Behaviour Policy Optimization: Provably Lower Variance Return Estimates for Off-Policy Reinforcement Learning

Alexander W. Goodall[1], **Edwin Hamel-De le Court**[1], **Francesco Belardinelli**[1]
[1]Imperial College London
{a.goodall22, e.hamel-de-le-court, francesco.belardinelli}@imperial.ac.uk

Abstract

Many reinforcement learning algorithms, particularly those that rely on return estimates for policy improvement, can suffer from poor sample efficiency and training instability due to high-variance return estimates. In this paper we leverage new results from off-policy evaluation; it has recently been shown that well-designed behaviour policies can be used to collect off-policy data for provably lower variance return estimates. This result is surprising as it means collecting data on-policy is not variance optimal. We extend this key insight to the online reinforcement learning setting, where both policy evaluation and improvement are interleaved to learn optimal policies. Off-policy RL has been well studied (e.g., IMPALA), with correct and truncated importance weighted samples for de-biasing and managing variance appropriately. Generally these approaches are concerned with reconciling data collected from multiple workers in parallel, while the policy is updated asynchronously, mismatch between the workers and policy is corrected in a mathematically sound way. Here we consider only one worker – the behaviour policy, which is used to collect data for policy improvement, with provably lower variance return estimates. In our experiments we extend two policy-gradient methods with this regime, demonstrating better sample efficiency and performance over a diverse set of environments.

Code — https://github.com/sacktock/BPO
Extended version — https://arxiv.org/abs/2511.10843

1 Introduction

Reinforcement learning (RL) (Sutton and Barto 2018) has seen notable success in a diverse range of sequential decision making tasks, including advanced control systems (Schulman et al. 2017), high dimensional and complex games (e.g., Atari, StarCraft II, Go) (Mnih et al. 2015; Vinyals et al. 2019; Silver et al. 2017), and fine-tuning for Large Reasoning Models (LRM) (Guo et al. 2025). Learning optimal policies in such scenarios might require millions, even billions of agent-environment interactions (Mnih et al. 2015, 2016; Jaderberg et al. 2016), which is often slow and expensive in many scenarios (Zhang 2024). Off-policy RL (Wang et al. 2017; Espeholt et al. 2018) aims to improve sample efficiency by reusing past experiences or data generated by a different behaviour policy, rather than following the current policy. However, off-policy methods may introduce new challenges – the mismatch between the behaviour policy and the target policy can lead to high-variance return estimates and unstable learning (Agarwal et al. 2017). Even naïve off-policy correction with importance sampling (IS) can suffer from variance explosion when the behaviour policy and target policy differ (Precup, Sutton, and Singh 2000).

An alternative paradigm is offline RL (Ernst, Geurts, and Wehenkel 2005; Fujimoto, Meger, and Precup 2019; Levine et al. 2020), which offers a promising direction for learning optimal policies from limited data without having to interact online. However, many practitioners still deploy their policies online, to evaluate the final performance or tune hyperparameters (Fu et al. 2021; Schrittwieser et al. 2021; Mathieu et al. 2023), essentially violating the core assumption of this paradigm. Thus, off-policy evaluation is a crucial challenge in offline RL (Zhang, Liu, and Whiteson 2020; Liu and Zhang 2024), where the performance of a policy must be evaluated based on a fixed offline dataset. The Offline Data Informed (ODI) algorithm proposed by Liu and Zhang (2024), offers a surprising new result – well-designed behaviour policies can provide provably lower-variance return estimates for more (sample) efficient policy evaluation.

Contributions. We propose behaviour policy optimization (BPO), a new variance reduction technique for online RL, using off-policy data collected by a well-designed behaviour policy for provably lower-variance return estimates. For designing the behaviour policy we apply new results from off-policy evaluation in (Liu and Zhang 2024) to the (discounted) online RL setting. We note that this extension is non-trivial and comes with additional technical challenges, e.g., non-stationarity of the target policy. Furthermore, the full undiscounted Monte Carlo return typically considered in off-policy evaluation is usually a bad choice for policy improvement in online RL (Williams 1992; Mnih et al. 2016), thus we introduce the truncated IS weighted TD(λ) returns for estimating the target policy returns, we provide a proof of variance reduction and unbiasedness for this new estimator.

Practically, we build upon and extend two established policy-gradient algorithms: REINFORCE (Williams 1992) and Proximal Policy Optimization (PPO) (Schulman et al.

2017); both of which use return estimates in some way for policy improvement. Rather than collecting data on-policy, we concurrently optimize a behaviour policy used to collect off-policy data for training the target policy. The behaviour policy is optimized using a different loss function that includes Q-value estimates of the target policy, thus in addition to the behaviour policy, we train two additional Q-networks with Fitted Q-Evaluation (FQE) (Le, Voloshin, and Yue 2019). Naturally, this increases the complexity of the full algorithm, and introduces additional challenges with hyperparameter tuning. However, for both REINFORCE and PPO we demonstrate no-worse and often better sample efficiency and final performance with respect to episode returns. Finally, we note that our methodology is not limited to discrete actions, we design an alternative loss function for optimizing the behaviour policy with continuous actions, which has a sound theoretical justification.

2 Related Work

Variance Reduction and Off-policy RL. Variance reduction techniques have long been studied in RL to improve learning efficiency and evaluation accuracy. In policy gradient methods, for example, using a baseline or advantage function to reduce the variance of gradient estimates is standard practice (Williams 1992; Sutton and Barto 2018). Generalized Advantage Estimation (Schulman et al. 2018) further trades off bias and variance by averaging multi-step returns, greatly stabilizing on-policy learning. Off-policy RL underlines many significant breakthroughs in RL, e.g., training from experience replay (Mnih et al. 2015), scalable and parallelize training, e.g., IMPALA (Espeholt et al. 2018) and (offline) AlphaStar (Mathieu et al. 2023). If the offline dataset has good coverage, optimal policies can be learnt without having to collect data online, which is often more expensive (Zhang 2024). Importance sampling (IS) is the classical technique for correcting policy mismatch (Precup, Sutton, and Singh 2000), although unbounded importance weights can lead to high-variance estimates (Precup, Sutton, and Singh 2000). To combat this issue Wang et al. (2017) introduced an off-policy actor-critic with truncated importance weights (ACER), Espeholt et al. (2018) also showed that truncating importance weights for V-trace (IMPALA) reduced variance and yields more stable learning in deep RL, although IMPALA is orthogonal to our research here as it is used to correct for off-policy data collected by multiple workers in parallel with asynchronous policy updates. Furthermore, Retrace(λ) (Munos et al. 2016) is a alternative returns estimator that truncates importance weights to 1, incorporating TD(λ) for bias-variance trade-off, obtaining lower-variance updates (and unbiasedness under certain conditions). These approaches generally acknowledge that some bias is acceptable if it dramatically lowers variance and prevents explosive updates. Our work, focused on an orthogonal direction: we actively optimize a behaviour policy that makes the importance weights naturally well-behaved and the data collected most useful without necessarily introducing bias. However, in practice some truncation (bias) is used, although it is typically less aggressive than in related works (Munos et al. 2016; Espeholt et al. 2018)

Off-policy evaluation. In off-policy evaluation the per-decision IS (PDIS) (Precup, Sutton, and Singh 2000) returns estimator, (corrected Monte Carlo returns), is commonly used as it is unbiased for the target policy. Unfortunately, the PDIS estimator suffers from high-variance, when the offline data and target policy are poorly aligned, motivating alternatives such as model-based evaluation (Mukherjee, Hanna, and Nowak 2022) and value-function approximation, e.g., via FQE (Le, Voloshin, and Yue 2019). Recent advances in off-policy evaluation include the design of variance-reducing behaviour policies for collecting offline data for off-policy evaluation. Several prior works have explicitly tackled this problem, Hanna et al. (2017) formulated this as a gradient-based search for an optimal behaviour policy to minimize the variance of the IS estimator. They demonstrated improvements in data efficiency, although their approach was limited to (trajectory-level) IS, which is less general than the PDIS estimator and not well-suited to online RL. Mukherjee, Hanna, and Nowak (2022) derived a variance-optimal behaviour policy for PDIS, but only under the restrictive assumption of tree-structured MDPs and an accurate estimate the transition dynamics (model-based). Robust On-Policy Sampling (ROS) (Zhong et al. 2022) re-weights an existing behaviour policy toward under-represented states to reduce variance. However, ROS assumes the offline data consists of complete trajectories generated by known policies, and it forgoes IS corrections entirely. In contrast, we leverage ODI (Liu and Zhang 2024), which is a state-of-the-art method for behaviour policy design that operates in general MDPs and with no assumptions on underlying dynamics or data completeness. ODI can leverage (possibly incomplete) arbitrary offline data, and with correct IS weights preserves unbiasedness and comes with theoretical guarantees of variance reduction. This makes ODI much more practical for online RL where we don't necessarily want to wait for complete trajectories before updating our policy. However, all of these methods including ODI, are only concerned with estimating the full undiscounted Monte Carlo returns, which can exhibit high-variance and lead to unstable policy updates in online RL (Williams 1992), for a more practical implementation we consider an alternative estimator based on TD(λ). Additional challenges arise with online RL, e.g., the underlying distribution generating the "offline" dataset is non-stationary, including the policy, which is constantly updated. Preventing over-fitting to past data and quick adaptation to policy updates is also crucial for optimizing the the behaviour policy and Q networks.

3 Background

In this section present the necessary background material, we start with the preliminaries on RL, followed by off-policy corrections with IS ratios. We then describe how importance sampling can be used for variance reduction, and we summarize how variance optimal sampling distributions or behaviour policies can be designed for off-policy evaluation.

3.1 Preliminaries for RL

We model the environment as a finite horizon discrete time Markov decision process (MDP) (Puterman 2014), defined as a tuple $\mathcal{M} := \langle \mathcal{S}, \mathcal{A}, r, p, p_0 \rangle$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, $p : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the probabilistic transition function, and $p_0 : \mathcal{S} \to [0, 1]$ is the initial state distribution.

In this paper we consider with the infinite horizon discounted setting, the discount factor $\gamma \in [0, 1)$ trades-off the relative weighting of short- and long-term rewards.

At time step t, an action A_t is sampled according to the agent's policy $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$ which defines a probability distribution over actions from the current state S_t. The successor state S_{t+1} is then sampled from $p(\cdot|S_t, A_t)$. Each episode generates a sequence of states, actions and rewards $S_0 A_0 R_1 S_1 A_1 R_2 S_2 \ldots$. The discounted return, which defines the accumulated reward from timestep t onwards, is given by $G_t := \sum_{k=t}^{\infty} \gamma^{k-t} R_{k+1}$. The state- and state-action value functions for the policy π are defined as, $v_\pi(s) := \mathbb{E}_\pi[G_t \mid S_t = s]$ and $q_\pi(s, a) := \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a]$ respectively.

In this paper we consider the total discounted return $J(\pi) := \mathbb{E}_{S_0 \sim p_0}[v_\pi(S_0)]$ as the performance metric with which to evaluate and optimize the policy. The optimal policy is defined as $\pi^* := \arg\sup_\pi J(\pi)$. The most straightforward way to estimate $J(\pi)$ (policy evaluation) is to sample directly with the policy π online in the environment. There are many sophisticated algorithms for updating the policy such that $J(\pi)$ is maximized (policy improvement), e.g., value-based (Mnih et al. 2015), policy-based (Haarnoja et al. 2018) and policy-gradient methods (Mnih et al. 2016; Schulman et al. 2017). In this paper, we are focused on improving the policy evaluation phase, whereby a well-designed behaviour policy μ is used to collect off-policy data for provably lower-variance return estimates.

3.2 Off-policy Corrections in RL

The goal in off-policy RL is to use trajectories (sequences) generated by an arbitrary behaviour policy μ, for (optionally) learning the value function v_π of the target policy π and optimizing the target policy π to maximize the objective function $J(\pi)$. Since the behaviour policy μ and the target policy π may differ the return estimates must be adjusted to account for the policy mismatch.

Importance Sampling in RL. Let $\rho_t := \frac{\pi(A_t|S_t)}{\mu(A_t|S_t)}$ and $\rho_{t:t'} := \prod_{i=t}^{t'} \frac{\pi(A_i|S_i)}{\mu(A_i|S_i)}$ denote the per-decision step IS ratio (at time t) and the (product) sequence level ratios respectively. The PDIS Monte Carlo returns estimator at time step t, G_t^{PDIS} (Precup, Sutton, and Singh 2000), is given by,

$$G_t^{\text{PDIS}} := \sum_{k=t}^{\infty} \left(\prod_{i=t}^{k} \rho_i \right) \gamma^{k-t} R_{k+1} = \sum_{k=t}^{\infty} \rho_{t:k} \gamma^{k-t} R_{k+1} \tag{1}$$

For any policy μ that covers π (i.e., $\forall s, a\, \mu(a|s) = 0 \Rightarrow \pi(a|s) = 0$), the G^{PDIS} Monte Carlo returns estimator is unbiased (Precup, Sutton, and Singh 2000). Formally, for all $s \in \mathcal{S}$ we have, $\mathbb{E}_\mu[G_t^{\text{PDIS}} \mid S_t = s] = v_\pi(s)$. To reduce the variance of (1) it suffices to design a specific behaviour policy μ that produces lower-variance return estimates, while still covering π (Liu and Zhang 2024).

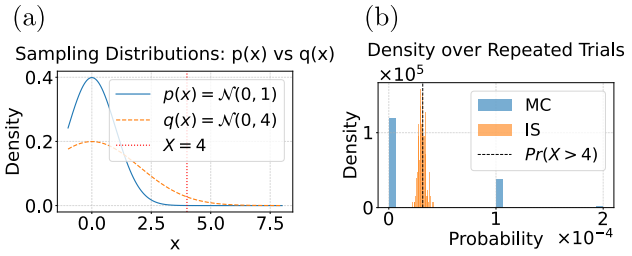

Figure 1: (a) Sampling densities. (b) Distribution of estimates over 100,000 trials.

3.3 Variance reduction

Variance reduction is an important tool in statistics, sampling can either be costly or time consuming; if the variance of our estimator is lower then we require fewer samples to get an accurate estimate (Owen 2013; Rubinstein and Kroese 2016). In RL the problem is two-fold, not only is collecting samples both costly and time consuming (Espeholt et al. 2018; Zhang 2024), but higher-variance estimates lead to higher-variance gradients and more unstable learning (Williams 1992; Mnih et al. 2016; Schulman et al. 2018). First, we provide an intuitive example of how importance sampling can be used to reduce variance.

Example. Consider the problem of estimating the probability that the random variable $X \sim p(\cdot)$ (with $p(\cdot) = \mathcal{N}(0, 1)$, i.e., standard normal) takes a value greater than 4, this can be written as $\mathbb{E}_{X \sim \mathcal{N}(0,1)}[\mathbf{1}(X > 4)]$, where $\mathbf{1}(\cdot)$ denotes the indicator function. After drawing N samples $X_1, X_2, \ldots X_N$, according to $p(\cdot)$, the Monte Carlo estimate for this quantity is given by $I^{(N)} := \sum_{i=1}^{N} \mathbf{1}[X_i > 4]$. This estimate is unbiased (i.e., $\mathbb{E}[I] = \mathbb{E}_{X \sim \mathcal{N}(0,1)}[\mathbf{1}[X > 4]]$), although it has high-variance; X taking a value greater than 4 is a low-probability event. Sampling with a longer tailed distribution, e.g., $q(\cdot) = \mathcal{N}(0, 4)$, is more efficient (lower-variance), correcting for the bias via IS weights $\rho(X) := p(X)/q(X)$, then $I_{IS}^{(N)} := \sum_{i=1}^{N} \rho(X_i) \mathbf{1}[X_i > 4]$ is the new estimate, for further intuition see Figure 1.

Variance Optimal Sampling. Optimal sampling distributions have been well-studied in statistics (Owen 2013; Rubinstein and Kroese 2016). Consider an arbitrary distribution $p : \mathcal{X} \to [0, 1]$ over the set $\mathcal{X}$ and a function $f : \mathcal{X} \to \mathbb{R}$. We want to estimate $\mathbb{E}_{X \sim p}[f(X)]$, the goal is to find the optimal sampling distribution $q : \mathcal{X} \to [0, 1]$ such that the variance of the IS estimator, $I_{\text{IS}} = \rho(X) f(x)$ is as small as possible, where $\rho(X) := p(X)/q(X)$ denotes the correct IS weights. Consider the following set of distributions Λ,

$$\Lambda := \{q \in \Delta(\mathcal{X}) \mid \forall x \in \mathcal{X}\ q(x) = 0 \Rightarrow p(x) f(x) = 0\}$$

where $\Delta(\mathcal{X})$ denotes the set of all distributions over $\mathcal{X}$. We note, this is slightly weaker than requiring q to cover p, although the estimator remains unbiased (Owen 2013). Let $\mathbb{V}$

denote the variance, the optimization problem then becomes,

$$\min_{q\in\Lambda} \mathbb{V}_{X\sim q}[\rho(X)f(X)] \tag{2}$$

Owen (2013) give an optimal solution to (2).

Lemma 1 (Owen 2013, Ch. 9.1)**.** *$\forall x \in \mathcal{X}$ let $q^*(x) \propto p(x)|f(x)|$. Then q^* is an optimal solution to (2).*

We note here that $q^*(x) \propto p(x)|f(x)|$ is defined as,

$$q^*(x) = p(x)|f(x)|/\textstyle\int_{x'\in\mathcal{X}} p(x')|f(x')|$$

When $\mathcal{X}$ is discrete, this integral reduces to a sum and q^* can be computed in closed form. However, we note, when $\mathcal{X}$ is continuous this integral may not have a closed form solution unless f has a convenient expression. The following result gives some more intuition on the optimality of q^*.

Lemma 2 (Owen 2013, Ch. 9.1)**.** *If $\forall x \in \mathcal{X}, f(x) \geq 0$ or $\forall x \in \mathcal{X}, f(x) \leq 0$, then $\Lambda = \Lambda_+$ (where Λ_+ denotes the set of all distributions giving unbiased estimators), and q^* (defined in Lemma 1) gives zero variance, i.e., $\mathbb{V}_{q^*}(I_{IS}) = 0$.*

Variance Optimal Sampling for off-policy evaluation. We now summarise Liu and Zhang's results. Liu and Zhang (2024) only consider undiscounted finite horizon policy evaluation, thus the notation here is slightly adapted so that it fits into the discounted setting. The goal is to minimize $\mathbb{V}_\mu(G_0^{\text{PDIS}})$ by designing a behaviour policy μ tailored to a specific target policy π, such that $\mathbb{E}_\mu[G_0^{\text{PDIS}}]$ is still unbiased. Liu and Zhang (2024) restrict themselves to searching for the *one-step optimal behaviour policy* rather than seeking *global optimality*, which is requires an estimate of the transition probabilities (Liu and Zhang 2024) (model-based). The search space of policies is denoted Λ (similar to before), which is contained in the space of all unbiased policies, denoted Λ_+, and contains the space of all policies covering π, denoted Λ_-. The one-step optimal policy, $\hat{\mu}$, is constructed by assuming we sample with $\hat{\mu}$ from the current step and π thereafter, rather than behaving optimally with μ^* thereafter, where μ^* denotes the globally variance optimal policy in Λ. The optimization problem can be written as,

$$\min_{\mu\in\Lambda} \mathbb{V}_{A_t\sim\mu, A_{t+1}\sim\pi,\ldots}\big(G_t^{\text{PDIS}} \mid S_t = s\big) \tag{3}$$

First we define,

$$\hat{q}_\pi(s,a) = q_\pi^2(s,a) + \nu_\pi(s,a) + \gamma^2\mathbb{E}_{s'\sim p}\Big[\mathbb{V}_\pi\big(G_{t+1}^{\text{PDIS}} \mid S_{t+1} = s'\big)\Big] \tag{4}$$

where $\nu(s,a) = \mathbb{V}_{S_{t+1}\sim p}(\gamma v_\pi(S_{t+1}) \mid S_t = s, A_t = a)$, is the next step variance of the value function v_π. Using Lemma 1, Liu and Zhang (2024) show that the one-step optimal policy $\hat{\mu}$ satisfying (3) is given by,

$$\hat{\mu}(a|s) \propto \pi(a|s)\sqrt{\hat{q}_\pi(s,a)} \tag{5}$$

While $\hat{q}_\pi$ might have a complicated definition, Liu and Zhang provide a convenient form which we can be learnt.

4 Off-policy TD(λ)

In this section we introduce a new new off-policy returns estimator (based on TD(λ)) which gives us better control of the bias-variance trade-off, and immediately fits into algorithms like PPO (Schulman et al. 2017). For this estimator we provide a proof of unbiasedness and variance reduction under behaviour policy $\hat{\mu}$ designed in (5), we also provide a practical way of estimating $\hat{q}_\pi$ in the discounted setting.

Truncated IS-TD(λ) returns. We introduce the truncated IS weighted TD(λ) returns, $G^{\text{TIS},\lambda}$, formally,

$$G_t^{\text{TIS},\lambda} = v_\pi(S_t) + \sum_{k=t}^{\infty}(\gamma\lambda)^{k-t}\left(\prod_{i=t}^{k-1} c_i\right)\delta_k \tag{6}$$

where $c_t := \min(\bar{c}, \frac{\pi(A_t|S_t)}{\mu(A_t|S_t)})$ and $\rho_t := \min(\bar{\rho}, \frac{\pi(A_t|S_t)}{\mu(A_t|S_t)})$ are the truncated IS ratios, and $\delta_k := \rho_k(R_{k+1} + \gamma v_\pi(S_{k+1}) - v_\pi(S_k))$ is the temporal difference (TD) error.

Remark 1. *Similar to eligibility traces (Sutton and Barto 2018), the truncated IS weighted TD(λ) returns can be computed efficiently, by stepping backwards through the following recursive definition,*

$$G_t^{TIS,\lambda} = v_\pi(S_t) + \delta_t + \gamma\lambda c_t\big(G_{t+1}^{TIS,\,\lambda} - v_\pi(S_{t+1})\big)$$

This estimator is reminiscent of V-trace (Espeholt et al. 2018) which also introduces truncated IS ratios, used to control the variance of the estimator. The truncation parameters $\bar{c}$ and $\bar{\rho}$ play different roles: since ρ_t appears in the definitions of the TD error, $\bar{\rho}$ affects the fixed point of convergence of the value function trained with targets $G^{\text{TIS},\lambda}$; for $\bar{\rho}$ this fixed point is exactly v_π, for $\bar{\rho} < \infty$ this fixed point is somewhere between v_π and v_μ. Here $\bar{c}$ does not affect the fixed point of convergence, the c_ts are "trace cutting" coefficients used as a variance reduction technique, so that TD errors further in the sequence may play less of a role in the update of the value function, reducing the chance of possible variance explosions (Espeholt et al. 2018). Furthermore, the parameter λ also acts as a variance reduction tool directly trading off bias and variance (with $\lambda = 1 \Rightarrow$ low-bias, $\lambda = 0 \Rightarrow$ low-variance). We summarize an important property of $G^{\text{TIS},\lambda}$.

Lemma 3. *Let $\bar{c}, \bar{\rho} \geq 1$, $\mu = \pi$, then, $\mathbb{E}_\mu[G_t^{TIS,\lambda}] = \mathbb{E}_\pi[G_t]$.*

This property is important, as with appropriate hyperparameter settings we can achieve unbiasedness. We treat $\bar{c}$ and $\bar{\rho}$ as hyperparameters; generally this result indicates that a good approach is to set $\bar{\rho} \geq \bar{c} \geq 1$. Additionally, λ should not stray too far from 1 as it biases our estimate when we rely on value estimates. We continue with the the following result.

Theorem 1 (Unbiasedness)**.** *Let $\bar{c}, \bar{\rho} = \infty$; then for all $\mu \in \Lambda$, $s \in \mathcal{S}$, $\lambda \in [0,1]$, we have, $\mathbb{E}_\mu[G_t^{TIS,\lambda} \mid S_t = s] = v_\pi(s)$*

Theorem 1 ensures that while we search over the space of policies Λ our estimator remains unbiased (under certain conditions). The full proof can be found in the extended version of the paper. Now we provide provable variance reduction for G_t^{TIS} under the behaviour policy $\hat{\mu}$ designed in (5).

Theorem 2 (Variance Reduction). *For any $s \in \mathcal{S}$, $\gamma \in [0, 1)$, and $\lambda = 1$, $\bar{c}, \bar{\rho} = \infty$,*

$$\mathbb{V}_{\hat{\mu}}\big(G_t^{TIS,\lambda} \mid S_t = s\big) \leq \mathbb{V}_{\pi}\big(G_t^{TIS,\lambda} \mid S_t = s\big) - \epsilon(s)$$

Where,

$$c(s) := \mathbb{E}_{a\sim\pi}[\hat{q}_\pi(s,a)] - \Big(\mathbb{E}_{a\sim\pi}\big[\sqrt{\hat{q}_\pi(s,a)}\big]\Big)^2$$

And, $\epsilon(s) := c(s) + \gamma^2 \mathbb{E}_{A_t\sim\hat{\mu}_t}\Big[\rho_t^2 \mathbb{E}_{S_{t+1}\sim p}\big[\epsilon(S_{t+1})\big]\Big]$

The full proof can be found in the extended version of the paper. Noting here that $c(s)$ is always non-negative by Jensen's inequality, which guarantees non-negativity of $\epsilon(s)$ and thus the property of variance reduction. $c(s) = 0$ occurs only in the degenerate case when all actions have the same $\hat{q}_\pi$ for state s. It remains to show how $\hat{q}_\pi$ can be learnt,

Theorem 3. *Let,*

$$\hat{r}_\pi(s,a) := 2r(s,a)q_\pi(s,a) - r^2(s,a) \tag{7}$$

Then for any $\gamma \in [0, 1)$,

$$\hat{q}_\pi(s,a) := \hat{r}_\pi(s,a) + \gamma^2 \mathbb{E}_{A_t\sim\pi, S_{t+1}\sim p}[\hat{q}_\pi(S_{t+1}, A_t)] \tag{8}$$

The proof here is a straightforward adaptation of (Liu and Zhang 2024) and intermediate results from the other theorems (see extended version). Theorem 3 demonstrates that $\hat{q}_\pi$ corresponds to the state-action value function of the policy π but for a different reward function $\hat{r}$ (c.f., (7)). Thus, designing a lower-variance behaviour policy boils down to learning an additional Q-value network to estimate $\hat{q}_\pi$ (also q_π – if not already available) and optimizing the behaviour policy μ so that it matches (5).

5 Implementation

We now provide a practical implementation of variance reduction via BPO. We implement BPO on top of two policy-gradient algorithms: REINFORCE (Williams 1992) and PPO (Schulman et al. 2017). In both cases the underlying algorithm remains unchanged, we simply introduce three additional features: (i) two additional Q-value networks are updated with several epochs of batch updates (via FQE (Le, Voloshin, and Yue 2019)) to provide estimates for q_π and $\hat{q}_\pi$; (ii) the behaviour policy is updated with several epochs of batch updates to match the target distribution in (5); (iii) the behaviour policy is then used to collect rollouts and the return estimates are computed with (6) – correctly accounting for the mismatch between the behaviour policy and target policy, and truncating IS weights to reduce variance. Algorithm 1 outlines a general instantiation of BPO implemented on top of a generic policy-gradient algorithm. We continue this section by describing the important implementation details.

5.1 Integration with Policy Gradient

Policy gradient methods directly optimize the policy given as a parametrized function π_θ, where θ denotes the policy parameters. The policy is updated with gradient steps using the policy gradient (Sutton and Barto 2018):

$$\nabla J(\theta) \propto \mathbb{E}_{\pi_\theta}\big[G_t \nabla \ln \pi_\theta(A_t|S_t)\big] \tag{9}$$

Algorithm 1: BPO

1: **Initialize** θ, ω, ζ, $\hat{\zeta}$, ξ and empty replay buffer D
2: **for** phase $= 0, 1, \ldots$ **do**
3: Generate $S_0 A_0 R_1 \ldots S_{T-1} A_{T-1} R_{T-1}$ with μ
4: Add experience tuples (s_i, a_i, r_i, s_i') to D
5: *// Policy gradient update*
.....
.....
.....
6: *// Auxilliary steps*
7: **for** epoch $= 0, 1, \ldots$ n_qvf_epochs **do**
8: Optimize Q_ζ with FQE (c.f., (13))
9: Compute $\hat{r}_t$ (c.f., (7))
10: **for** epoch $= 0, 1, \ldots$ n_qvf_epochs **do**
11: Optimize $\hat{Q}_{\hat{\zeta}}$ with FQE (c.f., (14))
12: **for** epoch $= 0, 1, \ldots$ n_mu_epochs **do**
13: Optimize behaviour policy μ_ξ (c.f., (15) or (16))

In PPO (Schulman et al. 2017) an additional parametrized value function V_ω is learnt, where ω denotes the parameters of the value function, and the policy parameters are updated using the clipped surrogate objective function,

$$L^{\text{clip}}(\theta) = \hat{\mathbb{E}}_t\big[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\big] \tag{10}$$

where $\hat{\mathbb{E}}$ corresponds to the empirical expectation under the rollouts. The ratios $r_t(\theta)$ are calculated differently here as μ is used to collect rollouts (c.f., line 3 in Algorithm 1), that is, $r_t(\theta) = \frac{\pi_\theta(A_t|S_t)}{\mu(A_t|S_t)}$ corresponds to the ratio between the current policy π_θ and the behaviour policy μ. Furthermore, the advantage estimates are modified to use the truncated IS TD(λ) returns,

$$\hat{A}_t := \hat{G}_t^{\text{TIS},\lambda} - V_\omega(S_t) \tag{11}$$

where $\hat{G}_t^{\text{TIS},\lambda}$ denotes the truncated IS TD(λ) returns (c.f., 6), but with value estimates given by V_ω.

The value function V_ω is updated in tandem to minimize the squared error between its predictions and the truncated IS TD(λ) estimates,

$$L^{\text{value}}(\omega) = \frac{1}{2}\hat{\mathbb{E}}_t\Big[\big(V_\omega(S_t) - \hat{G}_t^{\text{TIS},\lambda}\big)^2\Big] \tag{12}$$

These objectives are typically optimized jointly with an additional entropy bonus, $L^{\text{joint}}(\theta,\omega) = L^{\text{clip}}(\theta) + \beta_{\text{value}} \cdot L^{\text{value}}(\omega) + \beta_{\text{ent}} \cdot L^{\text{ent}}(\theta)$, where β_{value} and β_{ent} are coefficients providing the relative weightings for each objective.

5.2 Fitted Q-evaluation

Recall that to design variance optimal behaviour policies we need to estimate the state-action value function $\hat{q}_\pi$, which includes the term q_π in its definition (c.f., 8). Thus we train two additional parametrized functions, Q_ζ and $\hat{Q}_{\hat{\zeta}}$.

For training both Q_ζ and $\hat{Q}_{\hat{\zeta}}$ we use FQE (Le, Voloshin, and Yue 2019) – a method for learning the state-action value functions of arbitrary policies from an offline dataset. In

our instance, the offline dataset corresponds to a relatively small replay buffer D containing tuples $\{(s_i, a_i, r_i, s'_i)\}_{i=1}^m$ of the most recent rollout data collected by μ. The two Q-networks, Q_ζ and $\hat{Q}_{\hat{\zeta}}$ are updated for several epochs using one-step Bellman targets computed from batches of data sampled from the replay buffer D.

First Q_ζ is updated to estimate q_π, using the following loss function,

$$L^{FQE}(\zeta) = \frac{1}{2}\hat{\mathbb{E}}_t\Big[\big(Q_\zeta(S_t, A_t) - Q_t^{\text{targ}}\big)^2\Big] \quad (13)$$

where $Q_t^{\text{targ}} = r_t + \gamma \mathbb{E}_{A_{t+1}\sim\pi_\theta}\big[Q_\zeta(s'_t, A_{t+1})\big]$ corresponds to the one-step Bellman targets. Once Q_ζ is trained then the rewards $\hat{r}_t$ are calculated with (7) and the tuples are updated, i.e., $D = \{(s_i, a_i, \hat{r}_i, s'_i)\}_{i=1}^m$. The loss function for $\hat{Q}_{\hat{\zeta}}$ is,

$$\hat{L}^{FQE}(\hat{\zeta}) = \frac{1}{2}\hat{\mathbb{E}}_t\Big[\big(\hat{Q}_{\hat{\zeta}}(S_t, A_t) - \hat{Q}_t^{\text{targ}}\big)^2\Big] \quad (14)$$

except r_t, Q_ζ and γ, are replaced by $\hat{r}_t$, $\hat{Q}_{\hat{\zeta}}$ and γ^2 for computing the targets, $\hat{Q}_t^{\text{targ}} = \hat{r}_t + \gamma^2 \mathbb{E}_{A_{t+1}\sim\pi_\theta}\big[\hat{Q}_\zeta(s'_t, A_{t+1})\big]$.

Stabilizing learning. The stability of the estimates for q_π and $\hat{q}_\pi$ are critical to BPO. Value-based approaches, including FQE, are known to be prone to overestimation and approximation bias (Fujimoto et al. 2022). To stabilize learning we use *symlog targets* (Hafner et al. 2024), where the targets Q_t^{targ} and $\hat{Q}_t^{\text{targ}}$ are transformed with the symmetric log: $\text{symlog}(x) = \text{sign}(x)\ln(|x|+1)$. To preserve their magnitudes the predictions are transformed back with the symmetric exp: $\text{symexp}(x) = \text{sign}(x)(\exp(|x|) - 1)$. Symlog targets squash large magnitudes allowing the two networks to quickly move their predictions to large values when needed. This is crucial as the relative magnitude between Q_ζ and $\hat{Q}_{\hat{\zeta}}$ predictions can be large and symlog targets allow us to use the same hyperparameters (e.g., learning rate) for both Q_ζ and $\hat{Q}_{\hat{\zeta}}$ and keep the learning dynamics consistent for both networks. Further techniques used to deal with overestimation and stability issues can be found in the appendix of the extended version of the paper.

5.3 Behaviour Policy

The behaviour policy is also a parametrized function μ_ξ with parameters ξ. Generally we keep the architecture between π_θ and μ_ξ consistent, and initialize θ and ξ with the same seed, so they are identical at the start of training, although they will likely diverge after just one gradient step.

Discrete action spaces. For the discrete case, the target distribution (c.f., (5)) can be computed analytically, thus we could sample directly proportional to (5), however to alleviate any possible issues arising from overestimation we still train a separate parametrized behaviour policy μ_ξ using the cross entropy loss to the target distribution:

$$L_\mu^{\text{disc}}(\xi) = -\hat{\mathbb{E}}_t\Big[\sum_{a\in\mathcal{A}} \mu_\xi(a|S_t)\ln q(a|S_t)\Big] \quad (15)$$

$q(a|s) = \pi_\theta(a|s)\sqrt{\hat{Q}_{\hat{\zeta}}(s,a)} / \sum_{a'\in A}\pi_\theta(a'|s)\sqrt{\hat{Q}_{\hat{\zeta}}(s,a')}$ is the target distribution computed analytically.

Continuous action spaces. For the continuous case, the target distribution (c.f., (5)) cannot be computed analytically without making distributional assumptions on $\hat{q}_\pi$. We still seek to minimize the log probability distance between $\mu(a|s)$ and $\pi(a|s)\sqrt{\hat{q}_\pi(s,a)}$ (c.f., (5)). We design the following loss function,

$$L_\mu^{\text{cont}}(\xi) = \hat{\mathbb{E}}_t\Big[\ln\mu_\xi(A_t|S_t) - \ln\pi_\theta(A_t|S_t) - \frac{1}{2}\ln\hat{Q}_{\hat{\zeta}}(S_t, A_t)\Big] \quad (16)$$

We establish the following result for this loss function, the proof can be found in the extended version.

Theorem 4. *If $\mu_\xi(a|s) \propto \pi_\theta(a|s)\sqrt{\hat{Q}_{\hat{\zeta}}(s,a)}$ (i.e., matches (5)), then the loss $L_\mu^{cont}(\xi)$ is minimized.*

6 Experimental Results

In this section we present our experimental results. We start with REINFORCE (Williams 1992), which is a simple algorithm that uses the full Monte Carlo returns (i.e., $\lambda = 1$) to update the policy with the policy gradient (c.f., (9)). Given the relatively simplicity of REINFORCE we test our approach on a simple, yet non-trivial toy example from (Sutton and Barto 2018): the `ShortCorridor` grid world. Given the example is simple, the gains here are marginal, and these results are simply to demonstrate that we can use our method for a variety of policy-gradient algorithms. We then continue by presenting our results for BPO on much more complex environments, where the underlying algorithm we build on top of PPO (Schulman et al. 2017) – a more sophisticated and widely used algorithm.

REINFORCE. The action space for `ShortCorridor` grid world (c.f., Figure 2) is $\mathcal{A} = \{\text{left}, \text{right}\}$. All states, $\mathcal{S} = \{0, 1, 2, 3\}$ are identical under function approximation, $\mathbf{x}(s, \text{left}) = [0, 1]^T$ and $\mathbf{x}(s, \text{right}) = [1, 0]^T$. The problem is challenging as in state 1 the left and right actions are reversed. Given that the agent can't distinguish states, the optimal policy is a stochastic policy picking right with probability 0.59, achieving an optimal expected return of -11.6. We

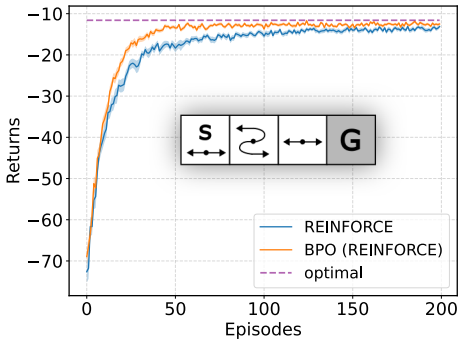

Figure 2: `ShortCorridor` environment (center) and mean returns (10 eval episodes); averaged over 100 independent runs with standard error (SE) bars reported.

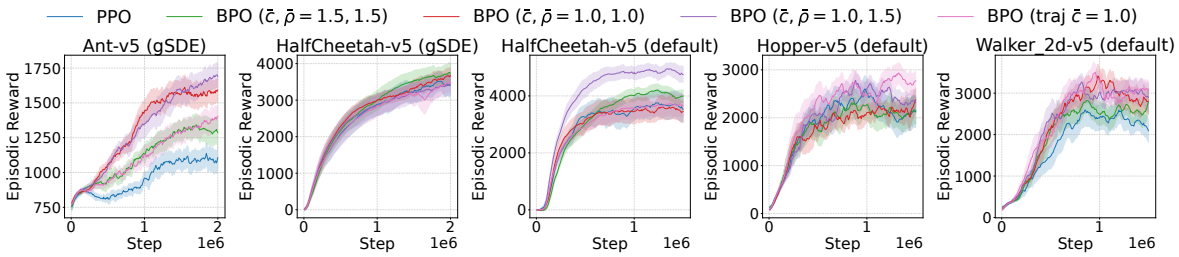

Figure 3: MuJoCo mean returns (10 eval episodes); averaged over 10 independent runs with SE bars reported.

	Ant-v5 (gSDE)	HalfCheetah-v5 (gSDE)	HalfCheetah-v5 (default)	Hopper-v5 (default)	Walker2d-v5 (default)
PPO	1106 ± 111	3425 ± 468	3527 ± 670	2126 ± 492	2091 ± 408
BPO ($\bar{c}, \bar{\rho} = 1.5, 1.5$)	$\mathbf{1287 \pm 148}$	*3742 ± 408*	$\mathbf{3999 \pm 320}$	*2143 ± 422*	$\mathbf{2782 \pm 456}$
BPO ($\bar{c}, \bar{\rho} = 1.0, 1.0$)	$\mathbf{1592 \pm 126}$	*3674 ± 321*	3449 ± 550	*2373 ± 377*	$\mathbf{2805 \pm 470}$
BPO ($\bar{c}, \bar{\rho} = 1.0, 1.5$)	$\mathbf{1690 \pm 125}$	*3431 ± 431*	$\mathbf{4749 \pm 419}$	*2316 ± 326*	$\mathbf{2898 \pm 513}$
BPO (traj $\bar{c} = 1.0$)	$\mathbf{1400 \pm 122}$	*3477 ± 356*	*3656 ± 588*	$\mathbf{2770 \pm 296}$	$\mathbf{3044 \pm 332}$

Table 1: MuJoCo mean returns (10 eval episodes after training) with two hyperparameter settings: (default) and (gSDE); **bold** denotes a statistically significant improvement over the baseline; *italic* denotes non-statistically significant improvements.

compare BPO with REINFORCE to standard REINFORCE (Williams 1992). Both the learning rate and underlying update rule for π_θ are fixed for a fair comparison. We see consistent policy improvement for both algorithms in the early stages of training, in the later stages of training BPO is more stable and converges faster, indicating the behaviour policy μ has converged to the optimal sampling distribution. The best truncation parameters we found here were $\bar{c} = 1.0$ and $\bar{\rho} = 1.5$. Even though the advantage here is small, BPO appears more stable towards the end of training, thus we could possibly push the learning rate further to its limit.

PPO. We now evaluate BPO on several MuJoCo (Todorov, Erez, and Tassa 2012) environments provided by Gymnasium (Towers et al. 2025), testing also, different hyperparameter settings for the underlying PPO algorithm. In particular we test with the standard hyperparameter settings (default) provided in (Schulman et al. 2017), we also test with generalized state-dependent exploration (gSDE) (Raffin, Kober, and Stulp 2022). All hyperparameter settings are detailed in the appendix of the extended version, we note that for a fair comparison between BPO and PPO, we keep the underlying PPO parameters identical and we only tune the BPO specific hyperparameters (e.g., $\bar{c}$, $\bar{\rho}$), although in practice one might be able to push the PPO hyperparameters further (e.g., $\lambda \to 1$) as BPO reduces variance. The results are summarized in Table 1. We also provide complete learning curves for each environment, see Figure 3.

Discussion. In our experiments we search over various hyperparameters settings for the truncation parameters $\bar{c}$ and $\bar{\rho}$, we also experiment with trajectory level truncation (traj), where the full product $(\prod_{i=t}^{k-1} c_i)\rho_k$ in (6) is truncated with $\bar{c}$, rather than truncating each individual IS ratio. The advantage of this, is ratios greater than $\bar{c}$ can propagate through the trajectory, but the product is always truncated for stable learning. Again, this can be efficiently computing by stepping backward through time. In Table 1 we see that almost all configurations of BPO improve upon the baseline (PPO), and in most cases these improvements are statistically significant. Studying Figure 3 we see that BPO often exhibits faster convergence at the start of training and is more stable towards the end, demonstrating the well-designed behaviour policies can not only theoretically but empirically improve the convergence and stability of online RL, by reducing the variance of the returns estimator. In the extended version of the paper we conduct some ablations for BPO, to observe the effect of removing the $\hat{q}_\pi$ estimate from the loss function in (16) and investigating the effect of *symlog targets*.

7 Conclusion

In this paper we demonstrate that well designed behaviour policies can provably reduce the variance of practical returns estimators for online RL. We develop a sound methodology for training variance-reducing behaviour policies which are used to collect off-policy data for concurrently optimizing a target policy. Empirically, we show that our approach enables stable and more efficient off-policy RL, compared to the on-policy counterparts. Directions for future work include, extending our methodology to other policy-gradient algorithms, e.g., A2C (Mnih et al. 2016). It would also be worth exploring if our methodology is useful for value-based methods, e.g., DDPG (Silver et al. 2014), TD3 (Fujimoto, Hoof, and Meger 2018) and SAC (Haarnoja et al. 2018). We might also use model-based RL (Sutton 1990) to assist in designing globally optimal behaviour policies.

Acknowledgments

The research described in this paper was partially supported by the EPSRC (grant number EP/X015823/1).

References

Agarwal, A.; Basu, S.; Schnabel, T.; and Joachims, T. 2017. Effective evaluation using logged bandit feedback from multiple loggers. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 687–696.

Ernst, D.; Geurts, P.; and Wehenkel, L. 2005. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6.

Espeholt, L.; Soyer, H.; Munos, R.; Simonyan, K.; Mnih, V.; Ward, T.; Doron, Y.; Firoiu, V.; Harley, T.; Dunning, I.; et al. 2018. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*, 1407–1416. PMLR.

Fu, J.; Kumar, A.; Nachum, O.; Tucker, G.; and Levine, S. 2021. D4RL: Datasets for Deep Data-Driven Reinforcement Learning. arXiv:2004.07219.

Fujimoto, S.; Hoof, H.; and Meger, D. 2018. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, 1587–1596. PMLR.

Fujimoto, S.; Meger, D.; and Precup, D. 2019. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, 2052–2062. PMLR.

Fujimoto, S.; Meger, D.; Precup, D.; Nachum, O.; and Gu, S. S. 2022. Why Should I Trust You, Bellman? The Bellman Error is a Poor Replacement for Value Error. In *International Conference on Machine Learning*, 6918–6943. PMLR.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. arXiv:2501.12948.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, 1861–1870. PMLR.

Hafner, D.; Pasukonis, J.; Ba, J.; and Lillicrap, T. 2024. Mastering Diverse Domains through World Models. arXiv:2301.04104.

Hanna, J. P.; Thomas, P. S.; Stone, P.; and Niekum, S. 2017. Data-efficient policy evaluation through behavior policy search. In *International Conference on Machine Learning*, 1394–1403. PMLR.

Jaderberg, M.; Mnih, V.; Czarnecki, W. M.; Schaul, T.; Leibo, J. Z.; Silver, D.; and Kavukcuoglu, K. 2016. Reinforcement Learning with Unsupervised Auxiliary Tasks. arXiv:1611.05397.

Le, H.; Voloshin, C.; and Yue, Y. 2019. Batch policy learning under constraints. In *International Conference on Machine Learning*, 3703–3712. PMLR.

Levine, S.; Kumar, A.; Tucker, G.; and Fu, J. 2020. Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems. arXiv:2005.01643.

Liu, S.; and Zhang, S. 2024. Efficient policy evaluation with offline data informed behavior policy design. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Mathieu, M.; Ozair, S.; Srinivasan, S.; Gulcehre, C.; Zhang, S.; Jiang, R.; Paine, T. L.; Powell, R.; Żołna, K.; Schrittwieser, J.; et al. 2023. AlphaStar Unplugged: Large-Scale Offline Reinforcement Learning. arXiv:2308.03526.

Mnih, V.; Badia, A. P.; Mirza, M.; Graves, A.; Lillicrap, T.; Harley, T.; Silver, D.; and Kavukcuoglu, K. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, 1928–1937. PmLR.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533.

Mukherjee, S.; Hanna, J. P.; and Nowak, R. D. 2022. Revar: Strengthening policy evaluation via reduced variance sampling. In *Uncertainty in Artificial Intelligence*, 1413–1422. PMLR.

Munos, R.; Stepleton, T.; Harutyunyan, A.; and Bellemare, M. 2016. Safe and efficient off-policy reinforcement learning. *Advances in neural information processing systems*, 29.

Owen, A. B. 2013. *Monte Carlo theory, methods and examples*. Stanford.

Precup, D.; Sutton, R. S.; and Singh, S. P. 2000. Eligibility Traces for Off-Policy Policy Evaluation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, 759–766. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. ISBN 1558607072.

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

Raffin, A.; Kober, J.; and Stulp, F. 2022. Smooth exploration for robotic reinforcement learning. In *Conference on robot learning*, 1634–1644. PMLR.

Rubinstein, R. Y.; and Kroese, D. P. 2016. *Simulation and the Monte Carlo Method*. Wiley Publishing, 3rd edition. ISBN 1118632168.

Schrittwieser, J.; Hubert, T.; Mandhane, A.; Barekatain, M.; Antonoglou, I.; and Silver, D. 2021. Online and offline reinforcement learning by planning with a learned model. *Advances in Neural Information Processing Systems*, 34: 27580–27591.

Schulman, J.; Moritz, P.; Levine, S.; Jordan, M.; and Abbeel, P. 2018. High-Dimensional Continuous Control Using Generalized Advantage Estimation. arXiv:1506.02438.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347.

Silver, D.; Lever, G.; Heess, N.; Degris, T.; Wierstra, D.; and Riedmiller, M. 2014. Deterministic policy gradient algorithms. In *International conference on machine learning*, 387–395. PMLR.

Silver, D.; Schrittwieser, J.; Simonyan, K.; Antonoglou, I.; Huang, A.; Guez, A.; Hubert, T.; Baker, L.; Lai, M.; Bolton, A.; et al. 2017. Mastering the game of go without human knowledge. *nature*, 550(7676): 354–359.

Sutton, R. S. 1990. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine learning proceedings 1990*, 216–224. Elsevier.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement Learning: An Introduction*. The MIT Press, second edition.

Todorov, E.; Erez, T.; and Tassa, Y. 2012. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, 5026–5033. IEEE.

Towers, M.; Kwiatkowski, A.; Terry, J.; Balis, J. U.; Cola, G. D.; Deleu, T.; Goulão, M.; Kallinteris, A.; Krimmel, M.; KG, A.; Perez-Vicente, R.; Pierré, A.; Schulhoff, S.; Tai, J. J.; Tan, H.; and Younis, O. G. 2025. Gymnasium: A Standard Interface for Reinforcement Learning Environments. arXiv:2407.17032.

Vinyals, O.; Babuschkin, I.; Czarnecki, W. M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D. H.; Powell, R.; Ewalds, T.; Georgiev, P.; et al. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *nature*, 575(7782): 350–354.

Wang, Z.; Bapst, V.; Heess, N.; Mnih, V.; Munos, R.; Kavukcuoglu, K.; and de Freitas, N. 2017. Sample Efficient Actor-Critic with Experience Replay. arXiv:1611.01224.

Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3): 229–256.

Zhang, S. 2024. A New Challenge in Policy Evaluation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(13): 15465–15465.

Zhang, S.; Liu, B.; and Whiteson, S. 2020. Gradientdice: Rethinking generalized offline estimation of stationary values. In *International Conference on Machine Learning*, 11194–11203. PMLR.

Zhong, R.; Zhang, D.; Schäfer, L.; Albrecht, S.; and Hanna, J. 2022. Robust on-policy sampling for data-efficient policy evaluation in reinforcement learning. *Advances in Neural Information Processing Systems*, 35: 37376–37388.