Borrowing From the Future: An Attempt to Address Double Sampling

Yuhua Zhu
Department of Mathematics, Stanford University, Stanford, CA 94305

Lexing Ying
Department of Mathematics and ICME, Stanford University, Stanford, CA 94305

Abstract

For model-free reinforcement learning, the main difficulty of stochastic Bellman residual minimization is the double sampling problem, i.e., while only one single sample for the next state is available in the model-free setting, two independent samples for the next state are required in order to perform unbiased stochastic gradient descent. We propose new algorithms for addressing this problem based on the key idea of borrowing extra randomness from the future. When the transition kernel varies slowly with respect to the state, it is shown that the training trajectory of new algorithms is close to the one of unbiased stochastic gradient descent. We apply the new algorithms to policy evaluation in both tabular and neural network settings to confirm the theoretical findings.

Keywords: Reinforcement learning; Policy evaluation; Double sampling; Bellman residual minimization; Stochastic gradient descent.

1. Introduction

Reinforcement learning (RL) has received a great deal of attention in recent years following the success of AlphaGo and AlphaZero (Silver et al., 2016, 2017). At the heart of RL is the problem of Markov decision process (MDP), i.e., finding the optimal policy that maximizes the return (Sutton and Barto, 2018). As a type of learning with minimal or no supervision, RL is more powerful than the traditional supervised learning and often closer to the natural learning process. On the other hand, as an optimization problem RL is also significantly harder with many practical challenges, such as high dimensional continuous state and action spaces, learning with limited and noisy samples, etc.

Background. This paper considers one of the most basic problems of RL: policy evaluation, or also known as prediction. In model-based RL, especially with small to medium-sized state space, value iteration is commonly used in practice as it guarantees the convergence (Bertsekas and Tsitsiklis, 1996). In model-free RL, the well-known temporal difference (TD) algorithm (Sutton, 1988) converges under the tabular setting or linear approximation. However, the stability and convergence of TD are not guaranteed when nonlinear approximation is used (Boyan and Moore, 1995; Baird, 1995; Tsitsiklis and Van Roy, 1997). With the recent development and empirical successes of deep neural networks (DNNs), it becomes even more important to understand and stabilize nonlinear approximations.

One direction for stabilizing the nonlinear approximation is to formulate the policy evaluation as a minimization problem rather than a fixed-point iteration; one such example is the mean-squared Bellman residual minimization (BRM), sometimes also called as Bellman error minimization in the literature. Unfortunately, BRM suffers from the so-called double sampling problem, i.e., at a
given state, two independent samples for the next state are required in order to perform unbiased stochastic gradient descent (SGD). Such a requirement is often hard to fulfill in a model-free setting with large or even infinite state space. Although several methods have been proposed over the years to circumvent this issue (Baird, 1995; Bhatnagar et al., 2009; Maei et al., 2010; Sutton et al., 2008, 2009; Dai et al., 2017; Wang et al., 2017, 2016; Liu et al., 2015), they typically come with increased complexities and/or more tuning parameters.

Contributions. In this paper, we revisit the Bellman residual minimization and develop two algorithms to alleviate the double sampling problem. The key idea of the new algorithms is to borrow extra randomness from the future. When the transition kernel varies slowly with respect to the state, we show that the training trajectories of the proposed algorithms are statistically close to the one of the unbiased SGD. The proposed algorithms are applied to the prediction problem (i.e., policy evaluation) in both the tabular and neural network settings to confirm the theoretical findings. Though the discussion here focuses on policy evaluation, the same techniques can be extended to Q-Learning (Watkins, 1989) or value iteration.

Organization. The rest of this paper is organized as follows. Section 2 introduces the new algorithms for the continuous state space setting, while Section 3 addresses the discrete state space case. Section 4 bounds the errors between the new algorithm and the accurate but unrealistic uncorrelated sampling algorithm. Numerical results are given in Section 5 to confirm the theoretical findings and demonstrate the efficiency of the new algorithms.

2. Continuous State Space

2.1. Model and key idea

Consider a discrete-time Markov decision process (MDP) with continuous state space. Throughout this section, the state space $S \subset \mathbb{R}^{d_s}$ is a compact set. Since we consider here the prediction problem, the policy is considered to be fixed. We assume that the one-step transition $P(s, s')$ is prescribed by an unknown drift $\alpha(\cdot)$ and an unknown diffusion $\sigma(\cdot)$,

$$s_{m+1} = s_m + \alpha(s_m) \epsilon + \sigma(s_m) \sqrt{\epsilon} Z_m, \quad Z_m \sim \text{Normal}(0, I_{d_s \times d_s}), \quad (2.1)$$

and when the state reaches the boundary, it follows a prescribed boundary condition. Here, we choose to work with a stochastic differential equation (SDE) setting in order to simplify the presentation of the algorithms and the theorems. The scalings $\epsilon$ and $\sqrt{\epsilon}$ of the drift and the noise terms correspond to discretizing the SDE with time step $\epsilon$. However, both the algorithms and theorems can be extended to more general cases. The real-valued immediate reward is denoted by $r(s, s')$ for $s, s' \in S$ and the discount factor $\gamma$ is in $(0, 1)$. With these notations ready, the value function $V(s)$ is the expected discounted return if the policy is followed from state $s$,

$$V(s) = \mathbb{E} \left[ \sum_{m=0}^{\infty} \gamma^m r(s_m, s_{m+1}) | s_0 = s \right]. \quad (2.2)$$

Let $R(s) = \mathbb{E}[r(s_m, s_{m+1}) | s_m = s]$ be the immediate reward under the fixed policy and $T$ be the Bellman operator defined as

$$TV(s) = R(s) + \gamma \mathbb{E}[V(s_{m+1}) | s_m = s]. \quad (2.3)$$
The value function \( V(s) \) is the fixed point of the operator \( \mathbb{T} \).

Let us consider approximating the value function in a parameterized form \( V(s, \theta) \) in model-free RL. Here, the value function approximation can either be linear or nonlinear with respect to the parameter \( \theta \). One way for computing the optimal parameter \( \theta^* \) is to perform gradient descent to the so-called mean-square Bellman residual

\[
\min_{\theta} \mathbb{E} \left( \mathbb{E} [R(s_m) + \gamma V(s_{m+1}; \theta) - V(s_m; \theta)]^2 \right). \tag{2.4}
\]

This approach is thus called the Bellman residual minimization (BRM). The stochastic gradient descent of BRM is based on an unbiased gradient estimation of the objective function (2.4), which requires two independent transitions from \( s_m \) to \( s_{m+1} \). However, for model-free RL, one does not know explicitly how the environment interacts with the agent. That is to say, besides observing the trajectory \( \{s_m\}_{m=0}^T \) of the agent under the given policy, one cannot generate \( s_{m+1} \) from \( s_m \) because one does not know the drift and diffusion in (2.1) explicitly. Since the trajectory \( \{s_m\}_{m=0}^T \) provides only one simulation from \( s_m \) to \( s_{m+1} \), there is no direct means to generate the second copy. This is the so-called double sampling issue.

In what follows, we propose an algorithm to alleviate the double sampling issue. Instead of requiring a new copy of \( s_{m+1}' \) from \( s_m \), one instead uses the difference between \( s_{m+2} \) and \( s_{m+1} \) to approximate

\[
s_{m+1}' \approx s_m + (s_{m+2} - s_{m+1}).
\]

When the derivatives of drift and diffusion terms are under control, the difference between \( \Delta s_m \) and \( \Delta s_{m+1} \) is small, which makes the new \( s'_{m+1} \) statistically close to the distribution of the true next state.

Before diving into the algorithmic details, let us first summarize our main findings. When the derivatives of the implicit drift and diffusion terms are small, we are able to show theoretically:

- The difference between the biased objective function in the new algorithm and the true objective function is only \( O(\epsilon^2) \) (Lemma 1);
- The equilibrium distribution of \( \theta \) of the new algorithm differs from the one of the unbiased SGD within an error of order \( O(\frac{\epsilon^2}{\eta}) \) (Theorem 3);
- The evolution of \( \theta \) of the new algorithm differs statistically from the unbiased SGD only within an error of \( O(1 + \frac{\epsilon^2}{\eta})O(\epsilon^2) \) (Theorem 4).

Here \( \eta \) is the ratio of the learning rate over the batch size. Note that in order to have the error under control, \( \eta \) cannot be too small. Intuitively, this is because: the algorithm actually minimizes a slightly biased objective function. If the optimization is done without any randomness, the solution will have little overlap with the exact one.

2.2. Algorithms

Let us write the BRM (2.4) in an abstract form,

\[
\theta^* = \min_{\theta \in \Omega} J(\theta), \quad J(\theta) := \mathbb{E} \left[ \frac{1}{2} \mathbb{E} \left[ f(s_m, s_{m+1}; \theta) | s_m \right]^2 \right], \tag{2.5}
\]
where $\Omega \subset \mathbb{R}^d$ is a compact domain and

$$f(s_m, s_{m+1}; \theta) = R(s_m) + \gamma V(s_{m+1}; \theta) - V(s_m; \theta)$$

is the Bellman residual (sometimes also called Bellman error in the literature). Note that, when $V$ is approximated by a neural network with standard activation functions, the boundedness of $\theta$ and $s_m$ implies that $V$ is also bounded. Following (2.1), define

$$\Delta s_m := s_{m+1} - s_m = a(s_m)\epsilon + \sigma(s_m)\sqrt{\epsilon} Z_m. \quad (2.6)$$

Suppose now that functions $\alpha(s)$ and $\sigma(s)$ were known explicitly. Then given an observed path $\{s_m\}_{m=0}^T$, the SGD can update the parameter $\theta$ as follows

**Algorithm 1 [Uncorrelated sampling]** Given a trajectory $\{s_m\}_{m=0}^T$, at step $k$, randomly select $M$ elements from $\{0, \cdots, T\}$ to form the index subset $B_k$, generate a new $s'_{m+1}$ from $s_m$ according to (2.1), then

$$\theta_{k+1} = \theta_k - \frac{\tau}{M} \sum_{m \in B_k} f(s_m, s_{m+1}; \theta_k) \nabla f(s_m, s'_{m+1}; \theta_k),$$

where $\tau$ is the learning rate and $M$ is the batch size.

However, as we pointed out already, generating another $s'_{m+1}$ is unrealistic as $\alpha(s)$ and $\sigma(s)$ are unknown in the model-free setting. Instead, the following double sampling algorithm is sometimes used instead.

**Algorithm 2 [Double sampling]** Given a trajectory $\{s_m\}_{m=0}^T$, at step $k$, $\tau, B_k, M$ are the same as in Algorithm 1,

$$\theta_{k+1} = \theta_k - \frac{\tau}{M} \sum_{m \in B_k} f(s_m, s_{m+1}; \theta_k) \nabla f(s_m, s_{m+1}; \theta_k).$$

Note that the gradient in the above algorithm is not an unbiased gradient of the objective function in (2.5). In fact, it is an unbiased gradient of $E \left[ \frac{1}{2} E \left[ f(s_m, s_{m+1}; \theta)^2 \right] | s_m \right]$. We shall see in Section 5 that this algorithm fails to identify the true solution $\theta^*$ even if the underlying drift and diffusion terms are smooth.

**Borrow from the future.** Below we propose two algorithms that approximate the minimizer efficiently when the underlying drift term $\alpha(s)$ and diffusion term $\sigma(s)$ change smoothly. Instead of minimizing $J(\theta)$, the first algorithm minimizes $\hat{J}(\theta)$,

$$\min_{\theta \in \Omega} \hat{J}(\theta), \quad \hat{J}(\theta) := \frac{1}{2} E \left[ f(s_m, s_{m+1}; \theta) | s_m \right] E \left[ f(s_m, s_m + \Delta s_{m+1}; \theta) | s_m \right] \quad (2.7)$$

where $\Delta s_{m+1}$ is defined as (2.6). The main idea is to borrow from the future: approximating $s'_{m+1} = s_m + \Delta s_m$ in Algorithm 1 with $s'_{m+1} \approx s_m + \Delta s_{m+1}$, i.e., creating another simulation of $s_m \rightarrow s_{m+1}$ by borrowing from the future step $s_{m+1} \rightarrow s_{m+2}$. When $\epsilon$ is small, and the change of the drift and the diffusion are small as well, we expect the approximation should be close to the unbiased gradient. Due to the independence between $\Delta s_m$ and $\Delta s_{m+1}$, we in fact have

$$\hat{J}(\theta) = \frac{1}{2} E \left[ f(s_m, s_{m+1}; \theta) f(s_m, s_m + \Delta s_{m+1}; \theta) \right]. \quad (2.8)$$

From (2.8), one can directly apply SGD algorithm to update the parameter $\theta$ from observed trajectory $\{s_m\}_{m=0}^T$. 


Algorithm 3 [BFF-loss] Given a trajectory $\{s_m\}_{m=0}^T$, at step $k$, $\tau$, $B_k$, $M$ are the same as in Algorithm 1

$$\theta_{k+1} = \theta_k - \frac{\tau}{M} \sum_{m \in B_k} \frac{1}{2} \nabla_\theta [f(s_m, s_{m+1}; \theta_k)f(s_m, s_m + \Delta s_{m+1}; \theta_k)],$$

where $\Delta s_{m+1} = s_{m+2} - s_{m+1}$.

BFF is short for "borrow from the future". An alternative algorithm is applying the same technique directly on the unbiased gradient of the true objective function. We will show in Section 5 that the two new algorithms behave similarly in practice.

Algorithm 4 [BFF-gradient] Given a trajectory $\{s_m\}_{m=0}^T$, at step $k$, $\tau$, $B_k$, $M$ are the same as in Algorithm 1

$$\theta_{k+1} = \theta_k - \frac{\tau}{M} \sum_{m \in B_k} f(s_m, s_{m+1}; \theta_k) \nabla_\theta f(s_m, s_m + \Delta s_{m+1}; \theta_k),$$

where $\Delta s_{m+1} = s_{m+2} - s_{m+1}$.

Compared with Algorithm 1, the above two algorithms are biased SGD methods. However, Section 4 proves that the bias of Algorithm 3 is small. More specifically, we show that the difference of the objective function, the evolution of SGD and the steady state of SGD between Algorithms 1 and 3 are small.

3. Discrete State Space

The discussion in Section 2 is concerned with continuous state space. The same idea can be applied to discrete state space, as long as a smooth parallel transport can be defined on the state space. Here we consider a simple setup with the discrete state space $\mathcal{S} = \{0, 1, \ldots, n-1\}$. Assume that transition matrix $P(s, s')$ of the Markov decision process under the given policy varies slowly in both $s, s'$. The immediate reward function under the current policy is $r \in \mathbb{R}^n$, and the discount rate $\gamma$ is between 0 and 1. Therefore, the value function $V^* \in \mathbb{R}^n$ satisfies the following Bellman equation,

$$V^* = \mathbb{T}(V^*) = r + \gamma PV^*.$$

In this discrete setting, the BRM becomes

$$V^* = \min_{v \in \mathbb{R}^n} \frac{1}{2} \|r + \gamma P v - v\|_\mu^2,$$

where $\mu$ is the stationary distribution of the Markov chain. The gradient of the above objective function can be written as

$$\nabla_v J = (\gamma P - I)^\top \text{diag}(\mu)(r + \gamma P v - v).$$

Since $P$ appears twice in the above formula, in order to obtain an unbiased approximation of the above gradient, we need two simulations from $s_m$ to $s_{m+1}$. Given a trajectory $\{s_m\}_{m=1}^T$, choose a state $s_m = i$ and the next state $s_{m+1} = j$. Assuming that the Markov chain reaches equilibrium,
such a choice is an unbiased estimate for $\text{diag}(\mu)$. If the transition matrix $P$ were known, we could generate a new state $s'_{m+1} = s_{j'}$ and construct an unbiased estimation of $\nabla_v J$: replacing the first and second copies of $P$ in (3.1) with $P_1$ and $P_2$ given as follows:

$$
(P_1)_{il} = \begin{cases} 1, & l = j' \\ 0, & \text{otherwise} \end{cases}, \quad (P_2)_{il} = \begin{cases} 1, & l = j \\ 0, & \text{otherwise} \end{cases}.
$$

Equivalently, the unbiased estimation of the gradient can be written as

$$
(\nabla_v J)_i = -(r_i + \gamma v_j - v_i), \quad (\nabla_v J)_{j'} = \gamma(r_i + \gamma v_j - v_i), \quad (\nabla_v J)_l = 0, \quad \forall l \neq i, j'. \quad (3.2)
$$

However, in the setup of model-free RL, without knowing the transition matrix $P$, one needs to approximate $V^*$ based on the observed trajectory $\{s_m\}_{m=1}^T$ alone. The same technique that $s'_{m+1} \approx s_m + (s_{m+2} - s_{m+1})$ can also be applied here to give rise to the corresponding two new algorithms in the tabular form. The below are pseudocodes for Algorithms 1-4 in the tabular form. $v$ is updated based on an estimation $G$ of the true gradient $\nabla_v J$,

$$
v_{k+1} = v_k - \eta G_{m_k},
$$

where $m_k$ is randomly selected from $\{1, \cdots, T\}$. In the four algorithms listed above, $G_m$ (dropping the $k$ index for notation convenience) is estimated differently as follows.

- **Uncorrelated sampling**: Assume $s_m = i, s_{m+1} = j$ (same for the other three algorithms). Generate a new $s'_{m+1}$ by (5.3) and let $j' = s'_{m+1}$

$$
(G_m)_i = -(r_i + \gamma v_j - v_i), \quad (G_m)_{j'} = \gamma(r_i + \gamma v_j - v_i), \quad (G_m)_l = 0, \quad \forall l \neq i, j. \quad (3.3)
$$

The uncorrelated sampling algorithm gives unbiased estimation of the loss function (3.1). However, it is unrealistic for model-free RL.

- **Double sampling**:

$$
(G_m)_i = -(r_i + \gamma v_j - v_i), \quad (G_m)_j = \gamma(r_i + \gamma v_j - v_i), \quad (G_m)_l = 0, \quad \forall l \neq i, j. \quad (3.4)
$$

- **BFF-gradient**: Let $j' = s_m + (s_{m+2} - s_{m+1})$.

$$
(G_m)_i = -(r_i + \gamma v_j - v_i), \quad (G_m)_{j'} = \gamma(r_i + \gamma v_j - v_i), \quad (G_m)_l = 0, \quad \forall l \neq i, j. \quad (3.5)
$$

- **BFF-loss**: Let $j' = s_m + (s_{m+2} - s_{m+1})$.

$$
(G_m)_i = -\frac{1}{2}(r_i + \gamma v_j - v_i) - \frac{1}{2}(r_i + \gamma v_{j'} - v_i), \quad (G_m)_{j'} = \frac{\gamma}{2}(r_i + \gamma v_j - v_i),
$$

$$
(G_m)_j = \frac{\gamma}{2}(r_i + \gamma v_{j'} - v_i), \quad (G_m)_l = 0, \quad \forall l \neq i, j, j'. \quad (3.6)
$$

4. **Error estimates**

The aim of this section is to prove that Algorithm 3 (BFF-loss) is statistically close to Algorithm 1 (uncorrelated sampling).
4.1. Difference between objective functions

Let us introduce

\[ \tilde{J}(\theta) := \hat{J}(\theta) - J(\theta). \]  

(4.1)

Notice that \( \hat{J}(\theta) = \mathbb{E} \tilde{J}(s_m; \theta) \) with

\[ \tilde{J}(s_m; \theta) = \mathbb{E} [ f(s_m, s_{m+1}; \theta) | s_m ] \mathbb{E} [ f(s_m, s_m + \Delta s_{m+1}; \theta) - f(s_m, s_m + \Delta s_m; \theta) | s_m ] . \]  

(4.2)

The following lemma shows that if the values and derivatives of the drift, diffusion, and nonlinear approximation are bounded, then the difference between the two objective function \( J(\theta) \) and \( \hat{J}(\theta) \) is less than \( C\epsilon^2 \), with the constant depending only on the size of \( \alpha, \sigma, f \) and their derivatives until second order.

**Lemma 1** For \( \tilde{J}, \hat{J} \) defined in (4.1), (4.2), if \( \| \alpha^{(k)}(\cdot) \|_{L^\infty}, 0 \leq k \leq 3, \| \sigma^{(l)}(\cdot) \|_{L^\infty}, 0 \leq l \leq 4 \) are bounded, \( \| \partial_s^i f(s_1, s_2; \theta) \|_{L^\infty_{s_1, s_2}}, 0 \leq i \leq 5 \), are also uniformly bounded for any \( \theta \), then for all \( \theta \),

\[ \| \tilde{J}(s, \theta) \|_{L^\infty} \leq C \epsilon^2 + o(\epsilon^2), \]

\[ \hat{J}(\theta) \leq C \epsilon^2 + o(\epsilon^2), \]  

(4.3)

for some constant \( C \) depending on \( \| \alpha^{(i)}(\cdot) \|_{L^\infty}, \| \sigma^{(l)}(\cdot) \|_{L^\infty}, \| \partial_s^i f(s_1, s_2; \theta) \|_{L^\infty_{s_1, s_2}}, 0 \leq i \leq 2 \).

The boundedness of the residual \( f \) is followed by the boundedness of \( R \) and \( V \). Since we assume that the state space \( S \) and the parameter space \( \Omega \) are both compact, for parametric value approximation, such as Neural Network, it is natural to assume \( R, V \) are bounded.

**Proof** Let

\[ \delta(s_m, \theta) = \mathbb{E} [ f(s_m, s_m + \Delta s_{m+1}; \theta) - f(s_m, s_m + \Delta s_m; \theta) | s_m ] , \]

then

\[ \tilde{J}(s_m, \theta) = \mathbb{E} [ f(s_m, s_{m+1}; \theta) | s_m ] \delta(s_m, \theta); \quad \tilde{J}(\theta) = \frac{1}{2} \mathbb{E} \tilde{J}(s_m, \theta). \]  

(4.4)

We first estimate the term \( \delta(s_m, \theta) \). By Taylor expansion,

\[ f(s_m, s_m + \Delta s_{m+1}; \theta) - f(s_m, s_m + \Delta s_m; \theta) \]

\[ = [f(s_m, s_m + \Delta s_{m+1}; \theta) - f(s_m, s_m; \theta)] - [f(s_m, s_m + \Delta s_m - f(s_m, s_m; \theta)] \]

\[ = \partial^1_{s_2} f(s_m, s_m; \theta) (\Delta s_{m+1} - \Delta s_m) + \frac{1}{2} \partial^2_{s_2} f(s_m, s_m; \theta) (\Delta s_{m+1}^2 - \Delta s_m^2) \]

\[ + \frac{1}{6} \partial^3_{s_2} f(s_m, s_m; \theta) (\Delta s_{m+1}^3 - \Delta s_m^3) + \frac{1}{24} \partial^4_{s_2} f(s_m, s_m; \theta) (\Delta s_{m+1}^4 - \Delta s_m^4) \]

\[ + \frac{1}{120} \partial^5_{s_2} f(s_m, s_m + s' \theta) \Delta s_{m+1}^5 - \partial^5_{s_2} f(s_m, s_m + s'' \theta) \Delta s_m^5 , \]

for some \( s' \in (0, \Delta s_{m+1}), s'' \in (0, \Delta s_m) \). By the definition of \( \Delta s_m \) in (2.6), we have \( \Delta s_m^5, \Delta s_{m+1}^5 \leq o(\epsilon^2) \), so the last term of the above equation is of order \( o(\epsilon^2) \). Therefore, one has,

\[ \delta(s_m, \theta) = \sum_{i=1}^4 \frac{1}{i!} \partial^i_{s_2} f(s_m, s_m; \theta) \mathbb{E} [ \Delta s_{m+1}^i - \Delta s_m^i | s_m ] + o(\epsilon^2). \]  

(4.5)
The Taylor expansion of $\alpha(s_{m+1}), \sigma(s_{m+1})$ can be represented by

\[
\alpha(s_{m+1}) = \alpha(s_m) + \alpha'(s_m)\Delta s_m + \frac{1}{2}\alpha''(s_m)\Delta s_m^2 + o(\epsilon), \\
\sigma(s_{m+1}) = \sigma(s_m) + \sigma'(s_m)\Delta s_m + \frac{1}{2}\sigma''(s_m)\Delta s_m^2 + \frac{1}{6}\sigma'''(s_m)\Delta s_m^3 + o(\epsilon^{3/2}),
\]

which gives,

\[
\Delta s_{m+1} - \Delta s_m = (\alpha(s_{m+1}) - \alpha(s_m))\epsilon + \sigma(s_m)Z_{m+1}\sqrt{\epsilon} - \sigma(s_m)Z_m\sqrt{\epsilon} \\
= \left(\alpha'(s_m)\Delta s_m + \frac{1}{2}\alpha''(s_m)\Delta s_m^2\right)\epsilon + o(\epsilon^2) \\
+ \left(\sigma(s_m) + \sigma'(s_m)\Delta s_m + \frac{1}{2}\sigma''(s_m)\Delta s_m^2 + \frac{1}{6}\sigma'''(s_m)\Delta s_m^3\right)Z_{m+1}\sqrt{\epsilon} - \sigma(s_m)Z_m\sqrt{\epsilon} + o(\epsilon^2).
\]

Since $E[h(s_m)Z_{m+1}|s_m] = E[g(s_m)Z_{m+1}|s_m] = 0$ for any functions $h, g$, the last line of the above equation vanishes after taking conditional expectation on $s_m$. This implies,

\[
E \left[\Delta s_{m+1} - \Delta s_m|s_m\right] = E \left[\left(\alpha'(s_m)\Delta s_m + \frac{1}{2}\alpha''(s_m)\Delta s_m^2\right)\epsilon|s_m\right] + o(\epsilon^2) \\
= E \left[\left(\alpha'(\alpha + \sigma\sqrt{\epsilon}Z_m) + \frac{1}{2}\alpha''(\alpha^2\epsilon^2 + 2\alpha\sigma\epsilon Z_m + \sigma^2\epsilon Z_m^2\right)\epsilon|s_m\right] + o(\epsilon^2) \\
= \alpha'\alpha\epsilon^2 + \frac{1}{2}\alpha''\sigma^2\epsilon^2 + o(\epsilon^2).
\]

Here $\alpha, \alpha', \sigma$ all refers to the function’s value at $s_m$, similar for $\sigma', \sigma'', \partial^i f$. We omit $(s_m)$ when the functions has its value at $s_m$. 

Using (4.6), one can estimate $\Delta s_{m+1}^i$ for $i = 2, 3, 4$ as follows,

\[
\Delta s_{m+1}^2 = \alpha(s_{m+1})^2\epsilon^2 + \sigma(s_{m+1})^2Z_{m+1}^2\epsilon + 2\alpha(s_{m+1})\sigma(s_{m+1})Z_{m+1}\epsilon + o(\epsilon^2) \\
= \alpha'\epsilon^2 + (\alpha^2 + \sigma^2)\Delta s_m^2 + 2\alpha\sigma\Delta s_m + \sigma^2\Delta s_m^2 + o(\epsilon^2); \\
\Delta s_{m+1}^3 = 3\alpha(s_{m+1})\sigma^2(s_{m+1})Z_{m+1}\epsilon^2 + \sigma^3(s_{m+1})Z_{m+1}\epsilon^3 + o(\epsilon^2) \\
= 3\alpha\sigma^2Z_{m+1}\epsilon^2 + \sigma^3(s_{m+1})Z_{m+1}\epsilon^3 + o(\epsilon^2); \\
\Delta s_{m+1}^4 = \sigma^4(s_{m+1})Z_{m+1}\epsilon^2 + o(\epsilon^2) = \sigma^4Z_{m+1}\epsilon^2 + o(\epsilon^2);
\]

Therefore,

\[
E \left[\Delta s_{m+1}^2 - \Delta s_m^2|s_m\right] = E \left[\left((\sigma^2\Delta s_m^2 + 2\sigma\sigma'\Delta s_m + \sigma\sigma''\Delta s_m^2\right)\epsilon|s_m\right] + o(\epsilon^2) \\
= E \left[\left(\alpha'\alpha + \sigma\sigma'\right)(\alpha^2\epsilon^2 + o(\epsilon)) + 2\sigma\sigma'\alpha\sigma\epsilon\right] + o(\epsilon^2) \\
= (\alpha'\alpha + \sigma\sigma')\sigma^2\epsilon^2 + 2\sigma\sigma'\alpha\sigma\epsilon + o(\epsilon^2); \\
E \left[\Delta s_{m+1}^3 - \Delta s_m^3|s_m\right] = o(\epsilon^2); \\
E \left[\Delta s_{m+1}^4 - \Delta s_m^4|s_m\right] = o(\epsilon^2).
\]
Hence, by inserting (4.7) and (4.8) into (4.5) gives,

\[ \delta(\theta) = \left[ \partial_{s_2} f \left( \alpha' \alpha + \frac{1}{2} \alpha'' \sigma^2 \right) + \partial^2_{s_2} f \left( \sigma^2 \sigma^2 + \sigma \sigma'' \sigma^2 + 2 \sigma \sigma' \alpha \right) \right] \epsilon^2 + o(\epsilon^2). \]  

As defined in (4.2) and (4.4), the completion of the proof is followed by,

\[ \tilde{j} = \frac{1}{2} \mathbb{E}[f(s_m, s_{m+1}; \theta)|s_m] \delta \leq C \epsilon^2 + o(\epsilon^2); \]
\[ \tilde{J} = \mathbb{E}\tilde{j} \leq C \epsilon^2 + o(\epsilon^2). \]

### 4.2. Difference in asymptotic regime

In this section and Section 4.3, we assume the optimization region of (2.5) is a bounded connected open subset \( \Omega \) in \( \mathbb{R}^d \). This assumption is to guarantee the Poincare inequality. The updates of the parameter \( \theta \) of \( J(\cdot) \) and \( \hat{\theta} \) of \( \tilde{J}(\cdot) \) by SGD according to Algorithms 1 and 3 can be approximated by stochastic differential equations (SDEs) with \( \eta = \frac{\tau}{M} \) (Li et al., 2017; Hu et al., 2017)

\[ d\theta_t = -\nabla J(\theta_t) dt + \sqrt{\eta} \Sigma(\theta_t) dB_t; \]
\[ d\hat{\theta}_t = -\nabla \left( J(\hat{\theta}_t) + \tilde{J}(\hat{\theta}_t) \right) dt + \sqrt{\eta} \left( \Sigma(\hat{\theta}_t) + \tilde{\Sigma}(\hat{\theta}_t) \right)^{1/2} dB_t, \]

where
\[ \Sigma(\theta_t) = \mathbb{V} \left[ \frac{1}{2} \mathbb{E} [f(s_m, s_{m+1}; \theta)|s_m]^2 \right]; \]
\[ \tilde{\Sigma}(\hat{\theta}_t) = \mathbb{V} \left[ \frac{1}{2} \mathbb{E} [f(s_m, s_{m+1}; \theta)|s_m]^2 + \hat{j} \right] - \Sigma(\hat{\theta}). \]

Here \( \mathbb{V} \) represents the variance and \( \hat{j} \) is defined in (4.2).

Therefore, the corresponding probability density functions \( p(t, \theta), \hat{p}(t, \theta) \) of \( \theta_t, \hat{\theta}_t \) can be described by (Pavliotis, 2014)

\[ \partial_t p(t, \theta) = \nabla \cdot \left[ \left( \nabla J \right) p + \frac{\eta}{2} \nabla \cdot \left( \Sigma p \right) \right]; \]  
\[ \partial_t \hat{p}(t, \theta) = \nabla \cdot \left[ \left( \nabla J + \nabla \tilde{J} \right) \hat{p} + \frac{\eta}{2} \nabla \cdot \left( \Sigma + \tilde{\Sigma} \right) \hat{p} \right], \]

with the same initial data \( p(0, \theta) = \hat{p}(0, \theta) \). We use reflecting boundary condition on \( \partial \Omega \),

\[ \left( \nabla J p + \frac{\eta}{2} \nabla \cdot (\Sigma p) \right) \cdot \mathbf{n} \bigg|_{\partial \Omega} = 0, \]
\[ \left( \left( \nabla J + \nabla \tilde{J} \right) \hat{p} + \frac{\eta}{2} \nabla \cdot \left( \Sigma + \tilde{\Sigma} \right) \hat{p} \right) \cdot \mathbf{n} \bigg|_{\partial \Omega} = 0, \]

which means that \( \theta \) will be reflected after hitting the boundary.
In addition, from the estimation we obtained in (4.3), we know that $\nabla \hat{J} \leq O(\epsilon^2)$ and it is easy to see that $\tilde{\Sigma} \leq O(\epsilon^2)$ because

$$\tilde{\Sigma}(\hat{\theta}) = \Sigma(\hat{\theta}) + \nabla [\hat{J}] + \mathbb{E} \left[ \hat{J} \left( \mathbb{E} \left[ f(s_m, s_{m+1}; \theta) \right] s_m \right)^2 \right] - \mathbb{E} \left[ \hat{J} \right] \mathbb{E} \left[ \left( \mathbb{E} \left[ f(s_m, s_{m+1}; \theta) s_m \right] \right)^2 \right] - \Sigma(\hat{\theta}) \leq O(\epsilon^4) + C(\hat{\theta})\epsilon^2 \leq C\epsilon^2 + o(\epsilon^2).$$

(4.13)

**Assumption 2** We assume the loss function $J(\theta)$ and $\hat{J}(\theta)$ both satisfy the following:

- $\int e^{-J(\theta)} d\theta \leq \infty$, and $\int e^{-\hat{J}(\theta)} d\theta \leq \infty$.

- The Frobenius norm of $G = \left\| H(J) \right\|_{L_\theta^\infty}$ and $\hat{G} = \left\| H(\hat{J}) \right\|_{L_\hat{\theta}^\infty}$ are bounded by a constant $M$, where $H$ represents the Hessian and $L_\theta^\infty$ is taken element-wisely to the matrix.

The first assumption is to guarantee that the steady state is well defined. The second assumption is used to prove the boundedness of $\nabla \hat{p}$. 

**Theorem 3** Assume $\Sigma \sim O(1)$, $\tilde{\Sigma}$ are both constants, then there exists steady states $p^\infty$, $\hat{p}^\infty$ for (4.10), (4.11)

$$p^\infty(\theta) = \frac{1}{Z} e^{-\beta J(\theta)}, \quad \hat{p}^\infty(\theta) = \frac{1}{\hat{Z}} e^{-\hat{\beta}(J(\theta)+\hat{J}(\theta))}, \quad \beta = \frac{2}{\eta\Sigma}, \quad \hat{\beta} = \frac{2}{\eta(\Sigma + \tilde{\Sigma})},$$

where $Z = \int e^{-\frac{2J}{\eta\Sigma}} d\theta$, $\hat{Z} = \int e^{-\frac{2(J+\hat{J})}{\eta(\Sigma + \tilde{\Sigma})}} d\theta$ are normalized constants. In addition,

$$\left\| \frac{\hat{p}^\infty}{p^\infty} \right\|_{L^\infty} \leq 1 + O \left( \frac{\epsilon^2}{\eta} \right).$$

Theorem 3 implies the following:

- If the probability of the unbiased SGD (Algorithm 1) converging to the optimal $\theta^*$ is $p$, then the probability of Algorithm 3 is $\left( 1 + O \left( \frac{\epsilon^2}{\eta} \right) \right) p$.

- In order to make Algorithm 3 behaves similarly to the unbiased SGD, we have to let $\frac{\epsilon^2}{\eta}$ be small, which means we require $\epsilon$ to be small, but $\eta$ to be larger than $\epsilon^2$. This makes sense because we are minimizing a biased objective function, so if we do the biased SGD too carefully, it will end up a worse minimizer of the true objective function.

**Proof**

$$\frac{\hat{p}^\infty}{p^\infty} = \frac{Z}{\hat{Z}} e^{-\hat{\beta}(J(\theta)+\hat{J}(\theta))} = \frac{Z}{\hat{Z}} e^{-(\hat{\beta} - \beta)J(\theta)} e^{-\hat{\beta}J(\theta)} = \frac{Z}{\hat{Z}} e^{\frac{\Sigma}{\eta(\Sigma + \tilde{\Sigma})} \frac{2J(\theta)}{\eta(\Sigma + \tilde{\Sigma})}} e^{\frac{\hat{J}(\theta)}{\eta}} \frac{2}{\eta}. $$

By the fact that $\tilde{\Sigma} \leq O(\epsilon^2)$, $\hat{J} \leq O(\epsilon^2)$, we have

$$\left\| \frac{\hat{p}^\infty}{p^\infty} \right\|_{L^\infty} \leq \frac{Z}{\hat{Z}} e^{O \left( \frac{\epsilon^2}{\eta} \right)} \leq \frac{Z}{\hat{Z}} \left( 1 + O \left( \frac{\epsilon^2}{\eta} \right) \right).$$
Similarly, it is easy to see that \( \frac{Z}{\hat{Z}} \sim (1 + O(\epsilon^2/\eta)) \) because,

\[
\frac{Z}{\hat{Z}} = \frac{\int_\Omega e^{-\beta J} d\theta}{\int_\Omega e^{-\beta J} e^{-(\beta - \beta) J} e^{-\beta J} d\theta} = \frac{\int_\Omega e^{-\beta J} d\theta}{\int_\Omega e^{-\beta J} \left(1 + O\left(\frac{\epsilon^2}{\eta}\right)\right)^2 d\theta} = 1 + O\left(\frac{\epsilon^2}{\eta}\right).
\]

Therefore,

\[
\left\| \frac{\hat{p}^\infty}{p^\infty} \right\|_{L^\infty} \leq \left(1 + O\left(\frac{\epsilon^2}{\eta}\right)\right)^2 \leq 1 + O\left(\frac{\epsilon^2}{\eta}\right).
\]

4.3. Difference in finite time regime

Theorem 3 is about the asymptotic behavior of the algorithm. Now we will study the difference between the two algorithms at a finite time. The following Poincare inequality of the probability measure \( d\mu = p^\infty d\theta \) or \( d\mu = \hat{p}^\infty d\theta \) in a bounded connected domain valid for any \( \int \nabla h d\mu = 0 \),

\[
\int_\Omega |\nabla h|^2 d\mu \geq \lambda \int_\Omega h^2 d\mu,
\]

with a constant \( \lambda \) depending on \( d\mu \) and \( \Omega \). Based on the above Poincare inequality, we can prove the difference between the two algorithms, as shown in the following theorem. The difference is measured in the following norm,

\[
\| h \|^2 = \int_\Omega h^2 \frac{1}{p^\infty} d\theta,
\]

where \( p^\infty \) is defined in Theorem 3.

**Theorem 4**  Under Assumption 2 and the assumptions in Theorem 3, one has,

\[
\| p(t, \theta) - \hat{p}(t, \theta) \|^2 \leq \left(1 + O\left(\frac{\epsilon^2}{\eta}\right)\right) O(\epsilon^2).
\]

Theorem 4 tells us that the evolution of \( \theta \) in Algorithm 3 differs from the unbiased SGD within an error of \( \left(1 + O\left(\frac{\epsilon^2}{\eta}\right)\right) O(\epsilon^2) \).

**Proof**  Letting \( h(t, \theta) = \hat{p}(t, \theta) - p(t, \theta) \) and subtracting (4.10) from (4.11) leads to

\[
\partial_t h = \nabla \cdot \left[ (\nabla J + \nabla \hat{J}) \hat{p} + \frac{1}{\beta} \nabla p \right] - \nabla \cdot \left[ \nabla J p + \frac{1}{\beta} \nabla p \right]
= \nabla \cdot \left[ \nabla J h + \frac{1}{\beta} \nabla h \right] + \nabla \cdot \left[ \nabla \hat{J} \hat{p} + \left(\frac{1}{\beta} - \frac{1}{\beta}\right) \nabla \hat{p} \right]
= \nabla \cdot \left[ \frac{p^\infty}{p^\infty} \nabla \left(\frac{h}{p^\infty}\right) \right] + \nabla \cdot \left[ \nabla \hat{J} \hat{p} + \left(\frac{1}{\beta} - \frac{1}{\beta}\right) \nabla \hat{p} \right].
\]
Multiply $\frac{h}{p^\infty}$ to the above equation, then integrate it over $\theta$, one has,

$$
\frac{1}{2} \partial_t \|h\|^2_\star = \frac{1}{p^\infty} \left( (\nabla J + \nabla \hat{J}) \hat{p} + \frac{1}{\beta} \nabla \hat{p} \right) \cdot \mathbf{n} \left|_{\partial \Omega} \right. - \frac{h}{p^\infty} \left( \nabla J \hat{p} + \frac{1}{\beta} \nabla p \right) \cdot \mathbf{n} \left|_{\partial \Omega} \right.
- \int \left[ \nabla \left( \frac{h}{p^\infty} \right) \right]^2 p^\infty d\theta - \int \left[ \nabla \hat{J} \hat{p} + \left( \frac{1}{\beta} - \frac{1}{\beta} \right) \nabla \hat{p} \right] \cdot \nabla \left( \frac{h}{p^\infty} \right) d\theta.
$$

The first two terms on the RHS vanishes because of the reflecting boundary condition (4.12). Since $\nabla J, \Sigma \leq O(\epsilon^2)$ have been shown in Lemma 1 and (4.13), this leads to the two coefficients of the last term can be bounded by $\|\nabla \hat{J}\|_{L^\infty} \leq C_1 \epsilon^2$, $\frac{1}{\beta} - \frac{1}{\beta} = \eta \Sigma / 2 \leq C_2 \eta \epsilon^2$. Therefore, applying Young's inequality to the last term gives,

$$
\int \left[ \nabla \hat{J} \hat{p} + \left( \frac{1}{\beta} - \frac{1}{\beta} \right) \nabla \hat{p} \right] \cdot \nabla \left( \frac{h}{p^\infty} \right) d\theta
\leq \|\nabla \hat{J}\|_{L^\infty} \int \hat{p} \cdot \nabla \left( \frac{h}{p^\infty} \right) d\theta + \left( \frac{1}{\beta} - \frac{1}{\beta} \right) \int \left| \nabla \hat{p} \cdot \nabla \left( \frac{h}{p^\infty} \right) \right| d\theta
\leq \frac{1}{2} C_1 \epsilon^2 \left( \|\nabla \hat{p}\|^2_\star + \int \left[ \nabla \left( \frac{h}{p^\infty} \right) \right]^2 p^\infty d\theta \right) + \frac{1}{2} C_2 \eta \epsilon^2 \left( \|\nabla \hat{p}\|^2_\star + \int \left[ \nabla \left( \frac{h}{p^\infty} \right) \right]^2 p^\infty d\theta \right).
$$

The third term can be bounded according to the Poincare Inequality (4.14), thus one has

$$
\frac{1}{2} \partial_t \|h\|^2_\star
\leq -\frac{\lambda}{2} \|h\|^2_\star - \frac{1}{2} \int \left[ \nabla \left( \frac{h}{p^\infty} \right) \right]^2 p^\infty d\theta + \frac{\epsilon^2}{2} \left( C_1 \|\hat{p}\|^2_\star + C_2 \eta \|\nabla \hat{p}\|^2_\star + (C_1 + C_2 \eta) \int \left[ \nabla \left( \frac{h}{p^\infty} \right) \right]^2 p^\infty d\theta \right)
\leq -\frac{\lambda}{2} \|h\|^2_\star + \frac{\epsilon^2}{2} \left( C_1 \|\hat{p}\|^2_\star + \eta C_2 \|\nabla \hat{p}\|^2_\star \right).
$$

(4.17)

Since we only consider the case when $\epsilon \ll 1$, so the coefficient of $\int \left[ \nabla \left( \frac{h}{p^\infty} \right) \right]^2 p^\infty d\theta, -(1 - e^2 (C_1 + C_2 \eta)) / 2$, is always negative, which gives the last inequality of the above estimates.

Therefore, as long as $\|\hat{p}\|^2_\star, \|\nabla \hat{p}\|^2_\star$ are bounded, we can bound $\|h\|^2_\star$ from (4.17). We prove the boundedness of $\|\hat{p}\|^2_\star, \|\nabla \hat{p}\|^2_\star$ in Lemma 5 and Lemma 6 in Appendix A and B. Then, we can bound the last term of (4.17) by

$$
\frac{\epsilon^2}{2} \left( C_1 \|\hat{p}\|^2_\star + \eta C_2 \|\nabla \hat{p}\|^2_\star \right) \leq \frac{C}{2} \left\| \frac{\hat{p}^\infty}{p^\infty} \right\|_{L^\infty} \epsilon^2,
$$

for some constant $C$. Hence from (4.17), we have,

$$
\partial_t \left( e^{\lambda t} \|h\|^2_\star \right) \leq e^{\lambda t} \left( C \left\| \frac{\hat{p}^\infty}{p^\infty} \right\|_{L^\infty} \epsilon^2 \right),
$$

$$
e^{\lambda t} \|h\|^2_\star - \|h(0)\|^2_\star \leq \frac{1}{\lambda} (e^{\lambda t} - 1) C \left\| \frac{\hat{p}^\infty}{p^\infty} \right\|_{L^\infty} \epsilon^2.
$$
Since \( p(0, \theta) = \hat{p}(0, \theta) \), \( h(0, \theta) = 0 \). Therefore,

\[
\|h\|_2^2 \leq \frac{1}{\lambda} (1 - e^{-\lambda t}) C \|\hat{p}^\infty\|_{L^\infty} \epsilon^2 \leq \left(1 + O\left(\frac{\epsilon^2}{\eta}\right)\right) O(\epsilon^2).
\]

\[\Box\]

5. Numerical Examples

Several numerical examples are presented here to demonstrate the performance of the proposed algorithms. Recall that the goal of the prediction problem (i.e., policy evaluation) is to approximate \( V(s) \) based on the trajectories.

5.1. Continuous state space

Consider a Markov decision process with a continuous state space \( S = \{s \in (0, 2\pi]\} \). Suppose that the transition probability is prescribed implicitly via the following dynamics

\[
s_{m+1} = s_m + \alpha(s_m) \epsilon + \sigma(s_m) \sqrt{\epsilon} Z_m, \\
\alpha(s) = 2 \sin(s) \cos(s), \quad \sigma(s) = 1 + \cos(s)^2, \quad \epsilon = 0.1.
\]

The immediate reward function is \( R(s) = (\cos(2s) + 1) \) and the discount factor \( \gamma \) is 0.9.

A 3-layer fully connected neural network \( V(s; \theta) \) is used to approximate the value function. The network has two hidden layers with \( \cos \) as its activation function, and each hidden layer contains 50 neurons, i.e.,

\[
V(s; \theta) = V(x; \{w_i, b_i\}_{i=1}^3) = L_{w_3,b_3} \circ \cos \circ L_{w_2,b_2} \circ \cos \circ L_{w_1,b_1}((\cos s, \sin s)), \\
L_{w_i,b_i}(x) = w_i x + b_i, \quad w_i \in \mathbb{R}^{n_{i-1} \times n_i}, \quad b_i \in \mathbb{R}^{n_i}, \quad n_0 = 2, n_1 = n_2 = 50, n_3 = 1.
\]

The optimal \( \theta^* \) is computed with Algorithms 1-4 based on a trajectory \( \{s_m\}_{m=1}^{10^6} \) with

\[
f(s_m, s_{m+1}, \theta) = R(s_m) + \gamma V(s_{m+1}; \theta) - V(s_m; \theta), \quad \tau = 0.1, \quad M = 1000.
\]

In each experiment, the SGD algorithm runs for a single epoch with the same initialization \( \theta_0 \). The error \( e_k \) at each step \( k \) is defined as the squared \( L_2 \) norm \( \|V(\cdot, \theta_k) - V^*\|_2 \). Here the reference solution \( V^*(s) \) is computed by running Algorithm 1 for 10 epochs based a longer trajectory \( \{s_m\}_{m=1}^{10^7} \), with hyper-parameters \( \tau = 0.01 \) and \( M = 1000 \). The left plot of Figure 1 shows the final \( V(s, \theta) \) obtained by four different methods, while the relative error \( \log_{10}(e_k/e_0) \) in the log scale is shown in Figure 1 (right).

Figure 1 shows that the double sampling algorithm introduces a rather large error while the BFF algorithms are much closer to the (unrealistic) uncorrelated sampling algorithm.
5.2. Discrete state space

Consider a Markov decision process with a discrete state space $S = \{i = 0, \cdots, n - 1\}$ with $n = 32$. The transition matrix is the following,

$$
\begin{align*}
P_{i,i+1} &= \frac{1}{2} - 0.2 \sin \frac{2\pi i}{n}, \quad \text{when } i = n - 1, i + 1 = 0; \\
P_{i,i-1} &= \frac{1}{2} + 0.2 \sin \frac{2\pi i}{n}, \quad \text{when } i = 0, i - 1 = n - 1.
\end{align*}
$$

(5.3)

The immediate reward function is $r \in \mathbb{R}^n$ with $r_i = 1 + \cos \frac{2\pi i}{n}$ and the discount rate is $\gamma = 0.9$. The value function $V^* \in \mathbb{R}^n$ satisfies the following Bellman equation,

$$
V^* = T(V^*) = r + \gamma PV^*.
$$

(5.4)

One can solve for the exact value function $V^*$ directly from the above equation. We will test both the neural network approximation and the tabular case for representing the value function.

**Neural network.** We use the same neural network structure as in (5.2) with input $s = \frac{2\pi i}{n}$ for approximating the value function. Based on a trajectory $\{s_m\}_{m=1}^T$, $T = 4 \times 10^6$ simulated from (5.3), we run Algorithms 1-4 to approximate $\theta^*$. Figure 2 shows the result obtained from SGD with a single epoch at batch size $M = 1$ and learning rate $\tau = 5 \times 10^{-4}$. Figure 3 summarizes the result of a 2-epoch of SGD at $M = 1000$ and $\tau = 0.1$. The relative errors $e_k / e_0$, defined as

$$
e_k = \sqrt{\sum_{i=0}^{n-1} \left( V\left( \frac{2\pi i}{n}, \theta_k \right) - V^*_i \right)^2},
$$

are shown in the log scale on the right.

Figures 2 and 3 demonstrate that the performance of the two new algorithms (BFF-loss and BFF-gradient) are very similar. Although they are less accurate compared to the uncorrelated case,
Figure 2: Discrete state space: approximation with a 3-layer neural network with batch size 1.

Figure 3: Discrete state space: approximation with a 3-layer neural network with batch size 1000.
the performance is much better than the double sampling case. In Figure 2, the BFF algorithm exhibit a larger oscillation since the small batch size \((M = 1)\) results in more stochasticity in the training dynamics compared to the large match size \((M = 1000)\).

**Tabular case.** The BFF algorithms proposed in Section 3 are used to approximate \(V^*\) in tabular form. In Figure 4, we choose \(\tau = 0.1\) and run the SGD for 5 epochs. The results demonstrate that the BFF algorithms work much better than the double sampling algorithm. Comparing with Figure 2, we observe that the neural network approximation results in significantly faster error decay than the tabular case.

![Graph of Decay of the relative error in the log scale](image)

- double sampling
- BFF-gradient
- BFF-loss
- uncorrelated sampling (unrealistic)

**Figure 4:** Discrete state space: tabular approximation.

### Acknowledgments

The work of L.Y. and Y.Z. is partially supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Scientific Discovery through Advanced Computing (SciDAC) program. The work of L.Y. is also partially supported by the National Science Foundation under award DMS-1818449. L.Y. thanks Mohammad Ghavamzadeh, Yuandong Tian, and Amy Zhang for helpful discussions.
References

Leemon Baird. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings 1995*, pages 30–37. Elsevier, 1995.

Dimitri P Bertsekas and John N Tsitsiklis. *Neuro-dynamic programming*, volume 5. Athena Scientific Belmont, MA, 1996.

Shalabh Bhatnagar, Doina Precup, David Silver, Richard S Sutton, Hamid R Maei, and Csaba Szepesvári. Convergent temporal-difference learning with arbitrary smooth function approximation. In *Advances in Neural Information Processing Systems*, pages 1204–1212, 2009.

Justin A Boyan and Andrew W Moore. Generalization in reinforcement learning: Safely approximating the value function. In *Advances in neural information processing systems*, pages 369–376, 1995.

Bo Dai, Albert Shaw, Lihong Li, Lin Xiao, Niao He, Zhen Liu, Jianshu Chen, and Le Song. Sbeed: Convergent reinforcement learning with nonlinear function approximation. *arXiv preprint arXiv:1712.10285*, 2017.

Wenqing Hu, Chris Junchi Li, Lei Li, and Jian-Guo Liu. On the diffusion approximation of non-convex stochastic gradient descent. *arXiv preprint arXiv:1705.07562*, 2017.

Qianxiao Li, Cheng Tai, et al. Stochastic modified equations and adaptive stochastic gradient algorithms. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2101–2110. JMLR. org, 2017.

Bo Liu, Ji Liu, Mohammad Ghavamzadeh, Sridhar Mahadevan, and Marek Petrik. Finite-sample analysis of proximal gradient TD algorithms. In *UAI*, pages 504–513. Citeseer, 2015.

Hamid R Maei, Csaba Szepesvári, Shalabh Bhatnagar, and Richard S Sutton. Toward off-policy learning control with function approximation. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 719–726, 2010.

Grigorios A Pavliotis. *Stochastic processes and applications: Diffusion processes, the Fokker-Planck and Langevin equations*, volume 60. Springer, 2014.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Stochastic processes and application-sature*, 529(7587):484, 2016.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

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

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
Richard S Sutton, Csaba Szepesvári, and Hamid Reza Maei. A convergent O(n) algorithm for off-policy temporal-difference learning with linear function approximation. *Advances in neural information processing systems*, 21(21):1609–1616, 2008.

Richard S Sutton, Hamid Reza Maei, Doina Precup, Shalabh Bhatnagar, David Silver, Csaba Szepesvári, and Eric Wiewiora. Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 993–1000. ACM, 2009.

John N Tsitsiklis and Benjamin Van Roy. Analysis of temporal-difference learning with function approximation. In *Advances in neural information processing systems*, pages 1075–1081, 1997.

Mengdi Wang, Ji Liu, and Ethan Fang. Accelerating stochastic composition optimization. In *Advances in Neural Information Processing Systems*, pages 1714–1722, 2016.

Mengdi Wang, Ethan X Fang, and Han Liu. Stochastic compositional gradient descent: algorithms for minimizing compositions of expected-value functions. *Mathematical Programming*, 161(1-2):419–449, 2017.

Christopher John Cornish Hellaby Watkins. Learning from delayed rewards. 1989.

### Appendix A.

**Lemma 5** The solution to (4.11) is bounded

\[
\|\hat{\rho}\|^2 \leq C \left\| \frac{\hat{\rho}^\infty}{\hat{\rho}^\infty} \right\|_{L^\infty},
\]

with some constant \( C \) related to the initial data.

**Proof** We first prove that the difference of \( \hat{\rho} \) and \( \hat{\rho}^\infty \) is exponentially decay. Define norm \( \|g\|^2 \) as follows,

\[
\|g\|^2 = \int g^2 \frac{1}{\hat{\rho}^\infty} d\theta,
\]

where \( \hat{\rho}^\infty \) is defined in Theorem 3. Let \( g = \hat{\rho} - \hat{\rho}^\infty \), then \( g \) satisfies,

\[
\partial_t g = \nabla \cdot \left[ \hat{\rho}^\infty \nabla \left( \frac{g}{\hat{\rho}^\infty} \right) \right]. \tag{A.1}
\]

Multiplying \( \frac{g}{\hat{\rho}^\infty} \), and integrating it over \( \theta \), after integration by parts, one has

\[
\frac{1}{2} \partial_t \|g\|^2 = -\int \hat{\rho}^\infty \left[ \nabla \left( \frac{g}{\hat{\rho}^\infty} \right) \right]^2 d\theta \leq -\lambda \|g\|^2,
\]

where the last inequality follows from the Poincaré inequality (4.14) and the fact that \( \int gd\theta = 1 - 1 = 0 \). Solve the above ODE, one has,

\[
\|g(t)\|^2 \leq e^{-2\lambda t} \|g(0)\|^2 \tag{A.2}
\]
Therefore, one can bound $\hat{p}$ by

\[
\|\hat{p}\|^2 = \left\| \hat{p} \sqrt{\frac{\hat{p}}{p_{\infty}}} \right\|^2 \leq \left\| \frac{\hat{p}}{p_{\infty}} \right\|_{L^\infty} \|\hat{p}\|^2 \leq \left\| \frac{\hat{p}}{p_{\infty}} \right\|_{L^\infty} \left( \|\hat{p} - \hat{p}_{\infty}\|^2 + \|\hat{p}_{\infty}\|^2 \right) \leq \left\| \frac{\hat{p}}{p_{\infty}} \right\|_{L^\infty} \left( e^{-2\lambda t} \|g(0)\|^2 + 1 \right)
\]

(A.3)

Appendix B.

**Lemma 6** The gradient of the solution to (4.11) is bounded

\[
\|\nabla \hat{p}\|^2 \leq C \left\| \frac{\hat{p}}{p_{\infty}} \right\|_{L^\infty},
\]

with some constant $C$ related to the initial data.

**Proof** Similar to (A.3) in the proof of Lemma 5, it is sufficient to prove this Lemma if we get the estimation for $\|\nabla g(t)\|^2$ with $g = \hat{p} - \hat{p}_{\infty}$, because

\[
\|\nabla \hat{p}\|^2 \leq \left\| \frac{\hat{p}}{p_{\infty}} \right\|_{L^\infty} \left( \|\nabla g\|^2 + 1 \right)
\]

(B.1)

First notice that reflecting boundary condition also holds for $\partial_\theta \hat{p}$, that is,

\[
\partial_\theta \left( (\nabla J + \nabla J)\hat{p} + \frac{\eta}{2} \nabla \cdot (\Sigma + \hat{\Sigma} \hat{p}) \right) \cdot n \bigg|_{\partial \Omega} = 0,
\]

Then take $\partial_\theta$ to (A.1), one has,

\[
\partial_t \partial_\theta g = \nabla \cdot \left[ \hat{p}_{\infty} \nabla \left( \frac{\partial_\theta g}{\hat{p}_{\infty}} \right) \right] + \nabla \cdot \left[ \nabla \left( \partial_\theta J \right) g \right].
\]

Multiplying $\frac{\partial_\theta g}{\hat{p}_{\infty}}$ and summing it over $i$, integrating it over $\theta$ gives

\[
\frac{1}{2} \partial_t \|g\|^2 = \partial_\theta_i \left( (\nabla J + \nabla J)g + \frac{\eta}{2} \nabla \cdot (\Sigma + \hat{\Sigma} \hat{p}) \right) \cdot n \bigg|_{\partial \Omega}
\]

\[
- \sum_i \int \hat{p}_{\infty} \left[ \nabla \left( \frac{\partial_\theta_i g}{\hat{p}_{\infty}} \right) \right]^2 d\theta - \sum_i \int \left[ \nabla \left( \partial_\theta J \right) g \right] \cdot \nabla \left( \frac{\partial_\theta_i g}{\hat{p}_{\infty}} \right) d\theta
\]

\[
\leq -\frac{\lambda}{2} \sum_i \|\partial_\theta_i g\|^2 - \frac{1}{2} \sum_i \int \hat{p}_{\infty} \left[ \nabla \left( \frac{\partial_\theta_i g}{\hat{p}_{\infty}} \right) \right]^2 d\theta
\]

\[
+ \frac{1}{2} \sum_i \int \left[ \nabla \left( \partial_\theta J \right) g \right]^2 \frac{1}{\hat{p}_{\infty}} d\theta + \frac{1}{2} \sum_i \int \hat{p}_{\infty} \left[ \nabla \left( \frac{\partial_\theta_i g}{\hat{p}_{\infty}} \right) \right]^2 d\theta
\]

\[
\leq -\frac{\lambda}{2} \|g\|^2 + \frac{1}{2} \sum_i \int \left[ \nabla \left( \partial_\theta J \right) g \right]^2 \frac{1}{\hat{p}_{\infty}} d\theta \leq -\frac{\lambda}{2} \|g\|^2 + \frac{M}{2} \|g\|^2,
\]
where the second assumption in Assumption 2 is applied to obtain the last inequality. Use the estimation of $\|g\|_2^2$ in (A.2) to solve the above ODE,

$$\partial_t \left( e^{\lambda t} \|\nabla g\|_2^2 \right) \leq M e^{\lambda t} \left( e^{-2\lambda t} \|g(0)\|_2^2 \right) \leq M e^{-\lambda t} \|g(0)\|_2^2,$$

$$e^{\lambda t} \|\nabla g\|_2^2 - \|\nabla g(0)\|_2^2 \leq M \|g(0)\|_2^2 \frac{1}{\lambda} (1 - e^{-\lambda t}),$$

$$\|\nabla g\|_2^2 \leq e^{-\lambda t} \left( \|\nabla g(0)\|_2^2 + \frac{M}{\lambda} \|g(0)\|_2^2 \right).$$

Therefore, by (B.1), one has

$$\|\nabla \hat{p}\|_2^2 \leq \left\| \frac{\hat{p}^\infty}{p^\infty} \right\|_{L^\infty} \left( e^{-\lambda t} \left( \|\nabla g(0)\|_2^2 + \frac{M}{\lambda} \|g(0)\|_2^2 \right) + 1 \right).$$

$\blacksquare$