf{e}_t^\top\mathbf{h}_t\left(\bar{\mathbf{x}}_{t+1}-\tilde{\mathbf{x}}_{t+1}\right)\Big), \tag{30}$$

$$\mathbf{h}_{t+1} \overset{\text{def}}{=\!=} \mathbf{h}_t+\beta_t\left(\delta_t\mathbf{e}_t-\left(\mathbf{h}_t^\top\mathbf{x}_t\right)\mathbf{x}_t\right). \tag{31}$$

The iteration (30) carries out stochastic gradient-descent steps to minimize the MSPBE (19), and it differs from (18) as it includes a gradient-correction term $-\gamma\mathbf{e}_t^\top\mathbf{h}_t\left(\bar{\mathbf{x}}_{t+1}-\tilde{\mathbf{x}}_{t+1}\right)$. This correction term involves an extra vector parameter $\mathbf{h}_t$. Note that no importance sampling ratios are needed in the update of $\mathbf{h}_t$ or in the gradient-correction term. The vector $\mathbf{h}_t$ is updated according to (31) at a faster timescale than $\mathbf{w}_t$ by using a second step-size parameter $\beta_t \gg \alpha_t$ when both step sizes are diminishing. One can achieve that by letting $\alpha_t = O(1/t), \beta_t = O(1/t^c), c \in (1/2, 1)$, for instance. In practice, when stability is not a problem, smaller values of $\beta$ often lead to better performance (White & White 2016a).

## 7. Experimental results

We empirically evaluate ABQ($\zeta$) on three policy evaluation tasks: the two-state off-policy task from Section 5, an off-policy policy evaluation adaptation of the Mountain Car domain (Sutton & Barto 1998), and the 7-star Baird's counterexample (Baird 1995, White 2015). In the first two tasks we investigated whether ABQ($\zeta$) can produce correct estimates with less variance compared to GQ($\lambda$) (Maei 2011), the state-of-the-art importance sampling based algorithm for action-value estimation with function approximation. We validate the stability of ABQ($\zeta$) in the final task, where off-policy algorithms without a stability guarantee (e.g., off-policy Q($\lambda$)) tend to diverge.

Although the MDP involved in the two-state task is small, off-policy algorithms may suffer severely in this task as the importance sampling ratio, once in a while, can be as large as 9. We simulated both GQ($\lambda$) and ABQ($\lambda$) on this task for 10000 time steps, starting with $\mathbf{w}_0 = \mathbf{0}$. We averaged the MSE $\|\mathbf{X}\mathbf{w}_t - \mathbf{q}_\pi\|^2_{\mathbf{D}_\mu}$ for the last 5000 time steps. We further averaged this quantity over 100 independent runs. Finally, we divided this error by $\|\mathbf{q}_\pi\|^2_{\mathbf{D}_\mu}$ to obtain the normalized MSE (NMSE).

Figure 3 (left) shows curves for different combinations of the step-size parameters: $\alpha \in [0.001, 0.005, 0.01]$ and $\beta \in [0.001, 0.005, 0.01]$. Performance is shown in the estimated NMSE, with the corresponding standard error for different values of $\lambda$ and $\zeta$. With $\lambda > 0.6$, the error of GQ($\lambda$) increased sharply due to increased influence of importance sampling ratios. It clearly depicts the failure to perform effective multi-step learning by an importance sampling based algorithm when $\lambda$ is large. The error of ABQ($\zeta$) decreased substantially for most step-size combinations as $\zeta$ is increased from 0 to 0.5, and it decreased slightly for some step-size combinations as $\zeta$ was increased up to 1. This example clearly shows that ABQ($\zeta$) can perform effective multi-step learning while avoiding importance sampling ratios. On the other hand, the best performance of GQ($\lambda$) was better than ABQ($\zeta$) for the smallest value of the first step size (i.e., $\alpha = 0.001$). When the step size was too small, GQ($\lambda$) in fact benefited from the occasional large scaling from importance sampling, whereas ABQ($\zeta$) remained conservative in its updates to be safe.

The Mountain Car domain is typically used for policy improvement tasks, but here we use it for off-policy policy evaluation. We constructed the task in such a way that the importance sampling ratios for GQ can be as large as 30, emphasizing the variance issue regarding ratios. In this task, the car starts in the vicinity of the bottom of the valley with a small nonzero speed. The three actions are: *reverse throttle*, *no throttle*, and *forward throttle*. Rewards are -1 at every time step, and state transitions are deterministic. The discount factor $\gamma$ is 0.999. The policies used in the experiments were based on a simple handcrafted policy, which chooses to move forward with full throttle if the velocity of the

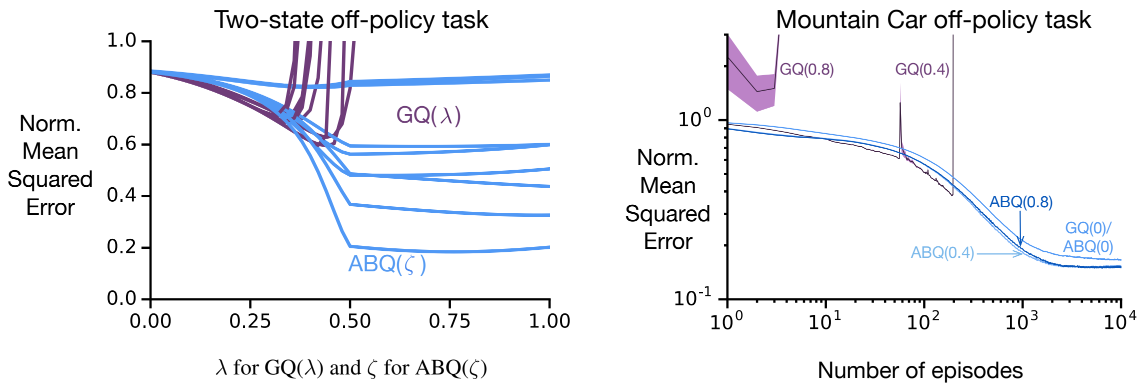

Figure 3: **Left**: Comparison of empirical performance of GQ($\lambda$) and ABQ($\zeta$) on a two-state off-policy policy evaluation task. Performance is shown in normalized mean squared error with respect to different values of $\lambda$ for GQ($\lambda$) and $\zeta$ for ABQ($\zeta$). Different curves are for different combinations of step-size values. GQ($\lambda$) produces large MSE when large $\lambda$ is used. ABQ($\zeta$) tolerates larger values of $\zeta$ and thus can better retain the benefits of multi-step learning compared to GQ($\lambda$).
**Right**: Comparison of empirical performance of GQ($\lambda$) and ABQ($\zeta$) on an off-policy policy evaluation task based on the Mountain Car domain. Each curve shows how learning progresses in terms of estimated normalized mean squared error as more episodes are observed. Different learning curves are for different values of $\lambda$ for GQ($\lambda$) and $\zeta$ for ABQ($\zeta$). All of them are shown for a particular combination of step-size values. The spikes in GQ($\lambda$)'s learning curves are the result of the occasional large values of importance sampling ratios, increasing the variance of the estimate for GQ($\lambda$) as large values of $\lambda$ are chosen. ABQ($\zeta$), on the other hand, can achieve lower mean squared error with larger values of $\zeta$ by reducing the estimation variance.

car was nonzero and toward the goal, and chooses to move away from the goal with full throttle otherwise.

Both the target policy and the behavior policy are based on this policy but choose to randomly explore the other actions differently. More specifically, when the velocity was nonzero and toward the goal, the behavior policy probabilities for the actions *reverse throttle*, *no throttle*, and *forward throttle* are $\left[\frac{1}{300}, \frac{1}{300}, \frac{298}{300}\right]$, and the target policy probabilities are $[0.1, 0.1, 0.8]$, respectively. In the other case, the behavior and the target policy probabilities are $\left[\frac{298}{300}, \frac{1}{300}, \frac{1}{300}\right]$ and $[0.8, 0.1, 0.1]$, respectively. We set the policies this way so that episodes complete under both policies in a reasonable number of time steps, while the importance sampling ratio may occasionally be as large as $0.1 \times 300 = 30$.

The feature vector for each state-action pair contained 32 features for each action, where only the features corresponding to the given action were nonzero, produced using tile coding with ten $4\times4$ tilings. Each algorithm ran on the same 100 independent sequences of samples, each consisting 10,000 episodes. The performance was measured in terms of

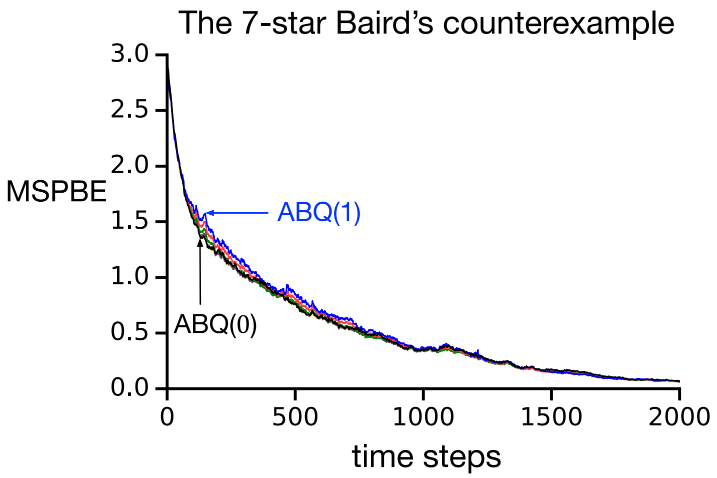

Figure 4: ABQ($\zeta$) is stable on Baird's counterexample for different values of $\zeta$.

the Mean-Squared-Error (MSE) with respect to the estimated value $\hat{\mathbf{q}} \in \mathbb{R}^{30}$ of 30 chosen state-action pairs. These pairs were chosen by running the agent under the behavior policy for 1 million time steps, restarting episodes each time termination occurs, and choosing 30 pairs uniformly randomly from the last half million time steps. The ground truth values for these 30 pairs $\hat{\mathbf{q}}$ were estimated by following the target policy 100 times from those pairs and forming the average. The mean-squared error from $\hat{\mathbf{q}}$ was normalized by $\|\hat{\mathbf{q}}\|_2^2$.

We use the mountain car off-policy task to illustrate through learning curves how $\lambda$ and $\zeta$ affect GQ($\lambda$) and ABQ($\zeta$), respectively. Figure 3 (right) shows the learning curves of ABQ($\zeta$) and GQ($\lambda$) with respect to mean squared errors for three difference values of $\lambda$ and $\zeta$: 0, 0.4, and 0.8, and a particular step-size combination: $\alpha = 0.1/($ # of active features) and $\beta = 0.0$. These learning curves are averages over 100 independent runs. The standard errors of ABQ's estimated MSE here are smaller than the width of the curves shown. ABQ achieved a significantly lower MSE with $\zeta > 0$ than with $\zeta = 0$. On the other hand, GQ performed unreliably with larger $\lambda$. With $\lambda = 0.4$, the learning curves are highly varying from each other due to occasionally having the largest ratio value 30, affecting the update at different time steps. Some learning curves even became unstable, causing the MSE to move away further and further with time, and affecting the average MSE, which explains the spikes. When $\lambda = 0.8$ was chosen, all learning curves became unstable in few steps.

In the 7-star Baird's counterexample, adopted from White (2015), step sizes were set as $\alpha = 0.05$ and $\beta = 0.1$, and the bootstrapping parameter $\zeta$ was chosen evenly between 0 and 1. The Mean Squared Projected Bellman Error (MSPBE) was estimated by averaging over 50 runs. As shown in Figure 4, ABQ($\zeta$) performed stably with all values of $\zeta$ used. This validates empirically the gradient correction in ABQ($\zeta$).

## 8. Action-dependent bootstrapping as a framework for off-policy algorithms

ABQ($\zeta$) is a result of this new idea of varying the bootstrapping parameter in an action-dependent manner so that an explicit presence of importance sampling ratios are mitigated from the update. However, it is not the only action-dependent bootstrapping scheme one can devise. It is possible to bound the product $\lambda_t^n \rho_t^n$ by using other action-dependent bootstrapping schemes. Different schemes not only allow us to derive new algorithms, they may also be used to understand some existing algorithms better. Here, we show how the Retrace algorithm by Munos et al. (2016) can be understood in terms of a particular way of setting the action-dependent bootstrapping parameter.

Retrace is a tabular off-policy algorithm that approaches the variance issue by truncating the importance sampling ratio. We show that such truncations can be understood as varying the action-dependent bootstrapping parameter in a particular manner, first, by constructing a forward-view update with a different action-dependent bootstrapping scheme than ABQ's, second, by deriving the asymptotic solution corresponding to that forward-view update, and third, by showing that the equivalent backward-view update is the same as the Retrace algorithm. For generality, we take these steps in the linear function approximation case and incorporate gradient corrections for stability. The resulting algorithm, we call *AB-Trace*($\zeta$) is a stable generalization of Retrace.

We construct a new forward-view update similar to (15) with $\lambda_\zeta = \nu_\zeta(s,a)\,\mu(a|s)$, where $\nu$ is redefined as $\nu_\zeta(s,a) \overset{\text{def}}{=\!=} \zeta \min\left(\frac{1}{\pi(a|s)}, \frac{1}{\mu(a|s)}\right)$. Here, we treat $1/0 = \infty$ and $0 \cdot \infty = 0$. Then we can directly use Appendix A.1 to derive its asymptotic solution:

$$\mathbf{w}_\infty^\zeta \overset{\text{def}}{=\!=} \mathbf{A}_\zeta^{-1}\mathbf{b}_\zeta, \tag{32}$$

$$\mathbf{A}_\zeta \overset{\text{def}}{=\!=} \mathbf{X}^\top \mathbf{D}_\mu \left(\mathbf{I} - \gamma \mathbf{P}_\pi \mathbf{\Lambda}_\zeta\right)^{-1} \left(\mathbf{I} - \gamma \mathbf{P}_\pi\right)\mathbf{X}, \tag{33}$$

$$\mathbf{b}_\zeta \overset{\text{def}}{=\!=} \mathbf{X}^\top \mathbf{D}_\mu \left(\mathbf{I} - \gamma \mathbf{P}_\pi \mathbf{\Lambda}_\zeta\right)^{-1} \mathbf{r}, \tag{34}$$

where the diagonal elements of $\mathbf{\Lambda}_\zeta$ are $\lambda_\zeta(s,a) = \nu_\zeta(s,a)\,\mu(a|s) = \zeta \min\left(\frac{1}{\pi(a|s)}, \frac{1}{\mu(a|s)}\right)\mu(a|s)$ for different state-action pairs.

A stable forward-view update with gradient correction corresponding to the above asymptotic solution can be derived by directly using the results of Appendix A.3. The resulting algorithmn, AB-Trace($\zeta$), is given by the following updates:

$$\delta_t \overset{\text{def}}{=\!=} R_{t+1} + \gamma \mathbf{w}_t^\top \bar{\mathbf{x}}_{t+1} - \mathbf{w}_t^\top \mathbf{x}_t, \tag{35}$$

$$\nu_\zeta(s,a) \overset{\text{def}}{=\!=} \zeta \min\left(\frac{1}{\pi(a|s)}, \frac{1}{\mu(a|s)}\right), \tag{36}$$

$$\tilde{\mathbf{x}}_{t+1} \overset{\text{def}}{=\!=} \sum_a \nu_\zeta(S_{t+1},a)\pi(a|S_{t+1})\mathbf{x}(S_{t+1},a), \tag{37}$$

$$\mathbf{e}_t \overset{\text{def}}{=\!=} \gamma \nu_{\zeta,t}\pi_t \mathbf{e}_{t-1} + \mathbf{x}_t, \tag{38}$$

$$\mathbf{w}_{t+1} \overset{\text{def}}{=\!=} \mathbf{w}_t + \alpha_t \left(\delta_t \mathbf{e}_t - \gamma \mathbf{e}_t^\top \mathbf{h}_t \left(\bar{\mathbf{x}}_{t+1} - \tilde{\mathbf{x}}_{t+1}\right)\right), \tag{39}$$

$$\mathbf{h}_{t+1} \overset{\text{def}}{=\!=} \mathbf{h}_t + \beta_t \left(\delta_t \mathbf{e}_t - \mathbf{h}_t^\top \mathbf{x}_t \mathbf{x}_t\right), \tag{40}$$

Note that, the factor $\nu_{\zeta,t}\pi_t$ in the eligibility trace vector update can be rewritten as:

$$\begin{aligned}\nu_{\zeta,t}\pi_t &= \zeta \min\left(\frac{1}{\pi_t}, \frac{1}{\mu_t}\right)\pi_t && (41)\\ &= \zeta \min\left(1, \rho_t\right). && (42)\end{aligned}$$

From here, it is easy to see that, if the feature representation is tabular and the gradient correction term is dropped, then the AB-Trace algorithm reduces to the Retrace algorithm.

Finally, we remark that the action-dependent bootstrapping framework provides a principled way of developing stable and efficient off-policy algorithms as well as unifying the existing ones, where AB-Trace and its connection to Retrace is only one instance.

## 9. Related works

A closely related algorithm is Tree Backup by Precup et al. (2000). This algorithm can also be produced as a special case of ABQ($\zeta$), if we remove gradient correction, consider the feature vectors always to be the standard basis, and $\nu_\zeta$ to be always set to a constant, instead of setting it in an action-dependent manner. In the on-policy case, the Tree Backup algorithm fails to achieve the TD(1) solution, whereas ABQ achieves it with $\zeta = 1$.

Our work is not a trivial generalization of this prior work. The Tree Backup algorithm was developed using a different intuition based on backup diagrams and was introduced only for the lookup table case. Not only does ABQ($\zeta$) extend the Tree Backup algorithm, but the idea of action-dependent bootstrapping also played a crucial role in deriving the ABQ($\zeta$) algorithm with gradient correction in a principled way. Another related algorithm is Retrace, which we have already shown to be a special case of the AB-Trace algorithm. The main differences are that Retrace was introduced and analyzed in the case of tabular representation, and thus Retrace is neither stable nor shown to achieve multi-step solutions in the case of function approximation.

Yet another class of related algorithms are where the bootstrapping parameter is adapted based on past data, for example, the works by White and White (2016b) and Mann et al. (2016). Beside adapting on a state-action-pair basis, the main difference between those algorithms and the ones introduced here under the action-dependent bootstrapping framework is that our algorithms adapt the bootstrapping parameter using only the knowledge of policy probabilities for the current state-action pair whereas the algorithms in the other class involve a separate learning procedure for adapting $\lambda$ and hence are more complex.

## 10. Discussion and conclusions

In this paper, we have introduced the first model-free off-policy algorithm ABQ($\zeta$) that can produce multi-step function approximation solutions without requiring to use importance sampling ratios. The key to this algorithm is allowing the amount of bootstrapping to vary in an action-dependent manner, instead of keeping them constant or varying only with states. Part of this action-dependent bootstrapping factor mitigates the importance sampling ratios while the rest of the factor is spent achieving multi-step bootstrapping. The resulting effect is that the large variance issue with importance sampling ratios is readily removed without giving up multi-step learning. This makes ABQ($\zeta$) more suitable for practical use. Action-dependent bootstrapping provides an insightful and well-founded framework for deriving off-policy algorithms with reduced variance. The same idea may be applied to state-value estimation. The ratios cannot be eliminated completely in this case; nevertheless, reduction in the variance can be expected. According to an alternative explanation based on backup diagrams (Precup et al. 2000), ABQ updates may be seen as devoid of importance sampling ratios, while the action-dependent bootstrapping framework can now provide us a clearer mechanistic view of how updates without importance sampling ratios can perform off-policy learning.

The convergence of this algorithm can be analyzed similarly to the work by Karmakar and Bhatnagar (2015) for diminishing step sizes and Yu (2015) for constant step sizes, by using properties of the eligibility traces and stochastic approximation theory. Investigating the possibility to include true online updates (van Seijen et al. 2016) is also an interesting direction for future work.

## References

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

Baird, L. C. (1995). Residual algorithms: Reinforcement learning with function approximation. In *Proceedings of the 12th International Conference on Machine Learning*, pp. 30–37.

Dudík, M., Langford, J., Li, L. (2011). Doubly robust policy evaluation and learning. In *Proceedings of the 28th International Conference on Machine Learning*, pp. 1097–1104.

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

Hallak, A., Tamar, A., Munos, R., Mannor, S. (2015a). Generalized emphatic temporal difference learning: bias-variance analysis. *arXiv preprint* arXiv:1509.05172.

Hallak, A., Schnitzler, F., Mann, T., Mannor, S. (2015b). Off-policy model-based learning under unknown factored dynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 711–719.

Hammersley, J. M. Handscomb, D. C. (1964). Monte Carlo methods, *Methuen & co. Ltd.*, London, pp. 40,

Harutyunyan, A., Bellemare, M. G., Stepleton, T., Munos, R. (2016). Q ($\lambda$) with off-policy corrections. *arXiv preprint* arXiv:1602.04951.

Karmakar, P., Bhatnagar, S. (2015). Two timescale stochastic approximation with controlled Markov noise and off-policy temporal difference learning. *arXiv preprint* arXiv:1503. 09105.

Jiang, N., Li, L. (2015). Doubly robust off-policy evaluation for reinforcement learning. *arXiv preprint* arXiv:1511.03722.

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

Li, L., Munos, R., Szepesvari, C. (2015). Toward minimax off-policy value estimation. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, pp. 608–616.

Liu, J. S. (2001). *Monte Carlo strategies in scientific computing.* Berlin, Springer-Verlag.

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

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

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.

Mahmood, A. R., Sutton, R. S. (2015). Off-policy learning based on weighted importance sampling with linear computational complexity. In *Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence.*

Mahmood, A. R., Yu, H., White, M., Sutton, R. S. (2015). Emphatic temporal-difference learning. *European Workshop on Reinforcement Learning 12,arXiv preprint* ArXiv:1507.01569.

Mann, T. A., Penedones, H., Mannor, S., Hester, T. (2016). Adaptive lambda least-squares temporal difference learning. arXiv preprint arXiv:1612.09465.

Paduraru, C. (2013). *Off-policy Evaluation in Markov Decision Processes*, PhD thesis, McGill University.

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.

Precup, D., Sutton, R. S., Dasgupta, S. (2001). Off-policy temporal-difference learning with function approximation. In *Proceedings of the 18th International Conference on Machine Learning.*

Munos, R, Stepleton, T, Harutyunyan, A, Bellemare, M. G. (2016). Safe and efficient off-policy reinforcement learning. In *Proceedings of Neural Information Processing Systems.*

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

Sutton, R.S., Singh, S.P. (1994). On bias and step size in temporal-difference learning. In *Proceedings of the Eighth Yale Workshop on Adaptive and Learning Systems*, pp. 91-96.

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

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

Sutton, R. S., Mahmood, A. R., Precup, D., van Hasselt, H. (2014). A new Q($\lambda$) with interim forward view and Monte Carlo equivalence. In *Proceedings of the 31st International Conference on Machine Learning.* JMLR W&CP 32(2).

Sutton, R. S., Mahmood, A. R., White, M. (2016). An emphatic approach to the problem of off-policy temporal-difference learning. *Journal of Machine Learning Research 17*, (73):1–29.

Thomas, P. S. (2015). *Safe Reinforcement Learning.* PhD thesis, University of Massachusetts Amherst.

Thomas, P. S., Brunskill, E. (2016). Data-efficient off-policy policy evaluation for reinforcement learning. *arXiv preprint* arXiv:1604.00923.

Tsitsiklis, J. N., Van Roy, B. (1997). An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42(5), 674–690.

van Hasselt, H. (2011). *Insights in Reinforcement Learning: formal analysis and empirical evaluation of temporal-difference learning algorithms*. PhD thesis, Universiteit Utrecht.

van Hasselt, H., Mahmood, A. R., Sutton, R. S. (2014). Off-policy TD($\lambda$) 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($\lambda$). In *Proceedings of the 31st International Conference on Machine Learning*. JMLR W&CP 32(1):692–700.

van Seijen, H., Mahmood, A. R., Pilarski, P. M., Machado, M. C., Sutton, R. S. (2016). True online temporal-difference learning. *Journal of Machine Learning Research 17* (145):1–40.

White, A., Modayil, J., Sutton, R. S. (2012). Scaling life-long off-policy learning. In *Proceedings of the Second Joint IEEE International Conference on Development and Learning and on Epigenetic Robotics*, San Diego, USA.

White, A. (2015). *Developing a Predictive Approach to Knowledge*. PhD thesis, University of Alberta.

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

White, M., White, A. (2016b). A greedy approach to adapting the trace parameter for temporal difference learning. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, 557–565.

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

Yu, H. (2015). On convergence of emphatic temporal-difference learning. *arXiv preprint* arXiv:1506.02582; a shorter version appeared in The 28th Annual Conference on Learning Theory (COLT) 2015.

Yu, H. (2016). Weak convergence properties of constrained emphatic temporal-difference learning with constant and slowly diminishing stepsize. *Journal of Machine Learning Research 17*(220):1–58.

## Appendix

### A.1 Derivation of the ABQ($\zeta$) solution

We follow the approach taken by Sutton et al. (2016) to derive the asymptotic solution. The key-step in their approach is to derive the expected update corresponding to the stochastic update.

The off-line backward-view update defined by (15) can be rewritten as:

$$\begin{aligned}
\Delta \mathbf{w}_t &= \alpha_t \left( H_t^\zeta - \mathbf{w}^\top \mathbf{x}_t \right) \mathbf{x}_t && (43)\\
&= \alpha_t \sum_{n=t}^{\infty} \gamma^{n-t} \nu_{\zeta,t+1}^{n} \pi_{t+1}^{n} \delta_n \mathbf{x}_t && (44)\\
&= \alpha_t \sum_{n=t}^{\infty} \gamma^{n-t} \nu_{\zeta,t+1}^{n} \pi_{t+1}^{n} \left( R_{n+1} + \gamma \mathbf{w}^\top \bar{\mathbf{x}}_{n+1} - \mathbf{w}^\top \mathbf{x}_n \right) \mathbf{x}_t && (45)\\
&= \alpha_t \left( \underbrace{\sum_{n=t}^{\infty} \gamma^{n-t} \nu_{\zeta,t+1}^{n} \pi_{t+1}^{n} R_{n+1} \mathbf{x}_t}_{\mathbf{b}_t} - \underbrace{\sum_{n=t}^{\infty} \gamma^{n-t} \nu_{\zeta,t+1}^{n} \pi_{t+1}^{n} \mathbf{x}_t \left( \mathbf{x}_n - \gamma \bar{\mathbf{x}}_{n+1} \right)^\top}_{\mathbf{A}_t} \mathbf{w} \right) && (46)\\
&= \alpha_t \left( \mathbf{b}_t - \mathbf{A}_t \mathbf{w} \right). && (47)
\end{aligned}$$

With $\mathbf{w}$ held fixed, the expected update direction for (15) is given by the expectation of $\mathbf{b}_t - \mathbf{A}_t\mathbf{w}$ with respect to the stationary Markov chain $\{(S_t, A_t, R_{t+1})\}$ induced by $\mu$. Specifically, let $\mathrm{E}_0$ denote expectation with respect to this stationary Markov chain. We calculate the expectation $\mathrm{E}_0[\mathbf{A}_t]$ (for any fixed $t$) as:

$$\begin{aligned}
\mathrm{E}_0[\mathbf{A}_t] &= \mathrm{E}_0 \left[ \sum_{n=t}^{\infty} \gamma^{n-t} \nu_{\zeta,t+1}^{n} \pi_{t+1}^{n} \mathbf{x}_t \left( \mathbf{x}_n - \gamma \bar{\mathbf{x}}_{n+1} \right)^\top \right] && (48)\\
&= \sum_{s,a} d_\mu(s,a) \mathrm{E}_\mu \left[ \sum_{n=t}^{\infty} \gamma^{n-t} \nu_{\zeta,t+1}^{n} \pi_{t+1}^{n} \mathbf{x}_t \left( \mathbf{x}_n - \gamma \bar{\mathbf{x}}_{n+1} \right)^\top \middle| S_t = s, A_t = a \right] && (49)\\
&= \sum_{s,a} \mathbf{x}(s,a) d_\mu(s,a) \underbrace{\mathrm{E}_\mu \left[ \sum_{n=t}^{\infty} \gamma^{n-t} \nu_{\zeta,t+1}^{n} \pi_{t+1}^{n} \left( \mathbf{x}_n - \gamma \bar{\mathbf{x}}_{n+1} \right)^\top \middle| S_t = s, A_t = a \right]}_{\mathbf{e}^\top(s,a) \in \mathbb{R}^n} && (50)\\
&= \sum_{s,a} \mathbf{x}(s,a) d_\mu(s,a) \mathbf{e}(s,a)^\top && (51)\\
&= \mathbf{X}^\top \mathbf{D}_\mu \mathbf{E}. && (52)
\end{aligned}$$

Here the matrix $\mathbf{E} \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|\times n}$ is such that $[\mathbf{E}]_{sa,:} \overset{\text{def}}{=\!=} \mathbf{e}(s,a)$. We can write $\mathbf{e}(s,a)^\top$ recursively as:

$$\mathbf{e}(s,a)^\top = \mathrm{E}_\mu \left[ \sum_{n=t}^{\infty} \gamma^{n-t} \nu_{\zeta,t+1}^{n} \pi_{t+1}^{n} \left( \mathbf{x}_n - \gamma \bar{\mathbf{x}}_{n+1} \right)^\top \middle| S_t = s, A_t = a \right] \qquad (53)$$

$$= \mathrm{E}_\mu\left[(\mathbf{x}_t - \gamma\bar{\mathbf{x}}_{t+1})^\top \middle| S_t = s, A_t = a\right] \tag{54}$$

$$+ \mathrm{E}_\mu\left[\sum_{n=t+1}^{\infty} \gamma^{n-t}\lambda_{\zeta,t+1}^{n}\rho_{t+1}^{n}\left(\mathbf{x}_n - \gamma\bar{\mathbf{x}}_{n+1}\right)^\top \middle| S_t = s, A_t = a\right] \tag{55}$$

$$= \left(\mathbf{x}(s,a)^\top - \mathrm{E}_\mu\left[\bar{\mathbf{x}}_{t+1}^\top \middle| S_t = s, A_t = a\right]\right) \tag{56}$$

$$+ \gamma\mathrm{E}_\mu\left[\lambda_{\zeta,t+1}\rho_{t+1}\sum_{n=t+1}^{\infty} \gamma^{n-t-1}\lambda_{\zeta,t+2}^{n}\rho_{t+2}^{n}\left(\mathbf{x}_n - \gamma\bar{\mathbf{x}}_{n+1}\right)^\top \middle| S_t = s, A_t = a\right] \tag{57}$$

$$= \left(\mathbf{x}(s,a)^\top - \gamma\sum_{s'} p(s'|s,a)\sum_{a'}\pi(a'|s')\mathbf{x}(s',a')^\top\right) \tag{58}$$

$$+ \gamma\sum_{s'a'} p(s'|s,a)\mu(a'|s')\lambda_\zeta(s',a')\frac{\pi(a'|s')}{\mu(a'|s')} \tag{59}$$

$$\times \mathrm{E}_\mu\left[\sum_{n=t+1}^{\infty} \gamma^{n-t-1}\nu_{\zeta,t+2}^{n}\pi_{t+2}^{n}\left(\mathbf{x}_n - \gamma\bar{\mathbf{x}}_{n+1}\right)^\top \middle| S_{t+1} = s', A_{t+1} = a'\right] \tag{60}$$

$$= \left(\mathbf{x}(s,a)^\top - \gamma\sum_{s',a'}[\mathbf{P}_\pi]_{sa,s'a'}\mathbf{x}(s',a')^\top\right) \tag{61}$$

$$+ \gamma\sum_{s'a'} p_\pi(s',a'|s,a)\lambda_\zeta(s',a')\mathbf{e}(s',a')^\top \tag{62}$$

$$= \left(\mathbf{x}(s,a)^\top - \gamma\sum_{s',a'}[\mathbf{P}_\pi]_{sa,s'a'}\mathbf{x}(s',a')^\top\right) + \gamma\sum_{s'a'}[\mathbf{P}_\pi\mathbf{\Lambda}_\zeta]_{sa,s'a'}\mathbf{e}(s',a')^\top. \tag{63}$$

Therefore, we can write $\mathbf{E}$ recursively as:

$$\mathbf{E} = (\mathbf{I} - \gamma\mathbf{P}_\pi)\mathbf{X} + \gamma\mathbf{P}_\pi\mathbf{\Lambda}_\zeta\mathbf{E} \tag{64}$$

$$= (\mathbf{I} - \gamma\mathbf{P}_\pi)\mathbf{X} + \gamma\mathbf{P}_\pi\mathbf{\Lambda}_\zeta(\mathbf{I} - \gamma\mathbf{P}_\pi)\mathbf{X} + (\gamma\mathbf{P}_\pi\mathbf{\Lambda}_\zeta)^2(\mathbf{I} - \gamma\mathbf{P}_\pi)\mathbf{X} + \cdots \tag{65}$$

$$= \left(\mathbf{I} + \gamma\mathbf{P}_\pi\mathbf{\Lambda}_\zeta + (\gamma\mathbf{P}_\pi\mathbf{\Lambda}_\zeta)^2 + \cdots\right)(\mathbf{I} - \gamma\mathbf{P}_\pi)\mathbf{X} \tag{66}$$

$$= (\mathbf{I} - \gamma\mathbf{P}_\pi\mathbf{\Lambda}_\zeta)^{-1}(\mathbf{I} - \gamma\mathbf{P}_\pi)\mathbf{X}. \tag{67}$$

It then follows that

$$\mathrm{E}_0\left[\mathbf{A}_t\right] = \mathbf{X}^\top\mathbf{D}_\mu\mathbf{E} = \mathbf{X}^\top\mathbf{D}_\mu(\mathbf{I} - \gamma\mathbf{P}_\pi\mathbf{\Lambda}_\zeta)^{-1}(\mathbf{I} - \gamma\mathbf{P}_\pi)\mathbf{X} \tag{68}$$

$$= \mathbf{A}_\zeta. \tag{69}$$

The proof of $\mathrm{E}_0[\mathbf{b}_t] = \mathbf{b}_\zeta$ is similar, and we omit it here.

Thus we have shown that the expected update corresponding to (15) is:

$$\Delta\mathbf{w} = \mathrm{E}_0[\Delta\mathbf{w}_t] = \alpha(\mathbf{b}_\zeta - \mathbf{A}_\zeta\mathbf{w}). \tag{70}$$

When $\mathbf{A}_\zeta$ is invertible, $\mathbf{w}_\infty^\zeta = \mathbf{A}_\zeta^{-1}\mathbf{b}_\zeta$ is the desired solution that (15) aims to attain in the limit. Note, however, that this expected update $\Delta\mathbf{w} = \alpha(\mathbf{b}_\zeta - \mathbf{A}_\zeta\mathbf{w})$ may not be stable, which is a common problem with many off-policy temporal-difference learning algorithms. A modification to (15), such as gradient correction, is needed to ensure stability, which is attained by the ABQ($\zeta$) algorithm described in Section 6.

## A.2 Derivation of the backward-view update of (15)

Using the forward-view update given by (15), the total update can be given by:

$$
\begin{aligned}
\sum_{t=0}^{\infty} \Delta \mathbf{w}_t &= \sum_{t=0}^{\infty} \alpha_t \left( H_t^{\zeta} - \mathbf{w}^{\top} \mathbf{x}_t \right) \mathbf{x}_t && (71) \\
&= \sum_{t=0}^{\infty} \sum_{n=t}^{\infty} \gamma^{n-t} \nu_{\zeta,t+1}^{n} \pi_{t+1}^{n} \delta_n \mathbf{x}_t && (72) \\
&= \sum_{t=0}^{\infty} \alpha_t \sum_{n=0}^{t} \gamma^{t-n} \nu_{\zeta,n+1}^{t} \pi_{n+1}^{t} \delta_t \mathbf{x}_n && (73) \\
&= \sum_{t=0}^{\infty} \alpha_t \delta_t \underbrace{\sum_{n=0}^{t} \gamma^{t-n} \nu_{\zeta,n+1}^{t} \pi_{n+1}^{t} \mathbf{x}_n}_{\mathbf{e}_t} && (74) \\
&= \sum_{t=0}^{\infty} \alpha_t \delta_t \mathbf{e}_t. && (75)
\end{aligned}
$$

Therefore, the backward-view update can be written as:

$$
\Delta \mathbf{w}_t^{B} = \alpha_t \delta_t \mathbf{e}_t. \qquad (76)
$$

The eligibility trace vector $\mathbf{e}_t \in \mathbb{R}^n$ can be written recursively as:

$$
\begin{aligned}
\mathbf{e}_t &= \sum_{n=0}^{t} \gamma^{t-n} \nu_{\zeta,n+1}^{t} \pi_{n+1}^{t} \mathbf{x}_n && (77) \\
&= \gamma \nu_{\zeta,t} \pi_t \sum_{n=0}^{t-1} \gamma^{t-n-1} \nu_{\zeta,n+1}^{t-1} \pi_{n+1}^{t-1} \mathbf{x}_n + \mathbf{x}_t && (78) \\
&= \gamma \nu_{\zeta,t} \pi_t \mathbf{e}_{t-1} + \mathbf{x}_t. && (79)
\end{aligned}
$$

## A.3 Derivation of ABQ($\zeta$)

The key step in deriving a gradient-based TD algorithm, as proposed by Maei (2011), is to formulate an associated Mean Squared Projected Bellman Error (MSPBE) and produce its gradient, which can then be sampled to produce stochastic updates.

The MSPBE for the update (15) is given by:

$$\begin{aligned}
J(\mathbf{w}) &= \left\|\Pi_\mu T_\pi^{(\boldsymbol{\Lambda}_\zeta)}\mathbf{X}\mathbf{w} - \mathbf{X}\mathbf{w}\right\|^2_{\mathbf{D}_\mu} && (80)\\
&= \left\|\Pi_\mu \left(T_\pi^{(\boldsymbol{\Lambda}_\zeta)}\mathbf{X}\mathbf{w} - \mathbf{X}\mathbf{w}\right)\right\|^2_{\mathbf{D}_\mu} && (81)\\
&= \left(T_\pi^{(\boldsymbol{\Lambda}_\zeta)}\mathbf{X}\mathbf{w} - \mathbf{X}\mathbf{w}\right)^\top \Pi_\mu^\top \mathbf{D}_\mu \Pi_\mu \left(T_\pi^{(\boldsymbol{\Lambda}_\zeta)}\mathbf{X}\mathbf{w} - \mathbf{X}\mathbf{w}\right) && (82)\\
&= \left(\mathbf{X}^\top\mathbf{D}_\mu \left(T_\pi^{(\boldsymbol{\Lambda}_\zeta)}\mathbf{X}\mathbf{w} - \mathbf{X}\mathbf{w}\right)\right)^\top (\mathbf{X}^\top\mathbf{D}_\mu\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{D}_\mu \left(T_\pi^{(\boldsymbol{\Lambda}_\zeta)}\mathbf{X}\mathbf{w} - \mathbf{X}\mathbf{w}\right). && (83)
\end{aligned}$$

Here, the Bellman operator corresponding to the bootstrapping matrix $\boldsymbol{\Lambda}_\zeta$ is defined for all $\mathbf{q}\in\mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|}$ as

$$T_\pi^{(\boldsymbol{\Lambda}_\zeta)}\mathbf{q} \overset{\text{def}}{=\!=} (\mathbf{I} - \gamma\mathbf{P}_\pi\boldsymbol{\Lambda}_\zeta)^{-1}\left[\mathbf{r} + \gamma\mathbf{P}_\pi(\mathbf{I}-\boldsymbol{\Lambda}_\zeta)\mathbf{q}\right].$$

$$\begin{aligned}
\text{Let } \mathbf{g} &\overset{\text{def}}{=\!=} \mathbf{X}^\top\mathbf{D}_\mu\left(T_\pi^{(\boldsymbol{\Lambda}_\zeta)}\mathbf{X}\mathbf{w} - \mathbf{X}\mathbf{w}\right) && (84)\\
&= \mathbf{X}^\top\mathbf{D}_\mu\left((\mathbf{I} - \gamma\mathbf{P}_\pi\boldsymbol{\Lambda}_\zeta)^{-1}\left[\mathbf{r} + \gamma\mathbf{P}_\pi(\mathbf{I}-\boldsymbol{\Lambda}_\zeta)\mathbf{X}\mathbf{w}\right] - \mathbf{X}\mathbf{w}\right) && (85)\\
&= \mathbf{X}^\top\mathbf{D}_\mu\left((\mathbf{I} - \gamma\mathbf{P}_\pi\boldsymbol{\Lambda}_\zeta)^{-1}\left[\mathbf{r} + \left((\mathbf{I} - \gamma\mathbf{P}_\pi\boldsymbol{\Lambda}_\zeta) - (\mathbf{I}-\gamma\mathbf{P}_\pi)\right)\mathbf{X}\mathbf{w}\right] - \mathbf{X}\mathbf{w}\right) && (86)\\
&= \mathbf{X}^\top\mathbf{D}_\mu\left((\mathbf{I} - \gamma\mathbf{P}_\pi\boldsymbol{\Lambda}_\zeta)^{-1}\mathbf{r} + \mathbf{X}\mathbf{w} - (\mathbf{I} - \gamma\mathbf{P}_\pi\boldsymbol{\Lambda}_\zeta)^{-1}(\mathbf{I}-\gamma\mathbf{P}_\pi)\mathbf{X}\mathbf{w} - \mathbf{X}\mathbf{w}\right) && (87)\\
&= \mathbf{X}^\top\mathbf{D}_\mu(\mathbf{I} - \gamma\mathbf{P}_\pi\boldsymbol{\Lambda}_\zeta)^{-1}(\mathbf{r} - (\mathbf{I}-\gamma\mathbf{P}_\pi)\mathbf{X}\mathbf{w}). && (88)
\end{aligned}$$

Also let $\mathbf{C} \overset{\text{def}}{=\!=} (\mathbf{X}^\top\mathbf{D}_\mu\mathbf{X})$. Then the gradient can be written as:

$$\begin{aligned}
\nabla J(\mathbf{w}) &= -\frac{1}{2}\left(\mathbf{X}^\top\mathbf{D}_\mu(\mathbf{I} - \gamma\mathbf{P}_\pi\boldsymbol{\Lambda}_\zeta)^{-1}(\mathbf{I}-\gamma\mathbf{P}_\pi)\,\mathbf{X}\right)^\top \mathbf{C}^{-1}\mathbf{g} && (89)\\
&= -\frac{1}{2}\left(\mathbf{X}^\top\mathbf{D}_\mu(\mathbf{I} - \gamma\mathbf{P}_\pi\boldsymbol{\Lambda}_\zeta)^{-1}\left((\mathbf{I} - \gamma\mathbf{P}_\pi\boldsymbol{\Lambda}_\zeta)\,\mathbf{X} - \gamma\mathbf{P}_\pi(\mathbf{I}-\boldsymbol{\Lambda}_\zeta)\mathbf{X}\right)\right)^\top \mathbf{C}^{-1}\mathbf{g} && (90)\\
&= -\frac{1}{2}\left(\mathbf{X}^\top\mathbf{D}_\mu\mathbf{X} - \gamma\mathbf{X}^\top\mathbf{D}_\mu\,(\mathbf{I} - \gamma\mathbf{P}_\pi\boldsymbol{\Lambda}_\zeta)^{-1}\,\mathbf{P}_\pi(\mathbf{I}-\boldsymbol{\Lambda}_\zeta)\mathbf{X}\right)^\top \mathbf{C}^{-1}\mathbf{g} && (91)\\
&= -\frac{1}{2}\left(\mathbf{g} - \gamma\left(\underbrace{\mathbf{X}^\top\mathbf{D}_\mu\,(\mathbf{I} - \gamma\mathbf{P}_\pi\boldsymbol{\Lambda}_\zeta)^{-1}\,\mathbf{P}_\pi(\mathbf{I}-\boldsymbol{\Lambda}_\zeta)\mathbf{X}}_{\mathbf{H}}\right)^\top \mathbf{C}^{-1}\mathbf{g}\right) && (92)\\
&= -\frac{1}{2}\left(\mathbf{g} - \gamma\mathbf{H}^\top\mathbf{C}^{-1}\mathbf{g}\right). && (93)
\end{aligned}$$

So if we know the gradient, the gradient-descent step would be to add the following to the parameter vector:

$$-\frac{1}{2}\alpha_t\nabla J(\mathbf{w}) = \alpha_t\left(\mathbf{g} - \gamma\mathbf{H}^\top\mathbf{C}^{-1}\mathbf{g}\right). \qquad (94)$$

Now to derive the stochastic updates for ABQ($\zeta$), let us consider a double-ended *stationary* Markov chain induced by $\mu$, $\{\dots,(S_{-2},A_{-2},R_{-1}),(S_{-1},A_{-1},R_0),(S_0,A_0,R_1),(S_1,A_1,R_2),\dots\}$. Let $\mathrm{E}_0$ denote expectation with respect to the probability distribution of this stationary Markov chain. Fix $t$ to be any integer. Then we can write the first term $\mathbf{g}$ in $\nabla J(\mathbf{w})$ in expectation form as follows:

$$\begin{aligned}
\mathbf{g} &= \mathbf{X}^\top\mathbf{D}_\mu(\mathbf{I}-\gamma\mathbf{P}_\pi\mathbf{\Lambda}_\zeta)^{-1}\left(\mathbf{r}-(\mathbf{I}-\gamma\mathbf{P}_\pi)\mathbf{X}\mathbf{w}\right) && (95)\\
&= \mathbf{X}^\top\mathbf{D}_\mu\left(\mathbf{I}-\gamma\mathbf{P}_\pi\mathbf{\Lambda}_\zeta\right)^{-1}\mathbf{r}-\mathbf{X}^\top\mathbf{D}_\mu\left(\mathbf{I}-\gamma\mathbf{P}_\pi\mathbf{\Lambda}_\zeta\right)^{-1}(\mathbf{I}-\gamma\mathbf{P}_\pi)\mathbf{X}\mathbf{w} && (96)\\
&= \mathbf{b}_\zeta-\mathbf{A}_\zeta\mathbf{w};\qquad (\mathbf{A}_\zeta \text{ and } \mathbf{b}_\zeta \text{ as defined by (16) and (17) }) && (97)\\
&= \mathrm{E}_0\left[\left(H_t^\zeta-\mathbf{w}^\top\mathbf{x}_t\right)\mathbf{x}_t\right] && (98)\\
&= \mathrm{E}_0\left[\sum_{n=t}^{\infty}\gamma^{n-t}\nu_{\zeta,t+1}^{n}\pi_{t+1}^{n}\delta_n\mathbf{x}_t\right] && (99)\\
&= \mathrm{E}_0\left[\delta_t\mathbf{x}_t+\sum_{n=t+1}^{\infty}\gamma^{n-t}\nu_{\zeta,t+1}^{n}\pi_{t+1}^{n}\delta_n\mathbf{x}_t\right] && (100)\\
&= \mathrm{E}_0\left[\delta_t\mathbf{x}_t+\sum_{n=t}^{\infty}\gamma^{n-(t-1)}\nu_{\zeta,t}^{n}\pi_{t}^{n}\delta_n\mathbf{x}_{t-1}\right];\quad \text{shifting indices and using stationarity} && (101)\\
&= \mathrm{E}_0\left[\delta_t\mathbf{x}_t+\gamma\nu_{\zeta,t}\pi_t\sum_{n=t}^{\infty}\gamma^{n-t}\nu_{\zeta,t+1}^{n}\pi_{t+1}^{n}\delta_n\mathbf{x}_{t-1}\right] && (102)\\
&= \mathrm{E}_0\left[\delta_t\mathbf{x}_t+\gamma\nu_{\zeta,t}\pi_t\left(\delta_t\mathbf{x}_{t-1}+\sum_{n=t+1}^{\infty}\gamma^{n-t}\nu_{\zeta,t+1}^{n}\pi_{t+1}^{n}\delta_n\mathbf{x}_{t-1}\right)\right] && (103)\\
&= \mathrm{E}_0\left[\delta_t\left(\mathbf{x}_t+\gamma\nu_{\zeta,t}\pi_t\mathbf{x}_{t-1}+\cdots\right)\right];\qquad \text{shifting indices and using stationarity} && (104)\\
&= \mathrm{E}_0\left[\delta_t\mathbf{e}_t\right], && (105)
\end{aligned}$$

where $\mathbf{e}_t$ is a well-defined random variable and can be written recursively as:

$$\mathbf{e}_t=\mathbf{x}_t+\gamma\nu_{\zeta,t}\pi_t\mathbf{x}_{t-1}+\cdots=\mathbf{x}_t+\gamma\nu_{\zeta,t}\pi_t\mathbf{e}_{t-1}. \qquad (106)$$

Similarly, we can also express the term $\mathbf{H}$ in $\nabla J(\mathbf{w})$ in expectation form. Let us define

$$\tilde{\mathbf{x}}_t=\sum_a\lambda_\zeta(S_t,a)\pi(a|S_t)\mathbf{x}(S_t,a). \qquad (107)$$

Then we can write:

$$\begin{aligned}
\mathbf{H} &= \mathbf{X}^\top\mathbf{D}_\mu\left(\mathbf{I}-\gamma\mathbf{P}_\pi\mathbf{\Lambda}_\zeta\right)^{-1}\mathbf{P}_\pi(\mathbf{I}-\mathbf{\Lambda}_\zeta)\mathbf{X} && (108)\\
&= \mathbf{X}^\top\mathbf{D}_\mu\mathbf{P}_\pi(\mathbf{I}-\mathbf{\Lambda}_\zeta)\mathbf{X}+\mathbf{X}^\top\mathbf{D}_\mu\gamma\mathbf{P}_\pi\mathbf{\Lambda}_\zeta\mathbf{P}_\pi(\mathbf{I}-\mathbf{\Lambda}_\zeta)\mathbf{X}+\cdots && (109)\\
&= \sum_{s,a}d_\mu(s,a)\mathbf{x}(s,a)\sum_{s'a'}p(s'|s,a)\pi(a'|s')(1-\lambda_\zeta(s',a'))\mathbf{x}(s',a')^\top && (110)\\
&\quad+\mathbf{X}^\top\mathbf{D}_\mu\gamma\mathbf{P}_\pi\mathbf{\Lambda}_\zeta\mathbf{P}_\pi(\mathbf{I}-\mathbf{\Lambda}_\zeta)\mathbf{X}+\cdots && (111)\\
&= \mathrm{E}_0\left[\mathbf{x}_t\left(\bar{\mathbf{x}}_{t+1}-\tilde{\mathbf{x}}_{t+1}\right)^\top\right] && (112)
\end{aligned}$$

$$+ \gamma \sum_{s,a} d_\mu(s,a)\mathbf{x}(s,a) \sum_{s',a'} p(s'|s,a)\pi(a'|s')\zeta(a'|s')\mu(a'|s') \tag{113}$$

$$\times \sum_{s''a''} p(s''|s',a')\pi(a''|s'')(1-\lambda_\zeta(s'',a''))\mathbf{x}(s'',a'')^\top + \cdots \tag{114}$$

$$= \mathrm{E}_0\left[\mathbf{x}_t\left(\bar{\mathbf{x}}_{t+1} - \tilde{\mathbf{x}}_{t+1}\right)^\top\right] \tag{115}$$

$$+ \mathrm{E}_0\left[\gamma\nu_{\zeta,t+1}\pi_{t+1}\mathbf{x}_t\left(\bar{\mathbf{x}}_{t+2} - \tilde{\mathbf{x}}_{t+2}\right)^\top\right] + \cdots; \quad \text{shifting indices and using stationarity} \tag{116}$$

$$= \mathrm{E}_0\left[\mathbf{x}_t\left(\bar{\mathbf{x}}_{t+1} - \tilde{\mathbf{x}}_{t+1}\right)^\top\right] \tag{117}$$

$$+ \mathrm{E}_0\left[\gamma\nu_{\zeta,t}\pi_t\mathbf{x}_{t-1}\left(\bar{\mathbf{x}}_{t+1} - \tilde{\mathbf{x}}_{t+1}\right)^\top\right] + \cdots; \quad \text{shifting indices and using stationarity} \tag{118}$$

$$= \mathrm{E}_0\left[\left(\mathbf{x}_t + \gamma\nu_{\zeta,t}\pi_t\mathbf{x}_{t-1} + \cdots\right)\left(\bar{\mathbf{x}}_{t+1} - \tilde{\mathbf{x}}_{t+1}\right)^\top)\right] \tag{119}$$

$$= \mathrm{E}_0\left[\mathbf{e}_t\left(\bar{\mathbf{x}}_{t+1} - \tilde{\mathbf{x}}_{t+1}\right)^\top\right]. \tag{120}$$

Therefore, a stochastic update corresponding to the expected gradient-descent update can be written as:

$$\Delta\mathbf{w} = \alpha_t\left(\delta_t\mathbf{e}_t - \gamma\left(\bar{\mathbf{x}}_{t+1} - \tilde{\mathbf{x}}_{t+1}\right)\mathbf{e}_t^\top\mathbf{C}^{-1}\mathbf{g}\right). \tag{121}$$

The vector $\mathbf{C}^{-1}\mathbf{g}$ can be estimated from samples by LMS but at a faster time scale and with a larger step-size parameter $\beta$:

$$\Delta\mathbf{h} = \beta_t\left(\delta_t\mathbf{e}_t - \mathbf{h}^\top\mathbf{x}_t\mathbf{x}_t\right). \tag{122}$$

It can be shown that with $\mathbf{w}$ held fixed, under standard diminishing step-size rules for $\beta_t$, the $\{\mathbf{h}_t\}$ produced by the above updates converges to

$$\mathbf{h}_\infty = \left(\mathrm{E}_0\left[\mathbf{x}_t\mathbf{x}_t^\top\right]\right)^{-1}\mathrm{E}_0\left[\delta_t\mathbf{e}_t\right] \tag{123}$$

$$= \left(\mathbf{X}^\top\mathbf{D}_\mu\mathbf{X}\right)^{-1}\mathrm{E}_0\left[\delta_t\mathbf{e}_t\right] \tag{124}$$

$$= \mathbf{C}^{-1}\mathbf{g}. \tag{125}$$

Putting these pieces together, and also allowing $\mathbf{w}$ and $\mathbf{h}$ to vary over time, we obtain the updates of the on-line gradient-based TD algorithm, which we call ABQ($\zeta$):

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha_t\left(\delta_t\mathbf{e}_t - \gamma\mathbf{e}_t^\top\mathbf{h}_t\left(\bar{\mathbf{x}}_{t+1} - \tilde{\mathbf{x}}_{t+1}\right)\right), \tag{126}$$

$$\mathbf{e}_t = \gamma\nu_{\zeta,t}\pi_t\mathbf{e}_{t-1} + \mathbf{x}_t, \tag{127}$$

$$\mathbf{h}_{t+1} = \mathbf{h}_t + \beta_t\left(\delta_t\mathbf{e}_t - \mathbf{h}_t^\top\mathbf{x}_t\mathbf{x}_t\right). \tag{128}$$