Machine Learning (2021) 110:1267–1317
https://doi.org/10.1007/s10994-020-05938-9

Importance sampling in reinforcement learning with an estimated behavior policy

Josiah P. Hanna[1] · **Scott Niekum**[2] · **Peter Stone**[2]

Received: 9 June 2020 / Revised: 9 June 2020 / Accepted: 21 December 2020 /
Published online: 7 May 2021

Abstract
In reinforcement learning, importance sampling is a widely used method for evaluating an expectation under the distribution of data of one policy when the data has in fact been generated by a different policy. Importance sampling requires computing the likelihood ratio between the action probabilities of a target policy and those of the data-producing behavior policy. In this article, we study importance sampling where the behavior policy action probabilities are replaced by their maximum likelihood estimate of these probabilities under the observed data. We show this general technique reduces variance due to sampling error in Monte Carlo style estimators. We introduce two novel estimators that use this technique to estimate expected values that arise in the RL literature. We find that these general estimators reduce the variance of Monte Carlo sampling methods, leading to faster learning for policy gradient algorithms and more accurate off-policy policy evaluation. We also provide theoretical analysis showing that our new estimators are consistent and have asymptotically lower variance than Monte Carlo estimators.

Keywords Reinforcement learning · Policy evaluation · Importance sampling

This article contains material that was previously presented at the International Conference on Machine Learning (ICML) 2019 and the International Conference on Autonomous Agents and Multi-agent Systems (AAMAS) 2019.

Editor: Alan Fern.

✉ Josiah P. Hanna
josiah.hanna@ed.ac.uk

Scott Niekum
sniekum@cs.utexas.edu

Peter Stone
pstone@cs.utexas.edu

[1] School of Informatics, University of Edinburgh, Edinburgh EH8 9AB, UK

[2] Department of Computer Science, University of Texas at Austin, Austin, TX 78712-1757, USA

1 Introduction

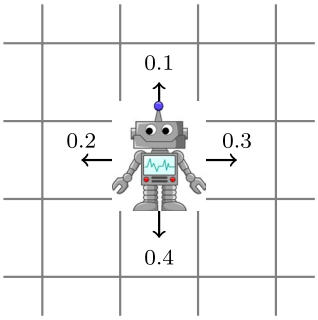
The field of *reinforcement learning* (RL) seeks to model an autonomous agent interacting with a task while learning through trial-and-error interaction. RL algorithms result in *policies* that tell the agent how to act in all possible world states in order to complete a particular task. Despite much recent empirical success (Mnih et al. 2015; Silver et al. 2016), many RL algorithms remain prohibitively sample inefficient—the amount of task interactions they require before a high-performing policy is found may be beyond what is possible on many real world problems found in fields such as medicine or robotics. If these RL algorithms are to be broadly applied, it is imperative to address this data inefficiency.

A fundamental problem in the reinforcement learning literature is estimating the expected value of a function under the distribution of data induced by a policy. For example, in policy gradient RL, algorithms must estimate the expected value of the policy gradient under the distribution of states and actions that the current policy induces (Sutton and Barto 1998). In batch policy evaluation (Li et al. 2015; Thomas and Brunskill 2016a), algorithms must estimate the expected return of a policy π under the distribution of state-action trajectories that π induces. We call this problem the *expectation evaluation* problem. Data efficient solutions to this problem are an important step towards data efficient RL. In this work, we introduce methods that increase the data efficiency of expectation evaluation methods in reinforcement learning.

One widely used approach for the expectation evaluation problem is to use a sample-average or *Monte Carlo* estimate of the desired expectation. This approach is straightforward: the policy is run to sample data and then the function values under the resulting data are averaged. In the limit, as the amount of sampled data increases, the estimate probabilistically converges to the true expected value. However, for a finite amount of data, it may exhibit high variance that causes error in the estimate. Variance in a Monte Carlo estimate arises when the observed samples occur at different frequencies than they would in expectation. For example, if a policy selects between two actions with equal probability in a given state, the resulting data may show that one action occurred 60% of the time while the other action occurred only 40% of the time. With this observed data, the Monte Carlo estimate will place too much emphasis on the first action and not enough emphasis on the second. We term this source of variance *sampling error* and provide an illustration in Fig. 1; reducing sampling error is the main benefit of the methods we introduce.

In this work, we frame the sampling error problem as an off-policy policy evaluation problem. In the off-policy policy evaluation problem, we are interested in observing data under one policy, π, but instead observe data from a different, *behavior* policy. We observe that though we are interested in observing data under a policy π, sampling error may result in our data appearing to have been generated by a different, *empirical* policy, $\hat{\pi}$. This observation motivates correcting sampling error with the well-known off-policy technique of importance sampling (Precup et al. 2000). In this article, we propose first estimating the empirical policy from observed state-action pairs and then using this policy as the behavior policy in an importance sampling estimate. Figure 1 illustrates how this approach corrects sampling error in Monte Carlo sampling. The combination of importance sampling with an estimated behavior policy to correct sampling error is the central contribution of this work.

It may be natural to assume that importance sampling with an estimated behavior policy will perform worse than with the true behavior policy probabilities because it is using an estimate in place of the "correct" behavior policy probability. Furthermore, it may appear that importance sampling is unnecessary in the on-policy case. However, in this work, we



Action	π	Observed proportion, $\hat{\pi}$	Monte Carlo weight	SEC weight **(ours)**
Up	0.1	0.15	0.15	0.1
Right	0.3	0.35	0.35	0.3
Down	0.4	0.3	0.3	0.4
Left	0.2	0.2	0.2	0.2

Fig. 1 Sampling error in a fixed state s of a Grid World environment. Each action a is sampled with probability $\pi(a|s)$ and is observed in the proportion given by $\hat{\pi}(a|s)$. Monte Carlo weighting gives each action the weight $\hat{\pi}(a|s)$ while our novel sampling error corrected (SEC) weighting gives each action the weight $\hat{\pi}(a|s)\frac{\pi(a|s)}{\hat{\pi}(a|s)} = \pi(a|s)$. In other words, the SEC estimator weights each action by the expected frequency for each a in s while the Monte Carlo estimator will have error unless the empirical frequency of sampled actions, $\hat{\pi}$, is equal to the expected frequency, π for all actions

show that importance sampling with an estimated behavior policy lowers the variance of expectation evaluation in both on- and off-policy settings. Our work complements existing approaches in the causal inference (Rosenbaum 1987; Hirano et al. 2003) and bandit (Li et al. 2015; Narita et al. 2019) literatures that has used importance sampling with an estimated behavior policy as a variance reduction strategy. We extend this general approach to sequential decision making tasks.

We first consider expectation evaluation for expectations of the form:

$$\mathbf{E}\left[\phi(S,A) \;\middle|\; S \sim d_\pi, A \sim \pi\right],$$

where ϕ is a vector or scalar-valued function of state-action pairs and d_π is the distribution of states that policy π will encounter. This form of expected value arises in policy gradient reinforcement learning (Peters and Schaal 2008; Schulman et al. 2015) as well as average reward reinforcement learning (Puterman 2014; Schwartz 1993; Mahadevan 1996). We introduce a novel expectation evaluation estimator called the *sampling error corrected* (SEC) estimator that reduces sampling error in Monte Carlo estimates by importance sampling with an estimated behavior policy. We prove (under a limiting set of assumptions) that the SEC estimator has variance at most that of the Monte Carlo estimator and (under lighter assumptions) that this approach has asymptotic variance at most that of the Monte Carlo estimator. We then instantiate the SEC estimator for the problem of estimating the policy gradient when running a batch policy gradient algorithm. We introduce the *sampling error corrected policy gradient* estimator and present an empirical study in which our new estimator leads to faster convergence of batch policy gradient algorithms for the REINFORCE algorithm (Williams 1992) and trust-region policy optimization (Schulman et al. 2015) compared to the these algorithms using the Monte Carlo estimator.

We next consider expectation evaluation when the target expectation takes the form:

$$\mathbf{E}\left[\chi(H) \;\middle|\; H \sim \pi\right],$$

where χ is a vector or scalar-valued function of trajectories, H, generated by following π. This form of expected value arises in the problem of policy evaluation where we wish to estimate the expected return when running a particular policy π (Jiang and Li 2016; Thomas and Brunskill 2016a). When expectations take this form, it is not always straightforward to recast the expectation as an expectation under state-action pairs, e.g., in finite-horizon off-policy evaluation. Thus our new SEC estimator is inapplicable. We show that sampling error can be viewed as an off-policy expectation evaluation problem where the behavior policy is a non-Markovian policy that conditions its action selection on the entire history of past states and actions. We introduce a family of regression importance sampling (RIS) estimators that estimate a possibly non-Markovian policy as the behavior policy for importance sampling. Under similar assumptions to those made for the SEC estimator, we prove that all RIS estimators are consistent and have asymptotic variance at most that of the Monte Carlo estimator. Finally, we instantiate RIS methods for the problem of off-policy batch policy evaluation and present an empirical study showing that regression importance sampling leads to lower mean squared error off-policy policy evaluation than standard importance sampling baselines.

This article proceeds as follows. In Sect. 2, we introduce necessary background: reinforcement learning notation, two common forms of expectation evaluation in RL, the on- and off-policy Monte Carlo estimator, and the concept of sampling error in the Monte Carlo estimator. In Sect. 3 we introduce the SEC estimator that uses importance sampling with an estimated behavior policy to correct sampling error in state-action expectations and establish theoretical properties of this novel estimator. Then, in Sect. 4, we apply the SEC estimator to estimating the policy gradient in a batch policy gradient algorithm and empirically show faster convergence rates on several RL tasks. In Sect. 5 we turn to trajectory expectations and introduce a family of regression importance sampling estimators that use importance sampling with an estimated behavior policy to reduce sampling error. We provide theoretical analysis of this family of estimators, establishing consistency and asymptotic variance analysis. Then, in Sect. 6, we apply RIS estimators to the problem of off-policy policy evaluation and show our new estimators yield lower mean squared error estimates than off-policy Monte Carlo methods. In Sect. 7, we discuss prior literature on importance sampling with an estimated behavior policy, addressing sampling error, and reducing variance in reinforcement learning. Finally, we discuss the strengths and limitations of our new methods and results, discuss avenues for future research, and conclude.

2 Background

In this section we first introduce the notation used throughout this work. We then discuss the expectation evaluation problem in the reinforcement learning literature. Finally, we discuss Monte Carlo sampling as a solution method for expectation evaluation problems.

2.1 Notation

We assume the environment is an episodic *Markov decision process* with state set $\mathcal{S}$, action set $\mathcal{A}$, transition function, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$, reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, discount factor γ, and initial state distribution d_0 (Puterman 2014). For simplicity, we assume that $\mathcal{S}$ and $\mathcal{A}$ are finite, though all methods and theoretical results discussed in this paper are applicable to both finite and infinite $\mathcal{S}$ and $\mathcal{A}$, unless otherwise noted. We assume that the transition and reward functions are unknown. A policy, $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$, is a function mapping states and actions to probabilities. We use $\pi(a|s) := \pi(s, a)$ to denote the conditional probability of action a given state s and $P(s'|s, a) := P(s, a, s')$ to denote the conditional probability of state s' given state s and action a.

The agent interacts with the environment MDP as follows: The agent begins in initial state $S_0 \sim d_0$. At discrete time-step t the agents takes action $A_t \sim \pi(\cdot|S_t)$. The environment responds with $R_t := r(S_t, A_t)$ and $S_{t+1} \sim P(\cdot|S_t, A_t)$ according to the reward function and transition function. After interacting with the environment for at most l steps the agent returns to a new initial state and the process repeats. For notational convenience, we assume that all interactions last for at most l steps. In the MDP definition, we also include a terminal state, s_∞, that allows the possibility of episodes ending before time-step l. If at any time-step, t, $S_t = s_\infty$, then for all $t' \geq t$, $S_{t'} = s_\infty$ and $R_{t'} = 0$.

Let $h := (s_0, a_0, r_0, s_1, \ldots, s_{l-1}, a_{l-1}, r_{l-1})$ be a *trajectory* and $g(h) := \sum_{t=0}^{l-1} \gamma^t r_t$ be the *discounted return* of h. For trajectory h, we will use $h_{t:t'}$ to denote the partial trajectory, $s_t, a_t, r_t, ..., s_{t'}, a_{t'}, r_{t'}$. If $t < 0$, $h_{t:t'}$ denotes the beginning of the trajectory until step t'. Any policy induces a distribution over trajectories, $\Pr(H = h|\pi)$, where H is a random variable representing a trajectory. The distribution over trajectories induces a distribution over sets of m trajectories, $\Pr(D = \{h_1, ...h_m\}|\pi)$, where D is a random variable representing a set of trajectories. We will write $H \sim \pi$ to denote sampling a trajectory by following π and $D \sim \pi$ to denote sampling a set of trajectories by following π. We use B for the random variable representing all k state-action pairs observed in D.[1] A policy also induces a distribution over state visitation frequencies, $d_\pi : \mathcal{S} \to [0, 1]$.

We define the *value* of a policy, $v(\pi) := \mathbf{E}[g(H)|H \sim \pi]$, as the expected discounted return when sampling a trajectory with policy π.

2.2 Expectation evaluation in reinforcement learning

An important problem that arises across the reinforcement learning literature is the problem of evaluating expectations of functions under the distribution of data induced by a policy. In this section we introduce this problem as the *expectation evaluation* problem. We describe two general forms of expected value that occur in the reinforcement learning literature and give examples of their occurrence. In the following subsection we will discuss how both forms of expected values can be approximated with Monte Carlo sampling.

[1] Because we allow early termination, k equals at most ml but may be smaller. We do *not* include (S, A) pairs in B if $S = s_\infty$.

2.2.1 State-action expectations

The first form of expected value we consider is the expectation of a function of state-action pairs under the distribution of states and actions that a policy induces.

Definition 1 (*state-action expectation*) Let $\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$ be any function mapping trajectories to d-dimensional vectors and let π be a policy. The state-action expectation takes the form:

$$\bar{\boldsymbol{\phi}} := \mathbf{E}\left[\phi(S,A) \;\middle|\; S \sim d_\pi, A \sim \pi(\cdot|S)\right] \tag{1}$$

Example 1 Policy Gradient Learning

An example state-action expectation from the reinforcement learning literature is the *policy gradient*. Let $\pi_{\boldsymbol{\theta}}$ be a policy parameterized by the vector $\boldsymbol{\theta}$. Policy gradient algorithms attempt to find $\boldsymbol{\theta}$ that maximize $v(\pi_{\boldsymbol{\theta}})$ with gradient ascent on $v(\pi_{\boldsymbol{\theta}})$ with respect to $\boldsymbol{\theta}$.

$$\frac{\partial}{\partial \boldsymbol{\theta}} v(\pi_{\boldsymbol{\theta}}) \propto \mathbf{E}\left[q^{\pi_{\boldsymbol{\theta}}}(S,A)\frac{\partial}{\partial \boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|S) \;\middle|\; S \sim d_{\pi_{\boldsymbol{\theta}}}, A \sim \pi_{\boldsymbol{\theta}}(\cdot|S)\right] \tag{2}$$

where $q^{\pi_{\boldsymbol{\theta}}}(s,a)$ is an estimate of the sum of rewards following action a in state s. Taking $\phi(s,a) := q^{\pi_{\boldsymbol{\theta}}}(s,a)\frac{\partial}{\partial \boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s)$, we obtain a state-action expectation form.

2.2.2 Trajectory expectations

The second form of expectation we consider is an expectation of a function under the distribution of trajectories the policy will generate.

Definition 2 (*trajectory expectation*) Let $\mathcal{H}$ be the set of all possible trajectories, let $\chi : \mathcal{H} \to \mathbb{R}^d$ be any function mapping trajectories to d dimensional vectors and let π be a policy. The trajectory expectation takes the form:

$$\bar{\chi} := \mathbf{E}\left[\chi(H) \;\middle|\; H \sim \pi\right] \tag{3}$$

Example 2 Policy Evaluation An example from the reinforcement learning literature where evaluating a trajectory expectation is necessary is the problem of *batch policy evaluation* (Thomas and Brunskill 2016a; Jiang and Li 2016). In this problem, we are given a fixed, *evaluation* policy, π_e, and tasked with estimating $v(\pi_e)$. Taking $\chi(h) := g(h)$, we obtain a trajectory expectation.

2.3 The Monte Carlo estimator

Directly evaluating expected values in reinforcement learning is difficult due to the unknown distribution over trajectories or states. Even if these distributions were known, the number of possible states and actions might make analytic computation, as used

in dynamic programming (Bellman 1966), intractable. As an alternative to analytic computation, one of the most straightforward and widely used methods for evaluating expectations in reinforcement learning is the sample average or *Monte Carlo* approach.

Given a set, B, of k state-action pairs, collected by repeatedly sampling $S \sim d_\pi$ and $A \sim \pi(\cdot|S)$, the Monte Carlo estimate for a state-action expectation is:

$$\text{MC}(B) := \frac{1}{k}\sum_{j=1}^{k} \phi(S_j, A_j) \tag{4}$$

Similarly, given a set, D, of m trajectories collected by repeatedly sampling $H \sim \pi$, the Monte Carlo approximation for a trajectory expectation is:

$$\text{MC}(D) := \frac{1}{m}\sum_{j=1}^{m} \chi(H_j) \tag{5}$$

These Monte Carlo estimators are *on-policy* approaches to expectation evaluation; they must use data collected from π to evaluate an expected value under distributions induced by π. We can generalize the Monte Carlo estimator to use data collected from a different *behavior* policy, π_b, by importance sampling. We call the *off-policy* Monte Carlo estimator the *ordinary importance sampling* (OIS) estimator. The OIS estimate for a state-action expectation is:

$$\text{OIS}(B) := \frac{1}{k}\sum_{j=1}^{k} \frac{d_\pi(S_j)\pi(A_j|S_j)}{d_{\pi_b}(S_j)\pi_b(A_j|S_j)}\phi(S_j, A_j). \tag{6}$$

The OIS estimate for a trajectory expectation is:

$$\text{OIS}(D) := \frac{1}{m}\sum_{j=1}^{m} \chi(H_j)\prod_{t=0}^{l-1}\frac{\pi(A_t^j|S_t^j)}{\pi_b(A_t^j|S_t^j)}. \tag{7}$$

Note that d_π is typically unknown and so (6) is *not* directly computable while the OIS estimate for a trajectory expectation is computable. Thus, when we consider state-action expectation evaluation, we will only consider the on-policy case. A recent line of work has explored estimation of the ratio $\frac{d_\pi(s)}{d_{\pi_b}(s)}$ (Liu et al. 2018; Gelada and Bellemare 2019; Hallak and Mannor 2017); this work offers one path towards extending our consideration of state-action expectations to the off-policy setting. When we consider trajectory expectation evaluation, we will also consider the more general off-policy case.

We make the following standard assumptions on the behavior policy.

Assumption 1 (*Full Support*) $\forall s, a\ \pi(a|s) > 0 \Rightarrow \pi_b(a|s) > 0$.

Assumption 2 (*Strong Ignorability*) There are no hidden confounders that influence the choice of actions other than the current observed state.

Assumption 1 is only an assumption on the data generating policy and *not* an assumption on the observed data. For a particular finite sample, there may be actions where $\pi(a|s) > 0$ but (s, a) was never seen.

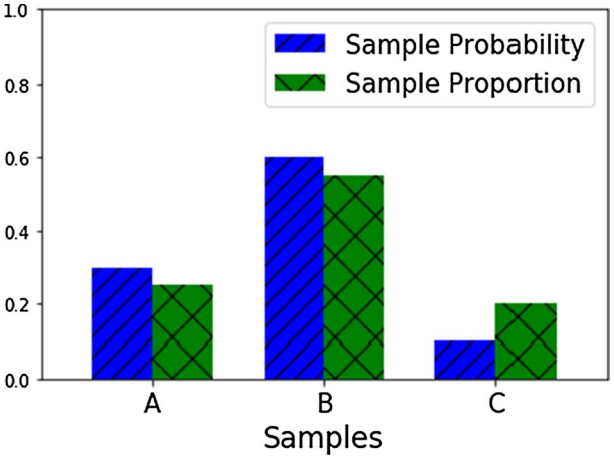


Fig. 2 Sampling error when sampling from a set with three possible samples. Samples are sampled i.i.d. with the given probabilities and are observed in the given proportion. A Monte Carlo estimate will place too much weight on (A), (C) and too little weight on (B)

To address a point of potential confusion, the Monte Carlo return in RL has become synonymous with using the sum of discounted rewards to approximate the return. This approach is typically contrasted with *bootstrapping methods* that truncate the sum of discounted rewards after a number of steps and then add an estimate of the expected reward after truncation to estimate the full return. These bootstrapping methods remain, at least partially, Monte Carlo methods. Thus, the methods we introduce later are of potential value for improving bootstrapping methods, though, we do not study this combination in this work.

2.4 Sampling error in the Monte Carlo estimator

In this section we describe how Monte Carlo estimators can have error for finite sample sizes. We present this discussion in a unified setting that captures both state-action and trajectory expectations.

Let $\mathcal{X}$ be a finite set, $p : \mathcal{X} \to [0, 1]$ be a probability distribution over elements of $\mathcal{X}$, and define $f : \mathcal{X} \to \mathbb{R}$. We assume p is known and f can be evaluated at any $x \in \mathcal{X}$. Suppose that we sample a set of m samples $X = \{X_1, ..., X_m\}$. The expectation, $\bar{f}$, of $f(X)$ with $X \sim p$ is defined as:

$$\bar{f} = \mathbf{E}\left[f(X) \,\middle|\, X \sim p\right] = \sum_{x \in \mathcal{X}} p(x) f(x), \tag{8}$$

and its Monte Carlo approximation is defined as:

$$\mathrm{MC}(X) := \frac{1}{m} \sum_{i=1}^{m} f(X_i). \tag{9}$$

The Monte Carlo approximation weights each $f(x)$ by the frequency at which x occurs in the data. However, this weighting is sub-optimal in that the weights are inaccurate unless we happen to observe each x according to its true probability, $p(x)$.

When the frequency of any element of $\mathcal{X}$ in X is unequal to its expected frequency under p, the Monte Carlo estimator puts either too much or too little weight on that element. We

refer to error due to some elements being either over- or under-represented in the observed data as *sampling error*. Figure 2 illustrates sampling error for $|\mathcal{X}| = 3$.

Sampling error in the Monte Carlo estimator can be viewed as a distribution shift problem; we want to observe samples weighted by p but instead they are weighted by the empirical distribution at which they occur. Let $p_X : \mathcal{X} \to [0, 1]$ be the proportion of times that x occurs in X. Formally, we define $p_X(x) := \frac{c(x)}{m}$ where $c(s)$ is the number of times that we observe x in X. We call p_X the *empirical* distribution of X. Given these definitions, the Monte Carlo estimator can be re-written as:

$$\begin{aligned} \text{MC}(X) &= \frac{1}{m} \sum_{j=1}^{m} f(X_j) \\ &= \frac{1}{m} \sum_{x \in \mathcal{X}} c(x) f(x) \\ &= \sum_{x \in \mathcal{X}} p_X(x) f(x) \\ &= \mathbf{E}\left[f(X) \;\middle|\; X \sim p_X \right] \end{aligned} \tag{10}$$

Notably, the sample average in (9) has been replaced with an exact expectation as in (8). However, the expectation is taken under the empirical distribution p_X and *not* p.

The Monte Carlo estimator is an unbiased estimator of the true value of the expectation (Hammersley and Handscomb 1964, Chapter 2). That is, if we were to repeatedly sample batches of data and compute the estimate, the estimates would be correct in expectation. However, once a single batch of data has been collected, we might ask, "can we correct for the sampling error observed in this fixed sample?"

In fact, (10) suggests a simple solution to correcting sampling error. If the Monte Carlo weights samples according to the empirical distribution, we need only apply importance sampling to correct from the empirical distribution, p_X, to the distribution of interest, p. Previous work in the causal inference (Rosenbaum 1987; Hirano et al. 2003) and Monte Carlo integration literature (Henmi et al. 2007) has shown such an approach to be effective at improving Monte Carlo estimators. However in RL, p is unknown for both state-action expectations and trajectory expectations and thus we cannot compute the numerator of the importance weight. Thus a direct application of previous research is impossible. In the following sections we show that, as long as we know the policy, we can still use importance sampling to partially correct sampling error.

3 Correcting sampling error in state-action expectations

We now introduce the first contribution of this work: a new estimator for on-policy, state-action expectations that corrects sampling error by importance sampling with an estimated behavior policy. The inspiration for this method comes from the view, presented in the previous section, that sampling error in a Monte Carlo estimate can be viewed as distribution shift—we are interested in an expectation weighting samples by their true distribution but instead have an expectation weighting samples by their empirical distribution. We call this new estimator the *sampling error corrected* (SEC) estimator. In this section and the following section, we only consider state-action expectations and the on-policy case; in Sect. 5 we will again consider trajectory expectations and discuss the off-policy case.

We assume that, in addition to the observed data B, we are given a set of policies, Π where each $\pi' \in \Pi$ is a Markovian policy, $\pi' : \mathcal{S} \times \mathcal{A} \to [0, 1]$. The SEC estimator first estimates $\hat{\pi}$ so that $\hat{\pi}$ is the maximum likelihood policy under the observed data:

$$\hat{\pi} := \underset{\pi' \in \Pi}{\operatorname{argmax}} \sum_{j=1}^{k} \log \pi'(A_j|S_j). \tag{11}$$

For many RL problems, (11) can be formulated as a supervised learning problem.

After estimating $\hat{\pi}$, the SEC estimator computes the estimate:

$$\text{SEC}(B) := \frac{1}{k} \sum_{j=1}^{k} \frac{\pi(A_j|S_j)}{\hat{\pi}(A_j|S_j)} \phi(S_j, A_j). \tag{12}$$

This estimate is similar to the Monte Carlo estimate (4) except each $\phi(S_i, A_i)$ is re-weighted by the ratio of the true likelihood $\pi(A_i|S_i)$ to the estimated empirical likelihood $\hat{\pi}(A_i|S_i)$. Intuitively, when an action is sampled more often than its expected frequency, SEC decreases the weight on that action. When an action is sampled less often than its expected frequency, SEC increases the weight on that action. Importantly, SEC estimates $\hat{\pi}$ with the same k samples that will be used to compute the estimate. If $\hat{\pi}$ is estimated with a different set of samples then $\hat{\pi}$ will contain no information for correcting sampling error in B.

Recall from the previous section that when the domain of samples is finite, the batch Monte Carlo estimator can be written as an exact expectation taken under the empirical distribution of samples. The same is true for the Monte Carlo estimator when estimating state-action expectations. Let $d_B(s) := \frac{c(s)}{k}$ and $\pi_B(a|s) = \frac{c(s,a)}{c(s)}$ where $c(s)$ is the number of times that state s appears in B and $c(s, a)$ is the number of times that action a occurred in state s in B. The Monte Carlo estimator can be written as:

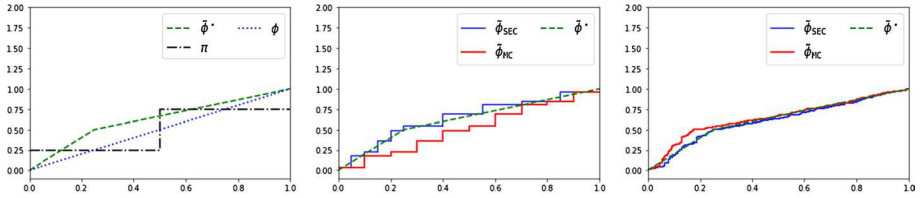
$$\text{MC}(B) = \frac{1}{k} \sum_{j=1}^{k} \phi(S_j, A_j) = \mathbf{E}\left[\phi(S, A) \;\middle|\; S \sim d_B, A \sim \pi_B(\cdot|S)\right]. \tag{13}$$

Suppose we learn $\hat{\pi}$ such that $\hat{\pi}(a|s) = \pi_B(a|s)$ for all s, a occurring in the realization of B. In this case,

$$\text{SEC}(B) = \frac{1}{k} \sum_{j=1}^{k} \frac{\pi(A_j|S_j)}{\pi_B(A_j|S_j)} \phi(S_j, A_j) = \mathbf{E}\left[\phi(S, A) \;\middle|\; S \sim d_B, A \sim \pi(\cdot|S)\right]. \tag{14}$$

Equation (14) shows that the SEC estimator can also be written as an exact expectation but the action weighting is now under π instead of π_B. The state weighting is still that of d_B; since d_π is unknown we are only able to correct sampling error due to sampling from the policy. Equation 14 demonstrates an equivalence between SEC and analytic expectation methods [e.g., all-action policy gradients (Sutton et al. 2000)] in discrete action spaces. In the following subsection we discuss a different intuition for SEC in continuous action spaces where analytic expectation methods are more challenging to apply.

Despite the use of importance sampling, we introduce SEC as an on-policy only estimator. In the off-policy setting, importance sampling corrects from the distribution that actions were sampled from to the distribution of actions under the policy of interest. SEC uses importance sampling to correct from the empirical distribution of actions to the distribution of actions under the policy of interest. SEC could possibly be extended to the



(a) Policy and ϕ **(b)** 10 Sample Approx. **(c)** 200 Sample Approx.

Fig. 3 Expectation evaluation in a continuous armed bandit task. **a** A reward function, $\phi(a) := a$, and the probability density function of a policy, π, with support on the range [0, 1]. With probability 0.25, π selects an action less than 0.5 with uniform probability; otherwise π selects an action greater than 0.5 with uniform probability. All figures show $\tilde{\phi}^\star$: a version of ϕ that is stretched according to the density of π; since the range [0.5, 1] has probability 0.75, ϕ on this interval is stretched over [0.25, 1]. **b**, **c** $\tilde{\phi}^\star$ and the piece-wise $\tilde{\phi}_{\text{MC}}$ and $\tilde{\phi}_{\text{SEC}}$ approximations to $\tilde{\phi}^\star$ after 10 and 200 samples respectively. SEC counts the frequency that action fall into the bins $a \leq 0.5$ or $a > 0.5$ to form its empirical estimate of π

off-policy setting by combining it with a method that estimates the state density ratio $\frac{d_\pi(s)}{d_{\pi_b}(s)}$ (Liu et al. 2018). However, this combination is outside the scope of this article.

3.1 Correcting sampling error with continuous actions

In the previous subsection, we discussed how SEC corrects for sampling error in finite MDPs. Here, we discuss how SEC corrects for sampling error in MDPs with continuous-valued action sets. The primary purpose of this discussion is to build intuition and we limit discussion to a setting that can be easily visualized. Specifically, we consider a multi-armed bandit problem with scalar, real-valued actions. We wish to estimate the expectation of function $\phi : \mathcal{A} \to \mathbb{R}$ under policy π which we assume to have bounded support in [0, 1]:

$$\bar{\phi} = \mathbf{E}[\phi(A)|A \sim \pi] = \int_0^1 \phi(a)\pi(a)da. \tag{15}$$

The Monte Carlo estimate of this expectation with k samples from π is:

$$\text{MC}(B) = \frac{1}{k}\sum_{i=1}^{k} \phi(A_i). \tag{16}$$

Even though the Monte Carlo estimate is a sum over a finite number of samples, we show it is exactly equal to an integral over a particular piece-wise function. We assume (w.l.o.g) that the A_i's are in non-decreasing order ($A_0 <= A_i <= A_m$). Imagine that we divide the range [0, 1] into k equal bins. We now define piece-wise constant function $\tilde{\phi}_{\text{MC}}$ where $\tilde{\phi}_{\text{MC}}(a) = \phi(A_i)$ if a is in the ith bin. The Monte Carlo estimate is exactly equal to the integral $\int_0^1 \tilde{\phi}_{\text{MC}}(a)da$.

It would be reasonable to assume that $\tilde{\phi}_{\text{MC}}(a)$ is approximating $\phi(a)\pi(a)$ since the Monte Carlo estimate (16) is approximating (15), i.e., $\lim_{m\to\infty} \tilde{\phi}_{\text{MC}}(a) = \phi(a)\phi(a)$. In reality, $\tilde{\phi}_{\text{MC}}$ approaches a *stretched* version of ϕ where areas with high density under π are stretched and areas with low density are contracted. We call this stretched version of ϕ, $\tilde{\phi}^\star$. The integral of $\int_0^1 \tilde{\phi}^\star(a)da$ is exactly the true expected value, $\bar{\phi}$.

Figure 3a gives a visualization of an example $\tilde{\phi}^\star$ using on-policy Monte Carlo sampling from an example π and linear ϕ. In contrast to the true $\tilde{\phi}^\star$, the Monte Carlo approximation to $\tilde{\phi}$, $\tilde{\phi}_{\text{MC}}$ stretches ranges of ϕ according to the number of samples in that range: ranges with many samples are stretched and ranges without many samples are contracted. As the sample size grows, any range of ϕ will be stretched in proportion to the probability of getting a sample in that range. For example, if the probability of drawing a sample from $[a, b]$ is 0.5 then $\tilde{\phi}^\star$ stretches ϕ on $[a, b]$ to cover half the range [0, 1]. Figure 3 visualizes $\tilde{\phi}_{\text{MC}}$ the Monte Carlo approximation to $\tilde{\phi}^\star$ for sample sizes of 10 and 200.

In this analysis, sampling error corresponds to over-stretching or under-stretching ϕ in any given range. The limitation of Monte Carlo sampling can then be expressed as follows: given π, we know the correct amount of stretching for any range and yet the Monte Carlo estimator ignores this information and stretches based on the empirical proportion of samples in a particular range. On the other hand, SEC first divides by the empirical probability density function (pdf) (approximately undoing the stretching from sampling) and then multiplies by the true pdf to more correctly stretch ϕ. Figure 3 also visualizes the $\tilde{\phi}_{\text{SEC}}$ approximation to $\tilde{\phi}^\star$ for sample sizes of 10 and 200. In this figure, we can see that $\tilde{\phi}_{\text{SEC}}$ is a closer approximation to $\tilde{\phi}^\star$ than $\tilde{\phi}_{\text{MC}}$ for both sample sizes. In both instances, the squared error of the SEC estimate is less than that of the Monte Carlo estimate.

Since ϕ may be unknown until sampled, we will still have non-zero error. However the Monte Carlo estimate has error due to *both* sampling error and unknown ϕ values. SEC has error only due to the unknown ϕ values for actions that remain unsampled.

3.2 Theoretical analysis

In this section we establish theoretical properties of the SEC estimator. Since SEC is a biased estimator of $\bar{\phi}$, the most important properties to establish are consistency and lower variance compared to the Monte Carlo estimator. In the following subsections we establish consistency and asymptotically lower variance under a set of general assumptions. Under a set of stronger assumptions, we show that the variance of the SEC estimator will always be at most that of the Monte Carlo estimator. To the best of our knowledge, the only prior theoretical work on importance sampling with an estimated behavior policy for state-action expectations is the variance and bias results of Dudík et al. (2011) for contextual bandits. However, this prior work made the assumption that $\hat{\pi}$ is estimated independently of the data used to compute the estimate of $\bar{\phi}$ and is thus inapplicable to SEC.

3.2.1 Consistency

We prove that the SEC estimator is a consistent estimator of $\bar{\phi}$ under the following assumption:

Assumption 3 (Consistent estimation of $\hat{\pi}$)

$$\operatorname*{argmax}_{\pi \in \Pi} \sum_{j=1}^{k} \log \pi(A_j|S_j) \xrightarrow{a.s.} \pi$$

where $\xrightarrow{a.s.}$ denotes almost sure convergence.

This assumption is fairly easy to satisfy assuming that the true policy, π, is included in Π and the log likelihood and estimated log likelihood satisfy smoothness assumptions with respect to Π. We discuss these mild assumptions further in "Appendix 1" when we provide a full proof of Proposition 1.

Proposition 1 *Under Assumption 3, the SEC estimator is a consistent estimator of* $\bar{\phi}$:

$$\text{SEC}(D) \xrightarrow{a.s.} \bar{\phi}.$$

Proof See "Appendix 1".

3.2.2 Asymptotic variance

Consistency is an important property as it establishes the asymptotic correctness of an estimator. We next establish an ordering between the variances of the SEC and Monte Carlo estimators. In this section, we show that the *asymptotic variance* of the SEC estimator is at most that of the Monte Carlo estimator when π and $\hat{\pi}$ both belong to the same parametric family. This result is a corollary to an existing result in the Monte Carlo integration literature (Henmi et al. 2007) and is shown under the following assumptions:

Assumption 4 The policy set, Π is a set of policies parameterized by a vector $\boldsymbol{\theta}$ and all policies $\pi_{\boldsymbol{\theta}} \in \Pi$ are twice differentiable with respect to $\boldsymbol{\theta}$.

Assumption 5 Policy π is in the parameterized set of policies considered by SEC. $\exists \tilde{\boldsymbol{\theta}}$ such that $\pi_{\tilde{\theta}} \in \Pi$ and $\pi_{\tilde{\theta}} = \pi_b$.

These assumptions cover widely used choices of policy approximation such as neural networks and linear functions. Under these assumptions, we prove Corollary 1:

Corollary 1 *Let* $\text{Var}_{\text{A}}(\text{EST})$ *denote the asymptotic variance of estimator* EST. *Under Assumptions 4 and 5,*

$$\text{Var}_{\text{A}}(\text{SEC}) \leq \text{Var}_{\text{A}}(\text{MC}).$$

Proof See "Appendix 3".

3.2.3 Variance

Corollary 1 is derived under a set of mild assumptions. With more restrictive assumptions we can compare the variance of the two estimators in the non-asymptotic case. This analysis is done under the following assumptions:

Assumption 6 The action space is discrete and if a state is observed then all actions have also been observed in that state.

Assumption 7 For all observed states, the estimated policy $\hat{\pi}$ is equal to π_B, i.e., if action a occurs k times in state s and s occurs n times in B then $\hat{\pi}(a|s) = \frac{k}{n}$.

These more restrictive assumptions are only made for the proof of Proposition 2.

Proposition 2 *Let* Var(EST) *denote the variance of estimator* EST. *Under Assumptions* 6 *and* 7, *for the Monte Carlo estimator*, MC, *and the SEC estimator*, SEC:

$$\text{Var}(\text{SEC}(B)) \leq \text{Var}(\text{MC}(B))$$

Proof The full proof is provided in "Appendix 4".

4 Empirical study: state-action expectations

We have introduced the SEC estimator as a general estimator for state-action expectations in reinforcement learning. In order to empirically evaluate the SEC estimator, we apply the general estimator to the problem of estimating the policy gradient for use in a policy gradient algorithm. Specifically, we focus on *batch* policy gradient algorithms that repeatedly collect a batch of on-policy trajectories, estimate the policy gradient, update the policy, and then discard previously collected data to collect more trajectories for the next update. We show that variants of *trust-region policy optimization* (TRPO) (Schulman et al. 2015) and *REINFORCE* (Williams 1992) that use the SEC estimator converge faster than their counterparts that use the Monte Carlo estimator.

Recall from Sect. 2.2.1 that in policy gradient reinforcement learning, a parameterized policy $\pi_{\boldsymbol{\theta}}$ is updated with stochastic gradient ascent, using the gradient of its expected return:

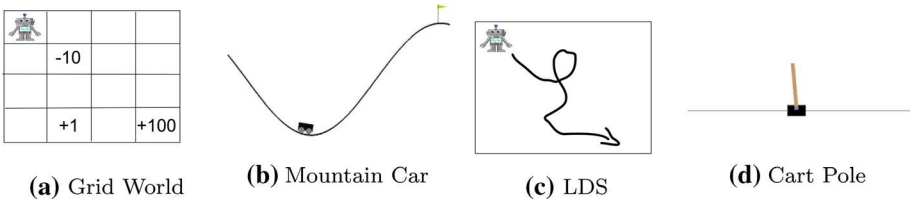
$$\frac{\partial}{\partial \boldsymbol{\theta}} v(\pi_{\boldsymbol{\theta}}) \propto \mathbf{E}\left[q^{\pi_{\boldsymbol{\theta}}}(S,A) \frac{\partial}{\partial \boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A|S) \;\middle|\; S \sim d_{\pi_{\boldsymbol{\theta}}}, A \sim \pi_{\boldsymbol{\theta}} \right]. \tag{17}$$

The SEC estimator for the right-hand side of (17) is given as:

$$\text{SEC}(B) := \frac{1}{k} \sum_{j=1}^{k} \frac{\pi_{\boldsymbol{\theta}}(A_j|S_j)}{\hat{\pi}(A_j|S_j)} \hat{q}^{\pi_{\boldsymbol{\theta}}}(S_j,A_j) \frac{\partial}{\partial \boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(A_j|S_j). \tag{18}$$

where $\hat{q}^{\pi_{\boldsymbol{\theta}}}$ is an estimate of $q^{\pi_{\boldsymbol{\theta}}}$. In Algorithm 1 we provide pseudocode for a generic batch policy gradient algorithm using the SEC estimator. Having instantiated the SEC estimator for batch policy gradient learning, we now conduct an empirical study comparing the SEC policy gradient estimator to the Monte Carlo policy gradient estimator. Our experiments are designed to answer the questions:

1. Does the SEC policy gradient estimator lead to faster convergence for batch policy gradient algorithms compared to the Monte Carlo estimator?
2. Does the SEC estimator reduce variance by correcting sampling error?



(a) Grid World **(b)** Mountain Car **(c)** LDS **(d)** Cart Pole

Fig. 4 Illustrations of the domains used in our experiments. LDS is short for linear dynamical system

Algorithm 1 Sampling Error Corrected Batch Policy Gradient
Input: Initial policy parameters, $\boldsymbol{\theta}_0$, batch size k, a step-size for each iteration, α_i, and number of iterations n.
Output: Optimized policy parameters $\boldsymbol{\theta}_n$.

1: **for all** $i = 0$ to n **do**
2: Sample k steps $(S, A) \sim \pi_{\boldsymbol{\theta}_i}$
3: $\hat{\pi}_i \leftarrow \underset{\pi'}{\text{argmax}} \sum_{j=1}^{k} \log \pi'(a_j|s_j)$
4: $g_{\text{sec}} \leftarrow \frac{1}{k} \sum_{j=1}^{k} \frac{\pi_{\boldsymbol{\theta}}(a_j|s_j)}{\hat{\pi}_i(a_j|s_j)} \widehat{q}^{\pi_{\boldsymbol{\theta}}}(s_j, a_j, \cdot) \frac{\partial}{\partial \boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}_i}(a_j|s_j)$
5: $\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i + \alpha_i g_{\text{sec}}$
6: **end for**
7: **Return** $\boldsymbol{\theta}_n$

4.1 Empirical set-up: state-action expectations

In each RL task that we consider we choose a policy gradient algorithm (either REINFORCE or TRPO) and evaluate the number of policy update steps until convergence for a variant that uses the SEC estimator as compared to a variant that used the Monte Carlo estimator. For each task and each algorithm variant we run a series of trials where a single trial consists of a fixed number of policy updates. The policy gradient algorithms considered require an estimate of $q^{\pi_\theta}(s, a)$ for any s, a that are observed in B. We use the sum of discounted rewards following action a in state s as an estimate of $q^{\pi_\theta}(s, a)$. We also use a state-dependent baseline, $v^{\pi_\theta}(s)$, as is common in the policy gradient literature (Greensmith et al. 2004; Schulman et al. 2016; Williams 1992).

We next describe four reinforcement learning tasks, the empirical set-up for each task, and the motivation for evaluating SEC in these domains. Figure 4 displays images of these domains.

4.1.1 Grid World

Our first domain is a 4×4 Grid World and we use REINFORCE (Williams 1992) as the underlying batch policy gradient algorithm. The agent begins in grid cell (0, 0) and trajectories terminate when it reaches (3, 3). The agent receives a reward of 100

at termination, -10 at (1, 1) and -1 otherwise. The agent's policy is a state-dependent softmax distribution over actions:

$$\pi_{\boldsymbol{\theta}}(a|s) = \frac{e^{\theta_{s,a}}}{\sum_{a' \in \mathcal{A}} e^{\theta_{s,a'}}}.$$

With this representation, the policy does *not* generalize across states or actions.

The SEC estimator estimates the policy by counting how many times each action is taken in each state. This domain closely matches the assumptions made in our theoretical analysis. Specifically, the action set is finite and $\hat{\pi}$ is exactly equal to π_B. While we do not explicitly enforce the assumption that all actions are observed in all states, the small size of the state and action space ($|\mathcal{S}| = 16$ and $|\mathcal{A}| = 4$) makes it likely that this assumption holds.

In our implementation of REINFORCE, we normalize the gradient estimates by dividing by their magnitudes and use a step-size of 1. At each iteration, each method collects a batch of 10 trajectories with the current policy.

4.1.2 Tabular Mountain Car

Our second domain is a discretized version of the classic Mountain Car domain (Moore 1990; Singh and Sutton 1996), where an agent attempts to move an under-powered car up a steep hill by accelerating to the left or right or not accelerating. The original task has a state of the car's position (a continuous scalar in the range $[-1.2, 0.6]$) and velocity (a continuous scalar in the range $[-0.07, 0.07]$). Following Jiang and Li (2016), we discretize position into 6 bins and velocity into 8 bins for a total of 4292 states. We use the discretized version of the task because the large number of discrete states makes it unlikely that all actions are observed in all visited states (in violation of Assumption 6). The domain does still match the assumptions in Sect. 3.2.3 in that the action set is finite and the estimated behavior policy is exactly equal to π_B.

We again use REINFORCE as the batch policy gradient algorithm. The agent's policy is a state-dependent softmax distribution over the three discrete actions as is used in the Grid World domain. The SEC estimator estimates the policy using the empirical proportion of times that each action is taken in each state.

As in Grid World we normalize the gradient estimates by dividing by their magnitudes and use a step-size of 1. We run each method with batch sizes of 100, 200, 600, and 800 trajectories.

4.1.3 Linear dynamical system

Our third domain is a two-dimensional linear dynamical system in which we evaluate SEC when actions are real-valued vectors. The reward is the agent's distance to the origin and trajectories last for 20 time-steps. In this domain the learning agent observes horizontal and vertical position and velocity and uses a linear Gaussian policy to select continuous valued accelerations in the horizontal and vertical direction:

$$\pi(\cdot|s) := \mathcal{N}(\mu(s), \boldsymbol{\theta}_\sigma)\mu(\mathbf{s}) := \mathbf{s} \cdot \boldsymbol{\theta}_w + \boldsymbol{\theta}_b,$$

where $\boldsymbol{\theta}_\sigma$, $\boldsymbol{\theta}_w$, and $\boldsymbol{\theta}_b$ are the policy parameters, $\boldsymbol{\theta}$. We use the OpenAI Baselines (Dhariwal et al. 2017) implementation of TRPO as the underlying batch policy gradient algorithm. We set the generalized advantage estimation (Schulman et al. 2016) parameters (γ, λ) both

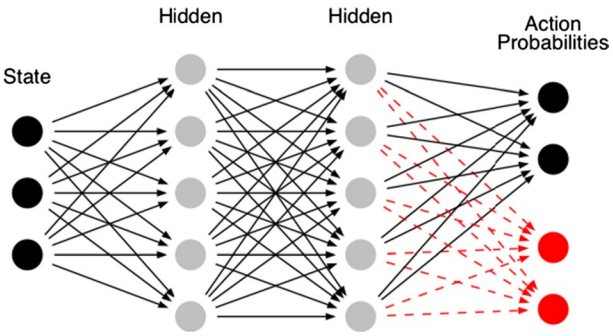


Fig. 5 A simplified version of the neural network architecture used in Cart Pole. The true architecture has 32 hidden units in each layer. The current policy π_θ is given by a neural network that outputs the action probabilities as a function of state (black nodes). The estimated policy, $\hat{\pi}$, is a linear policy that takes as input the activations of the final hidden layer of π_θ. Only the weights on the red, dashed connections are changed when estimating $\hat{\pi}$

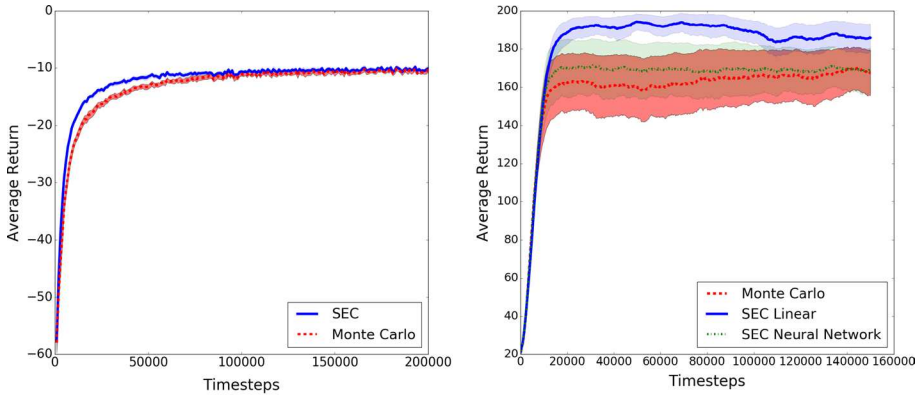
to 1. We estimate $\hat{\pi}$ with ordinary least squares and estimate a state-independent variance parameter. In this domain, none of our theoretical assumptions hold: the action and state sets are infinite and $\hat{\pi} \neq \pi_B$. We include it to evaluate SEC with simple function approximation. At each iteration, we use a batch size of 1000 time-steps and set the TRPO KL-divergence constraint, $\epsilon = 0.01$.

4.1.4 Cart Pole

Our final domain is the Cart Pole domain from OpenAI Gym (Brockman et al. 2016) and we again use TRPO as the underlying batch policy gradient algorithm. At each iteration, we run the current policy for 200 steps and set the KL-divergence constraint, $\epsilon = 0.001$. The policy representation is a two layer neural network with 32 hidden units in each layer where $\boldsymbol{\theta}$ consists of the weights and biases of the network. The input to the policy is the position and velocity of the cart and the angle and angular velocity of the pole. The output of the network is the parameters of a softmax distribution over the two actions. Estimating $\hat{\pi}$ is equivalent to learning a soft classifier that attempts to classify what action π_θ would take in a given state. We consider two parameterizations of Π:

1. Each $\pi \in \Pi$ is a neural network with the same architecture as π_θ. We learn $\hat{\pi}$ with gradient descent, using all data in B to estimate the gradient. We refer to this method as SEC Neural Network.
2. Each $\pi \in \Pi$ is a linear function that receives the activations of the last hidden layer of π_θ as input. The dual $\hat{\pi}$ and π_θ architecture is shown in Fig. 5. We estimate the weights of $\hat{\pi}$ with gradient descent, using all data in B to estimate the gradient. This method is labeled SEC Linear.

Again, this domain violates all assumptions made in our theoretical analysis. We include this domain to study SEC with more complex function approximation. This setting allows us to study SEC with neural network policies but is simple enough to avoid extensive tuning of hyper-parameters.



(a) LDS **(b)** Cart Pole

Fig. 6 Learning results for the Linear Dynamical System (LDS) and Cart Pole domains. The horizontal is the number of timesteps and the vertical axis is the average return of a policy. We run 25 trials of each method using different random seeds. The shaded region represents a 95% confidence interval. In both domains we see that all variants of sampling error corrected policy gradient outperform the batch Monte Carlo policy gradient in either time to optimal convergence or final performance

4.2 Empirical results: state-action expectations

We now present our empirical results for estimating state-action expectations with the SEC estimator.

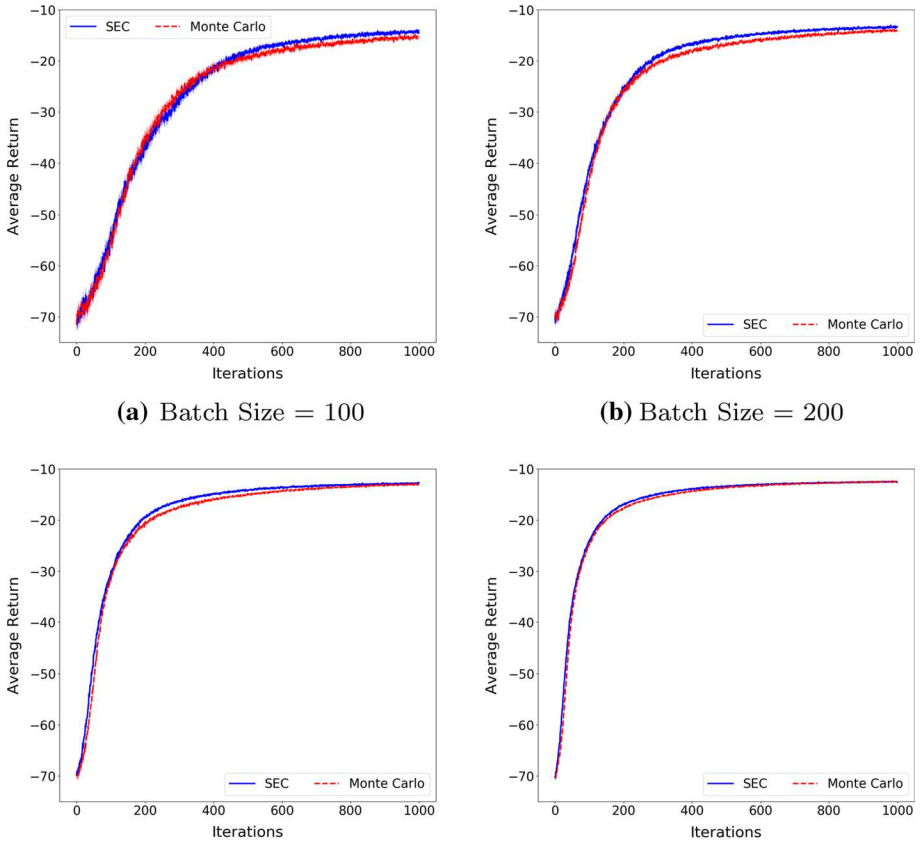
4.2.1 Main results

Results for the Linear Dynamical System (lds), and Cart Pole environment are given in Fig. 6. In both domains, we see that the SEC methods lead to a learning speed-up compared to the Monte Carlo based approaches. In the LDS domain, SEC outperforms Monte Carlo in time to convergence to optimal. In Cart Pole, both variants of SEC learn faster initially, however, Monte Carlo catches up to the neural network version of SEC. This result demonstrates that we can leverage intermediate representations of π_θ (in this case, the activations of the final hidden layer) to learn $\hat{\pi}$ with a simpler model class. In fact, results suggest that fitting a simpler model improves performance.

4.2.2 Tabular Mountain Car

We also compare SEC to Monte Carlo in the Mountain Car domain. We run our experiments four times with a different batch size in each experiment. Each experiment consists of 25 trials for each algorithm.

Figure 7 shows results for each of the different tested batch sizes. For each batch size, we can see that SEC improves upon the Monte Carlo approach. The relative improvement does change across batches. With the largest batch size, improvement is marginal as the large batch size means that the Monte Carlo estimate will have low variance. For the smallest batch size, improvement is again marginal—though the small batch size means Monte Carlo has higher variance, it also means that SEC may have higher bias as some actions will be unobserved in visited states. Intermediate batch sizes have the widest gap between



(a) Batch Size = 100 **(b)** Batch Size = 200

(c) Batch Size = 600 **(d)** Batch Size = 800

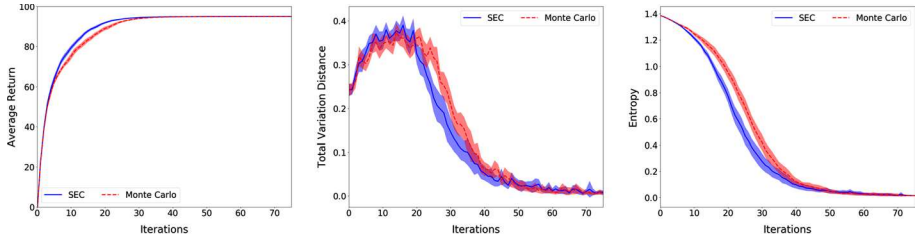
Fig. 7 Learning results for the Mountain Car domain with different batch sizes. The horizontal axis is the number of iterations (i.e., the number of times the policy has been updated). The vertical axis is average return. We run 25 trials of each method using different random seeds. The shaded region represents a 95% confidence interval. For all batch sizes we see that the sampling error corrected policy gradient outperforms the batch Monte Carlo policy gradient in either time to optimal convergence or final performance after 1000 iterations

the two methods—the batch size is small enough that Monte Carlo has high variance but that SEC has less bias.

4.2.3 Grid World experiments

Figure 8 shows several results in the Grid World domain. First, Fig. 8a shows that SEC leads to faster convergence compared to Monte Carlo. This domain most closely matches our theoretical assumptions where we showed SEC has lower variance than Monte Carlo estimates. The lower variance translates into faster learning.

We also use the Grid World domain to perform a quantitative evaluation of sampling error. As a measure of sampling error we use the total variation distance between the current policy π_θ and the empirical frequency of actions, π_B. For any state s, the total variation distance between the two policies is given by:



(a) Average Return **(b)** Sampling Error **(c)** Entropy

Fig. 8 Sampling error corrected policy gradient in the Grid World Domain. **a** The average return for SEC and Monte Carlo. **b** The total variation distance between the current policy and estimated policy at each iteration. **c** Policy entropy at each iteration. Results are averaged over 25 trials and confidence bars are for a 95% confidence interval

$$D_{\text{TV}}(\pi_\theta(\cdot|s), \pi_B(\cdot|s)) := \sum_{a \in \mathcal{A}} |\pi_\theta(a|s) - \pi_B(a|s)|.$$

We report the mean D_{TV} value over states in B as a measure of sampling error. We choose the total variation distance as opposed to the more commonly used KL-divergence since π_B and π_θ may not share support. That is, there may be an action, a, where $\pi_B(a|s)$ is 0 and $\pi_\theta(a|s) > 0$.

Figure 8b shows that sampling error increases and then decreases during learning. Peak sampling error correlates with where the learning curve gap between the two methods is greatest. Note that sampling error naturally decreases as learning converges because the policy becomes more deterministic. Figure 8c shows that the entropy of the current policy goes to zero, i.e., becomes more deterministic. A more deterministic policy will have less sampling error and so we expect to see less advantage from SEC as learning progresses.

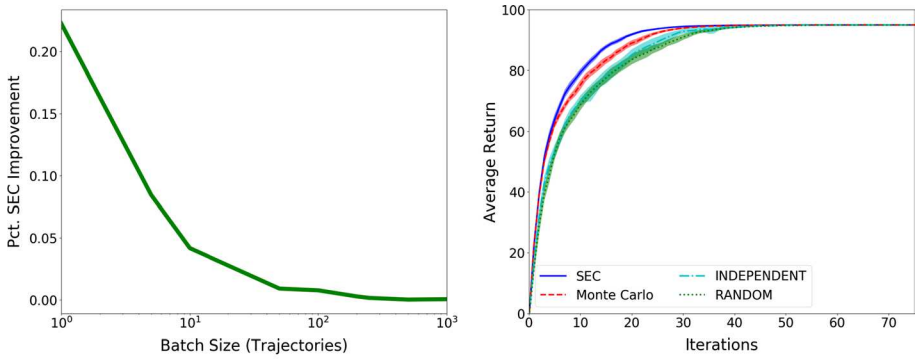
We also perform a sensitivity analysis of SEC to the batch-size at each iteration. We run 10 trials each of the SEC and Monte Carlo policy gradient algorithms with batch-sizes from 1 to 1000 trajectories. For each method and batch-size we compute the mean area-under-the-curve (AUC) for the average return up to iteration 20 (close to where learning converges). We then compute the relative improvement of SEC compared to Monte Carlo for each batch-size as:

$$\texttt{PctImprove} := \frac{\texttt{AUC}_{\texttt{SEC}} - \texttt{AUC}_{\texttt{MC}}}{\texttt{AUC}_{\texttt{MC}}}.$$

Figure 9a shows that the performance improvement is greatest when the batch-size is small and decreases as the batch-size grows. When the batch-size is small, the Monte Carlo policy gradient will have the highest sampling error and thus SEC has the most room for improvement. As the batch-size grows, sampling error decreases and the SEC improvement is more marginal.

Finally, we verify the importance of using the same data to both estimate $\hat{\pi}$ and estimate the policy gradient. Figure 9b introduces two alternatives to SEC:

- Independent: Estimates $\hat{\pi}$ with a separate set of k samples and then compute the SEC estimate using this $\hat{\pi}$.



(a) Batch Size Sensitivity **(b)** Alternative Weightings

Fig. 9 Sampling error corrected policy gradient ablations in the Grid World Domain. **a** The percent improvement of SEC compared to Monte Carlo for varying batch sizes. For each batch size, we compute area under the average return curve (AUC) for each method during the first 20 learning iterations. We compute the mean AUC over 10 trials and report the percent improvement of the SEC mean over Monte Carlo. **b** Average return for two alternative weight corrections. Results are averaged over 25 trials and confidence bars are for a 95% confidence interval

- Random: Instead of computing importance weights, we randomly sample weights from a normal distribution and use them in place of the learned SEC weights. The normal distribution has mean one and standard deviation chosen to approximately match the range of weights seen when using the SEC estimator.

Figure 8a shows that independent hurts performance compared to Monte Carlo. random performs marginally worse than Monte Carlo. This result demonstrates the need to use the same set of data to estimate $\hat{\pi}$ and compute the SEC estimate.

To conclude our empirical study of the SEC estimator for state-action expectations, we have shown that correcting sampling error with the SEC estimator can decrease the number of policy updates needed for a batch policy gradient algorithm to converge. This empirical study focused on using SEC to lower the variance of policy gradient estimates compared to a Monte Carlo estimator. However, SEC is a general estimator for any reinforcement learning problem that requires estimating a state-action expectation and is thus potentially applicable to other problems, for example, policy evaluation in average reward reinforcement learning. Unfortunately, not all expectations in reinforcement learning can be easily written as state-action expectations. In the next section, we describe how to correct sampling error when estimating trajectory expectations.

5 Correcting sampling error in trajectory expectations

In this section we introduce the second contribution of this article: a family of estimators called *regression importance sampling* (RIS) estimators that correct for sampling error in the set of observed trajectories, D, by importance sampling with an estimated behavior policy. In contrast to SEC that corrects sampling error when estimating state-action expectations with on-policy data, RIS estimators correct sampling error for estimating trajectory

expectations with either on-policy or off-policy data. Since we consider both the on- and off-policy cases, we will discuss the RIS estimator relative to the ordinary importance sampling (OIS) estimator that generalizes the Monte Carlo estimator to the off-policy setting (see Sect. 2).

As with SEC, we assume that, in addition to D, we are given a set of policies. Unlike SEC, we assume this set, Π^n, (possibly) contains non-Markovian policies: each $\pi \in \Pi^n$ is a distribution over actions conditioned on the immediate preceding state and the last n states and actions preceding that state: $\pi : \mathcal{S}^{n+1} \times \mathcal{A}^n \to [0, 1]$. The RIS($n$) estimator first estimates the maximum likelihood behavior policy in Π^n under D:

$$\hat{\pi}^{(n)} := \operatorname*{argmax}_{\pi \in \Pi^n} \sum_{i=1}^{m} \sum_{t=0}^{l-1} \log \pi(A_t^i | H_{t-n:t}^i). \tag{19}$$

When $n = 0$, RIS and SEC return the same $\hat{\pi}$. The RIS(n) estimate is then an OIS estimate with $\hat{\pi}^{(n)}$ replacing π_b.

$$\text{RIS}(n)(\pi, D) := \frac{1}{m} \sum_{i=1}^{m} \chi(H_i) \prod_{t=0}^{l-1} \frac{\pi(A_t^i | S_t^i)}{\hat{\pi}^{(n)}(A_t^i | H_{t-n:t}^i)} \tag{20}$$

We refer to $\frac{\pi(A_t|S_t)}{\hat{\pi}^{(n)}(S_t|H_{t-n:t})}$ as the RIS(n) weight for action A_t, state S_t, and trajectory segment $H_{t-n:t}$. Though RIS(0) and SEC would return the same $\hat{\pi}$, RIS(0) corrects sampling error along the entire trajectory since it uses the product of importance weights.

We have introduced RIS as a family of estimators where different RIS methods estimate the empirical behavior policy conditioned on different history lengths. Among these estimators, our primary method of study is RIS(0). For larger n, RIS(n) may be less reliable for small sample sizes as the $\hat{\pi}^{(n)}$ estimate will be highly peaked (it will be 1 for most observed actions.) We verify this claim empirically below. However, as we discuss in Sect. 6.2.2, larger n may produce asymptotically more accurate sampling error corrections and thus asymptotically more accurate estimates.

5.1 Correcting sampling error in discrete action spaces

We now present an example illustrating how RIS corrects for sampling error when used to estimate trajectory expectations. Our goal in this section is to build intuition and we make several limiting assumptions to facilitate presentation. These assumptions are removed for our more formal theoretical and empirical analysis and should *not* be understood as limitations of RIS methods. We make the following assumptions:

1. $\mathcal{S}$ and $\mathcal{A}$ are finite sets.
2. The distributions d_0 and P are deterministic, that is, $d_0(s) = 1$ for only one $s \in \mathcal{S}$ and for all s, a, $P(s'|s, a) = 1$ for only one $s' \in \mathcal{S}$.
2. Let $\mathcal{H}$ be the (finite) set of possible trajectories under behavior policy, π_b. We assume that our observed data, D, contains at least one of each $h \in \mathcal{H}$.

We define $c(h_{i:j})$ as the number of times that trajectory segment $h_{i:j}$ appears during any trajectory in D. Similarly, we define $c(h_{i:j}, a)$ as the number of times that action a is observed following trajectory segment $h_{i:j}$ during any trajectory in D. RIS(n) estimates the empirical behavior policy as:

$$\hat{\pi}(a|h_{i-n:i}) := \frac{c(h_{i-n:i}, a)}{c(h_{i-n:i})}.$$

Observe that both OIS and all variants of RIS can be written in one of two forms:

$$\underbrace{\frac{1}{m}\sum_{i=1}^{m}\frac{w_\pi(H_i)}{w_{\pi'}(H_i)}\chi(H_i)}_{(i)} = \underbrace{\sum_{h\in\mathcal{H}}\frac{c(h)}{m}\frac{w_\pi(h)}{w_{\pi'}(h)}\chi(h)}_{(ii)}$$

where $w_{\pi'}(h) = \prod_{t=0}^{l-1}\pi'(a_t|s_t)$ and for OIS, $\pi' := \pi_b$ and for RIS(n), $\pi' := \hat{\pi}^{(n)}$ as defined in Eq. (19).

If we had sampled trajectories using $\hat{\pi}^{(l-1)}$ instead of π_b, in a deterministic environment, the probability of each trajectory, h, would be $\Pr(H = h|H \sim \hat{\pi}^{(l-1)}) = \frac{c(h)}{m}$. Thus Form (ii) can be written as:

$$\mathbf{E}\left[\frac{w_\pi(H)}{w_{\pi'}(H)}\chi(H)\,\middle|H \sim \hat{\pi}^{(l-1)}\right].$$

To emphasize what we have shown so far: OIS and RIS are both sample-average estimators whose estimates can be written as exact expectations. However, this exact expectation is under the distribution that trajectories were observed and *not* the distribution of trajectories under π_b. Furthermore, the distribution that trajectories were observed is the trajectory distribution of a non-Markovian behavior policy.

Consider choosing $w_{\pi'} := w_{\pi_b}^{(l-1)}$ as RIS($l-1$) does. This choice results in (ii) being exactly equal to $\mathbf{E}[\chi(H)|H \sim \pi]$.[2] On the other hand, choosing $w_\pi := w_{\pi_b}$ will *not* return $\mathbf{E}[\chi(H)|H \sim \pi]$ unless we happen to observe each trajectory at its expected frequency (i.e., $\hat{\pi}^{(l-1)} = \pi_b$).

Choosing $w_{\pi'}$ to be $w_{\hat{\pi}^{(n)}}$ for $n < l-1$ also does *not* result in $\mathbf{E}[\chi(H)|H \sim \pi]$ being returned in this example. This observation is surprising because even though we know that the true $\Pr(H = h|\pi_b) = \prod_{t=0}^{l-1}\pi_b(a_t|s_t)$, it does not follow that the estimated probability of a trajectory is equal to the product of the estimated Markovian action probabilities, i.e., that $\frac{c(h)}{m} = \prod_{t=0}^{l-1}\hat{\pi}^{(0)}(a_t|s_t)$. With a finite number of samples, the data may have higher likelihood under a non-Markovian behavior policy—possibly even a policy that conditions on all past states and actions. Thus, to fully correct for sampling error, we must importance sample with an estimated non-Markovian behavior policy. However, $w_{\hat{\pi}^{(n)}}$ with $n < l-1$ still provides a better sampling error correction than w_{π_b} since any $\hat{\pi}^{(n)}$ will reflect the realized statistics of D while π_b only reflects the expected statistics. This statement is supported by our empirical results comparing RIS(0) to OIS and a theoretical result we present in the following section that states that, for all n, RIS(n) has lower asymptotic variance than the Monte Carlo estimator.

Before concluding this section, we discuss two limitations of the presented example—these limitations are *not* present in our theoretical or empirical results. First, the example lacks stochasticity in the rewards and transitions. In stochastic environments, sampling error arises from sampling states, actions, and rewards while in deterministic environments,

[2] This statement follows from the importance sampling identity: $\mathbf{E}[\frac{\Pr(H|\pi)}{\Pr(H|\pi')}\chi(H)|H \sim \pi] = \mathbf{E}[\chi(H)|H \sim \pi]$ and the fact that we have assumed a deterministic environment.

sampling error only arises from sampling actions. Like SEC, RIS is only able to correct for stochasticity in the action selection since d_0 and P are unknown. Second, we assumed that D contains at least one of each trajectory possible under π_b. If a trajectory is absent from D then RIS($l-1$) has non-zero bias. Theoretical analysis of this bias for both RIS($l-1$) and other RIS variants is an open question for future analysis.

5.2 Theoretical analysis

In this section we present theoretical properties of RIS estimators. Like SEC, we prove consistency and asymptotically lower variance than the Monte Carlo estimator. To the best of our knowledge, the only prior theoretical work on importance sampling with an estimated behavior policy for estimating trajectory expectations is the work of Farajtabar et al. (2018). This prior work makes the assumption that $\hat{\pi}$ is estimated with different data than the data used for the estimate and thus the analysis is inapplicable to RIS estimators.

5.2.1 Consistency

Following a similar proof to that of Proposition 1, we show that all RIS estimators are consistent estimators of $\bar{\chi}$. Like Proposition 1, we require the assumption of consistent estimation of the behavior policy.

Proposition 3 *Under Assumption 3, $\forall n$, RIS(n) is a consistent estimator of $\bar{\chi}$:* $\text{RIS}(n)(\pi, D) \xrightarrow{a.s.} \bar{\chi}$.

Proof See "Appendix 1" for a full proof.

5.2.2 Asymptotic variance

We also show that all RIS estimators have lower asymptotic variance compared to the OIS estimator or Monte Carlo estimator. The proof also requires Assumptions 4 and 5 to hold for the set of policies, Π_n, and behavior policy, π_b.

Corollary 2 *Under Assumptions 4 and 5, $\forall n$,*

$$\text{Var}_{\text{A}}(\text{RIS}(n)(\pi, D)) \leq \text{Var}_{\text{A}}(\text{OIS}(\pi, D, \pi_b))$$

where Var_{A} *denotes the asymptotic variance.*

Proof See "Appendix 3" for a full proof.

6 Empirical study: trajectory expectations

In the previous section, we introduced the RIS estimator as a general estimator for trajectory expectations in reinforcement learning. In order to empirically evaluate RIS, we apply the general estimator to the problem of batch policy evaluation. We show that using RIS and specifically the RIS(0) method leads to lower mean squared error policy evaluation than OIS in both the on- and off-policy case. We also show that RIS weights

can be used in conjunction with other variants of importance sampling to obtain even lower mean squared error policy evaluation.

Recall from Sect. 2.2.2 that in the batch policy evaluation problem, we seek to estimate $v(\pi_e)$ for some evaluation policy, π_e. We will assume we are given a batch of trajectories, D, that was collected by running some behavior policy, π_b. Our objective is to use a policy evaluation method, PE, that estimates $v(\pi_e)$ with low mean squared error:

$$\text{MSE}\left[\text{PE}\right] := \mathbf{E}\left[(\text{PE}(D) - v(\pi_e))^2 \;\middle|\; D \sim \pi_b\right].$$

Our primary baseline is the OIS estimator, though, we also consider extensions of OIS such as weighted importance sampling (Precup et al. 2000) and doubly robust estimators (Jiang and Li 2016; Thomas and Brunskill 2016a). Our experiments are designed to answer the following questions:

1. What is the empirical effect of replacing OIS weights, $\frac{\pi_e(a|s)}{\pi_b(a|s)}$, with RIS weights, $\frac{\pi_e(a|s)}{\hat{\pi}(a|s)}$, in policy evaluation for sequential decision making tasks?
2. How important is using D to both estimate the behavior policy and compute the importance sampling estimate?
3. How does the choice of n affect the MSE of RIS(n)?

With non-linear function approximation, our results suggest that the common supervised learning approach of model selection using hold-out validation loss may be suboptimal for the RIS estimator. Thus, we also investigate the question:

4. Does minimizing hold-out validation loss set yield the minimal MSE regression importance sampling estimator when estimating $\hat{\pi}$ with gradient descent and neural network function approximation?

6.1 Empirical set-up: trajectory expectations

We run policy evaluation experiments in several domains. We provide a short description of each domain here and the motivation for evaluating RIS methods in these domains.

6.1.1 Grid World

This domain is the same 4×4 Grid World used in Sect. 4 and has been used in prior off-policy policy evaluation work (Thomas 2015; Thomas and Brunskill 2016a). This domain allows us to study RIS separately from questions of function approximation as the small number of states and actions permits RIS to use count-based estimation of π_b. Our first set of experiments uses a behavior policy, π_b, that can reach the high reward terminal state and an evaluation policy, π_e, that is the same policy with lower entropy action selection. The second set of experiments uses the same behavior policy as both behavior and evaluation policy.

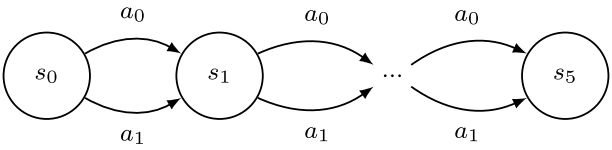


Fig. 10 The Single Path MDP. This environment has 5 states, 2 actions, and $l = 5$. The agent begins in state 0 and both actions either take the agent from state n to state $n + 1$ or cause the agent to remain in state n. *Not shown:* If the agent takes action a_1 it remains in its current state with probability 0.5

6.1.2 Single Path

See Fig. 10 for a description. This domain is small enough to make implementations of RIS($l - 1$) and the REG method from Li et al. (2015) tractable. We include the REG baseline since it can be shown to be equivalent to any RIS estimator in the contextless bandit setting; see "Appendix 5" for more discussion. All RIS methods use count-based estimation of π_b. In each state, π_b selects action, a_0, with probability $p = 0.6$ and π_e selects action, a_0, with probability $1 - p = 0.4$. Action a_0 causes a deterministic transition to the next state. Action a_1 causes a transition to the next state with probability 0.5, otherwise, the agent remains in its current state. The agent receives a reward of 1 for action a_0 and 0 otherwise. The REG baseline is given access to the environment's state transition function, P, which it needs to compute its estimate.

6.1.3 Linear dynamical system

This domain is the same LDS domain used in Sect. 4. We make one change which is that policies are linear in a second order polynomial transform of the state features instead of being linear in the state features. The intention of this change is to make the true behavior policy be a non-linear function of state features but still allow us to estimate $\hat{\pi}$ with ordinary least squares. We obtain a basic policy by optimizing the parameters of a policy for 10 iterations of the Cross-Entropy optimization method (Rubinstein and Kroese 2013). The basic policy maps the state to the mean of a Gaussian distribution over actions. The evaluation policy and true behavior policy both use the same basic policy to provide the mean but the evaluation policy uses a standard deviation of 0.5 and π_b uses a standard deviation of 0.6.

6.1.4 Simulated robotics

We also use two continuous control tasks from the OpenAI gym: Hopper and HalfCheetah.[3] In each task, we use neural network policies with 2 layers of 64 tanh hidden units each for π_e and π_b. Each policy maps the state to the mean of a Gaussian distribution with state-independent standard deviation. We obtain π_e and π_b by running the OpenAI Baselines (Dhariwal et al. 2017) implementation of proximal policy optimization (PPO) (Schulman et al. 2017) and then selecting two policies along the learning curve. For both environments, we use the policy after 30 updates for π_e and after 20 updates for π_b. These policies

[3] For these tasks we use the Roboschool versions: https://github.com/openai/roboschool.

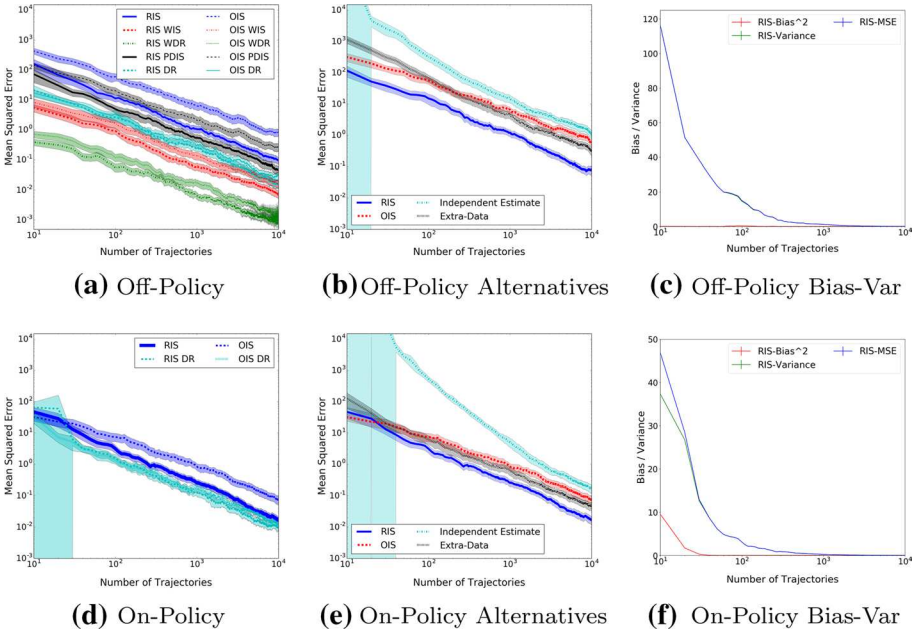


Fig. 11 Grid World policy evaluation results. In all subfigures, the horizontal axis is the number of trajectories collected and the vertical axis is mean squared error. Axes are log-scaled. The shaded region represents a 95% confidence interval. **a** Grid World Off-policy Policy Evaluation: The main point of comparison is the RIS variant of each method to the OIS variant of each method. **b** Grid World $\hat{\pi}$ Estimation Alternatives: This plot compares RIS and OIS to two methods that replace the true behavior policy with estimates from data sources other than D. **c** Empirical Bias^2 and Variance decomposition of MSE for RIS. **d–f** Identical experiments to **a–c** respectively except with the behavior policy from the first experiments as the evaluation policy (on-policy setting)

use tanh activations on their hidden units since these are the default in the OpenAI Baselines PPO implementation. RIS represents the behavior policy as a Gaussian distribution over possible actions with the mean given by a neural network function of the state and a state-independent standard deviation. RIS estimates the behavior policy with gradient descent on the negative log-likelihood of the actions with respect to the policy parameters. In all our experiments we use the Adam optimizer (Kingma and Ba 2015) with a learning rate of 1×10^{-3}. The neural network behavior policies learned by RIS have either 0, 1, 2, or 3 hidden layers with 64 hidden units with relu activations.

In all domains we run repeated trials of each experiment. Except for the simulated robotics domains, a trial consists of evaluating the squared error of different estimators over an increasing data set. The average squared error over multiple trials is an unbiased estimate of the mean squared error of each method. In the simulated robotics domain, a trial consists of collecting a single batch of 400 trajectories and evaluating the squared error of different estimators on this batch.

6.2 Empirical results: trajectory expectations

We now present our empirical results. Except where specified otherwise, RIS refers to RIS(0).

6.2.1 Grid World Policy evaluation

Our first experiment compares several importance sampling variants implemented with both RIS weights and OIS weights in the Grid World domain. Specifically, we use the basic IS estimator, the *weighted* IS estimator (Precup et al. 2000), *per-decision* IS, the *doubly robust* (Jiang and Li 2016), and the *weighted doubly robust* estimator (Thomas and Brunskill 2016a). Figure 11a shows the MSE of the evaluated methods averaged over 100 trials. The results show that, for this domain, using RIS weights lowers MSE for all tested IS variants relative to OIS weights.

We also evaluate alternative data sources for estimating $\hat{\pi}$ in order to establish the importance of using D to both estimate $\hat{\pi}$ and compute the estimate. Specifically, we consider:

1. *Independent estimate* In addition to D, this method has access to an additional set, $D_{\texttt{train}}$. The behavior policy is estimated with $D_{\texttt{train}}$ and the policy value estimate is computed with D. Since state-action pairs in D may be absent from $D_{\texttt{train}}$ we use Laplace smoothing (i.e., we add 1 to the count for each (s, a) pair (Manning et al. 2008)) to ensure that the importance weights never have a zero in the denominator.
2. *Extra-data estimate* This baseline is the same as **Independent Estimate** except it uses both $D_{\texttt{train}}$ and D to estimate π_b. Only D is used to compute the policy value estimate.

Figure 11b shows that these alternative data sources for estimating π_b decrease accuracy compared to RIS and OIS. **Independent Estimate** has high MSE when the sample size is small but its MSE approaches that of OIS as the sample size grows. We understand this result as showing that this baseline cannot correct for sampling error in the off-policy data since the behavior policy estimate is unrelated to the data used in computing the value estimate. **Extra-data Estimate** initially has high MSE but its MSE decreases faster than that of OIS. Since this baseline estimates π_b with data that includes D, it can partially correct for sampling error—though the extra data harms its ability to do so. Only estimating $\hat{\pi}$ with D and D alone lowers MSE over OIS for all sample sizes.

We also repeat these experiments for the on-policy setting and present results in Fig. 11c, d. We observe similar trends as in the off-policy experiments suggesting that RIS can lower variance in Monte Carlo sampling methods even when OIS weights are otherwise unnecessary.

In both the on- and off-policy setting, we measure the empirical decomposition of the MSE for RIS into its bias and variance components. In both settings we see that variance is the primary contributor to the MSE. In the on-policy setting, we find that RIS initially has a higher bias but this bias decreases to a negligible amount with a small number of trajectories.

6.2.2 RIS(n)

In the Grid World domain it is difficult to observe the performance of RIS(n) for various n because of the long horizon: smaller n perform similarly and larger n scale poorly with l. To see the effects of different n more clearly, we use the Single Path domain. Figure 12 gives the mean squared error for OIS, RIS, and the REG estimator of Li et al.

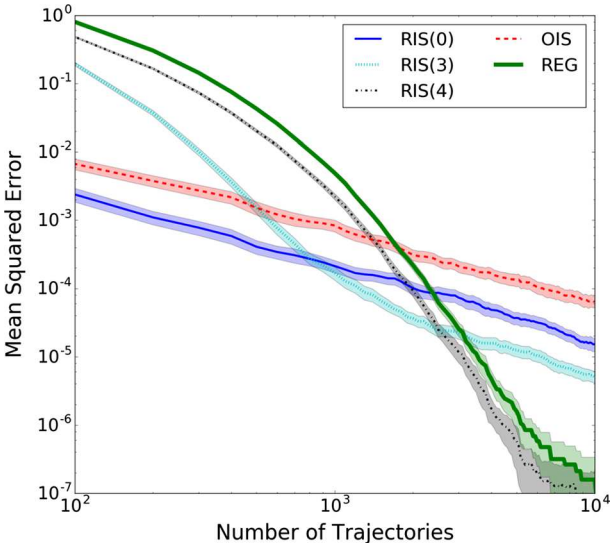


Fig. 12 Off-policy policy evaluation in the Single Path MDP for various n. The horizontal axis is the number of trajectories in D and the vertical axis is MSE. Both axes are log-scaled. The curves for REG and RIS(4) have been cut-off to more clearly show all methods. These methods converge to an MSE value of approximately 1×10^{-31}

(2015) that has full access to the environment's transition probabilities. For RIS, we use $n = 0, 3, 4$ and each method is run for 200 trials.

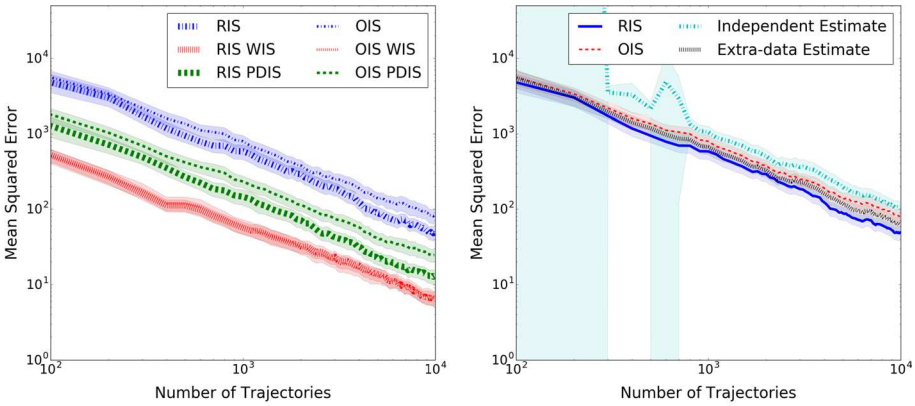
Figure 12 shows that higher values of n and REG tend to give inaccurate estimates when the sample size is small. However, as data increases, these methods give increasingly accurate value estimates. In particular, REG and RIS(4) produce estimates with MSE more than 20 orders of magnitude below that of RIS(3) (Fig. 12 is cut off at the bottom for clarity of the rest of the results). REG eventually passes the performance of RIS(4) since its knowledge of the transition probabilities allows it to eliminate sampling error in both the actions and the environment. In the low-to-medium data regime, only RIS(0) outperforms OIS. However, as data increases, the MSE of all RIS methods and REG decreases faster than that of OIS. We provide an additional, informal analysis of the observed similarities between RIS and REG in "Appendix 5".

6.2.3 RIS with linear function approximation

Our next set of experiments consider continuous state and action spaces in the Linear Dynamical System domain. RIS represents $\hat{\pi}$ as a Gaussian policy with mean given as a linear function of the state features. Similar to in Grid World, we compare three variants of IS, each implemented with RIS and OIS weights: the ordinary IS estimator, weighted IS (WIS), and per-decison IS (PDIS). Each method is averaged over 200 trials and results are shown in Fig. 13a.

We see that RIS weights lower the MSE of both IS and PDIS, while both WIS variants have similar MSE. This result suggests that the MSE reduction from using RIS weights depends, at least partially, on the variant of IS being used.

Similar to Grid World, we also consider estimating $\hat{\pi}$ with either an independent dataset or with extra data and see a similar ordering of methods. **Independent Estimate** gives high variance estimates for small sample sizes but then approaches OIS as the sample size grows. **Extra-Data Estimate** corrects for some sampling error and has lower MSE than OIS. RIS lowers MSE compared to all baselines.



(a) LDS **(b)** LDS Alt. Weights

Fig. 13 Linear dynamical system results. **a** Shows the mean squared error (MSE) for three IS variants with and without RIS weights. **b** Shows the MSE for different methods of estimating the behavior policy compared to RIS and OIS. Axes and scaling are the same as in Fig. 11a

6.2.4 RIS with neural network function approximation

Our remaining experiments use the Hopper and HalfCheetah domains with neural network function approximation. A practical concern for RIS estimators (and also SEC) is how to avoid over-fitting when using powerful function approximation to estimate the empirical policy. RIS uses all of the available data to both estimate $\hat{\pi}$ and compute the off-policy estimate of $\mathbf{E}[\chi(H)|H \sim \pi_e]$. Unfortunately, the RIS estimate may suffer from high variance if the function approximator is too expressive and $\hat{\pi}$ is over-fit to our data. Additionally, if the functional form of the true behavior policy, π_b, is unknown, it may be unclear what is the right function approximation representation for $\hat{\pi}$. A practical solution is to use a validation set—distinct from D—to select an appropriate policy class and appropriate regularization criteria for RIS. This solution is a small departure from the previous definition of RIS as selecting $\hat{\pi}$ to maximize the log likelihood on D and only D. Rather, we select $\hat{\pi}$ to maximize the log likelihood on D while avoiding over-fitting. This approach represents a trade-off between robust empirical performance and a potentially stronger sample correction by further maximizing log likelihood on the data used for computing the RIS estimate.

Figure 14 compares the MSE of RIS for different neural network architectures. Our main point of comparison is RIS using the architecture that achieves the lowest validation error during training (the darker bars in Fig. 14). Under this comparison, the MSE of RIS with a two-hidden-layer network is lower than that of OIS in both Hopper and HalfCheetah, though, in HalfCheetah, the difference is statistically insignificant. We also observe that the policy class with the best validation error does *not* always give the lowest MSE (e.g., in Hopper, the two hidden layer network gives the lowest validation loss but the network with a single layer of hidden units has $\approx 25\%$ less MSE than the two hidden layer network). This last observation motivates our final experiment.

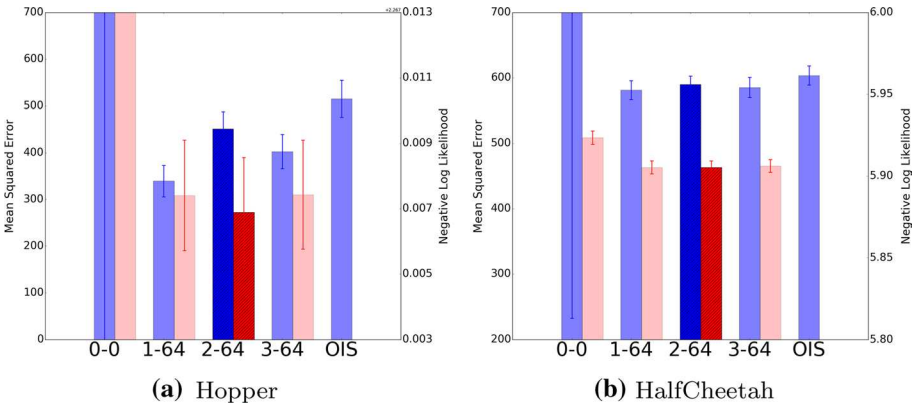


Fig. 14 **a**, **b** Compare different neural network architectures (specified as #-layers-#-units) for regression importance sampling on the Hopper and HalfCheetah domain. The darker, blue bars give the MSE for each architecture and OIS. Lighter, red bars give the negative log likelihood of a hold-out data set. Our main point of comparison is the MSE of the architecture with the lowest hold-out negative log likelihood (given by the darker pair of bars) compared to the MSE of OIS

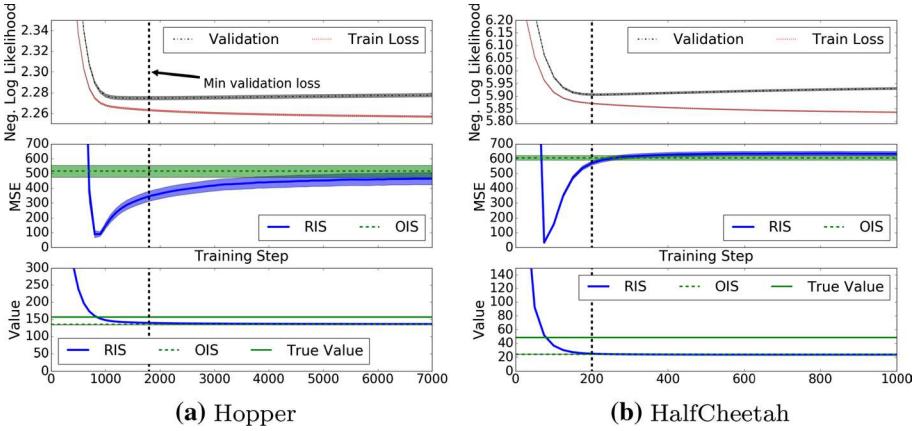


Fig. 15 Mean squared error and estimate of the importance sampling estimator during training of π_D. The horizontal axis is the number of gradient descent steps. The top plot shows the training and validation loss curves. The vertical axis of the top plot is the average negative log-likelihood. The y-axis of the middle plot is mean squared error (MSE). The y-axis of the bottom plot is the value of the estimate. MSE is minimized close to, but slightly before, the point where the validation and training loss curves indicate that overfitting is beginning. This point corresponds to where the RIS estimate transitions from over-estimating to under-estimating the policy value

6.2.5 RIS model selection

Our final experiment aims to better understand how hold-out validation error relates to the MSE of the RIS estimator when using gradient descent to estimate neural network approximations of $\hat{\pi}$. This experiment duplicates our previous experiment, except every 25 steps of gradient descent we stop optimizing $\hat{\pi}$ and compute the RIS estimate with the current $\hat{\pi}$ and its MSE. We also compute the training and hold-out validation negative log-likelihood.

Plotting these values gives a picture of how the MSE of RIS changes as our estimate of $\hat{\pi}$ changes. Figure 15 shows these plots for the Hopper and HalfCheetah domains.

We see that the policy with minimal MSE and the policy that minimizes validation loss are misaligned. If training is stopped when the validation loss is minimized, the MSE of RIS is lower than that of OIS (the intersection of the RIS curve and the vertical dashed line in Fig. 15. However, the $\hat{\pi}$ that minimizes the validation loss curve is *not* identical to the $\hat{\pi}$ that minimizes MSE.

To understand this result, we also plot the mean RIS estimate throughout behavior policy learning (bottom of Fig. 15). We can see that at the beginning of training, RIS tends to *over-estimate* $v(\pi_e)$ because the probabilities given by $\hat{\pi}$ to the observed data will be small (and thus the RIS weights are large). As the likelihood of D under $\hat{\pi}$ increases (negative log likelihood decreases), the RIS weights become smaller and the estimates tend to *under-estimate* $v(\pi_e)$. The implication of these observations, for RIS, is that during behavior policy estimation the RIS estimate will likely have zero MSE at some point. Thus, there may be an early stopping criterion—besides minimal validation loss—that would lead to lower mse with RIS, however, to date we have not found one. Note that OIS also tends to under-estimate policy value in MDPs as has been previously analyzed by Doroudi et al. (2017).

7 Related work

In this section we survey literature related to importance sampling with an estimated behavior policy, alternatives to Monte Carlo sampling in reinforcement learning, and variance reduction for Monte Carlo sampling.

7.1 Importance sampling with an estimated behavior policy

A number of research works have shown that estimating the denominator of importance weights (instead of using the true probabilities) lowers the variance of importance sampling. To the best of our knowledge, all such prior work has been done in the multi-armed bandit, contextual bandit, or causal inference communities. One can directly extend these methods to state-action expectations by estimating $d_\pi(s)\pi(s)$ or to trajectory expectations by estimating $\Pr(h|\pi)$. Unfortunately, such methods are often impractical as they require knowing $d_\pi(s)$ or $\Pr(h|\pi)$ for the numerator of the importance weights. Concurrent to this work, Pavse et al. (2020) built upon our prior work (Hanna et al. 2019; Hanna and Stone 2019) and showed that a SEC-like method could lower error in batch value function approximation.

Our work takes inspiration from Li et al. (2015) who prove, for contextless bandits, that importance sampling with an estimated behavior policy has lower minimax mean squared error than using the true behavior policy. They corroborate these theoretical findings with experiments showing that the mean squared error of the so-called REG estimator decreases faster than that of importance sampling with the true behavior policy. The main distinction between this work and the work of Li et al. (2015) is that we consider MDPs where actions affect both reward and the next state. Our theoretical results only address the asymptotic sample size while Li et al. (2015) provide variance and bias results for finite samples of any size.

For contextual bandits, Narita et al. (2019) prove that importance sampling with an estimated behavior policy minimizes asymptotic variance among all asymptotically

normal estimators (including ordinary importance sampling). They also provide a large-scale study of policy evaluation with the empirical behavior policy on an ad-placement task. Xie et al. (2018) provide similar results and prove a reduction in finite-sample mean squared error when using an estimated behavior policy. Again, our work differs from these two works in that we are concerned with full MDPs.

It has long been known in the causal inference literature that the empirical behavior policy produces lower variance estimates than using the true behavior policy for importance sampling. In this literature, the behavior policy action probabilities are known as *propensities* and importance sampling is known as *inverse propensity scoring* (Austin 2011). Rosenbaum (1987) first showed that using parametric propensity estimates lowered the variance of importance sampling. In later work, Hirano et al. (2003) studied this approach using non-parametric propensity score estimates. The causal inference problems studied can be viewed as a class of contextual bandit problems. Under that view, our work differs from these earlier studies in that we are concerned with MDPs.

Importance sampling is commonly defined as a way to use samples from a *proposal* distribution to estimate an expectation under a *target* distribution. Henmi et al. (2007) proved that importance sampling with a maximum likelihood parametric estimate of the proposal distribution has lower asymptotic variance than using the true proposal distribution. This result forms the basis of our own proofs that show SEC and all RIS methods have lower asymptotic variance than Monte Carlo estimates. Delyon and Portier (2016) proved asymptotic lower variance for using a non-parametric estimate of the proposal distribution.

Other works have explored directly estimating the importance weights instead of first estimating the proposal distribution (i.e., behavior policy) to compute the importance weights (Oates et al. 2017; Liu and Lee 2017). These "blackbox" importance sampling approaches show superior convergence rates compared to ordinary importance sampling. In recent years a number of methods have been proposed that attempt to weight (s, a) pairs with blackbox weights when estimating state-action expectations for policy evaluation (Liu et al. 2018; Mousavi et al. 2020; Yang et al. 2020). The stated focus of most of these works tends to be on reducing variance due to long horizons; an interesting question is whether some of the success of these methods is due to correcting sampling error.

In contextual bandit problems, Dudík et al. (2011) present theoretical results showing that an estimated behavior policy may increase the variance of importance sampling while also introducing bias. Farajtabar et al. (2018) prove similar results for full MDPs. However, in these works the behavior policy is estimated with a *separate* set of data than the set used for computing the off-policy value estimate. Because the behavior policy is estimated with a separate set of data it has no power to correct sampling error in the data used for the off-policy value estimate. In fact, these theoretical findings are in line with our experiments showing that it is important to use the same set of data both to estimate the behavior policy and to compute the regression importance sampling estimate (see Figs. 11e, f, 13b in Sect. 6).

Raghu et al. (2018) report that larger differences between the true behavior policy and estimated behavior policy lead to more error in the off-policy value estimate. However, they measure off-policy policy evaluation error with respect to the true behavior policy weighted importance sampling estimate and so it is unsurprising that as the policies become more different the error increases.

7.2 Analytic expectations

In this work we use importance sampling with an estimated behavior policy to correct sampling error in reinforcement learning. Here, we discuss alternative approaches in the reinforcement learning literature that avoid sampling error altogether.

The SARSA algorithm (Rummery and Niranjan 1994) uses (S, A, R, S', A') tuples to learn an estimate of the *action-value* function, q^π, for a policy π. The algorithm requires two sampled actions for each update and the second of these is used to form a Monte Carlo estimate of the expected value of q^π in state S'. The *expected* SARSA update (Van Seijen et al. 2009) replaces the Monte Carlo estimate with an analytic evaluation of the expected value of q^π in S'. By replacing the Monte Carlo estimate, sampling error is eliminated and *expected* SARSA may converge much faster than SARSA. Expected SARSA requires either a small discrete action-set or for π and q^π to have forms that allow analytic integration. In this work, we place no limitations on the action-set or policy and do *not* explicitly learn an action-value function.

Expected SARSA can be extended to a multi-step algorithm with the tree-backup algorithm (Precup et al. 2000; Sutton and Barto 1998). More recent work has shown that the amount of sampling as opposed to exact expectations can be done on a per-state basis using the $Q(\sigma)$ algorithm (Asis et al. 2018). Other tree-backup-like algorithms have been proposed and hold the promise to eliminate sampling error in off-policy data (Yang et al. 2018; Shi et al. 2019). Like expected SARSA, these algorithms require the ability to compute the sum of $\pi(a|s)q^\pi(s,a)$ over all $a \in \mathcal{A}$.

In policy gradient reinforcement learning, Sutton et al. (2000) introduced the *all-actions policy gradient* algorithm that avoids sampling in the action-space by first learning the function q^{π_θ} and then analytically computing the expectation of $q^{\pi_\theta}(s,a)\frac{\partial}{\partial \boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s)$. This approach has been further developed as the *expected policy gradient* algorithm (Ciosek and Whiteson 2018; Fellows et al. 2018), the *mean actor-critic* algorithm (Asadi et al. 2017), and the *MC-256* algorithm (Petit et al. 2019). With a good approximation of q^π, these algorithms learn faster than a Monte Carlo policy gradient estimator. However, requiring a good approximation of q^π undercuts one of the primary reasons for using policy gradient RL: it may be easier to represent a good policy than to represent the correct action-value function (Sutton and Barto 1998). The sampling error corrected policy gradient estimator provides an alternative method for reducing sampling error when q^π is difficult to learn. We also note that estimating π (as the sampling error corrected policy gradient estimator does) may be easier than estimating q^π since the right function approximator class for π is known while, in general, it is unknown for q^π.

7.3 Variance reduction in reinforcement learning

Aside from reducing sampling error, other approaches exists for lowering the variance of Monte Carlo expectation evaluations in reinforcement. Control variates use the known expected value of a second random variable to lower the variance of estimating the expected value of ϕ or χ. The most commonly considered type of control variate in the RL literature is the *additive control variate* which includes constant baselines (Thomas and Brunskill 2017), state dependent baselines (Greensmith et al. 2004; Schulman et al. 2016) and state-action dependent baselines (Jiang and Li 2016; Thomas and Brunskill 2016a) A second type of control variate is the *multiplicative* control variate of which the *weighted*

importance sampling estimator (Precup et al. 2000) may be the best known in the RL literature. As we have shown in our empirical study, control variate techniques are complementary to the sampling error correction methods we introduce.

Adaptive importance sampling methods change the data distribution to lower the variance of the Monte Carlo estimator. The data distribution of a Monte Carlo estimator can be adapted by either changing the behavior policy or the MDP transition probabilities. Hanna et al. (2017) show that the OIS estimator can have lower variance than on-policy Monte Carlo sampling and introduce a method that adapts the behavior policy to obtain low variance estimates for the problem of off-policy batch policy evaluation. Ciosek and Whiteson (2017) and Frank et al. (2008) consider adaptive importance sampling through changing P. This approach is possible when learning is done in a simulator and we can both know and control P. Regardless of how the data distribution is adapted, adaptive importance sampling methods still have variance due to sampling error.

Finally, bootstrapping from a learned value function is a widely used variance reduction strategy in RL (Sutton 1984; Mnih et al. 2016; Greensmith et al. 2004). In some cases, this technique would provide complementary variance reduction to that of SEC or RIS estimators. For example, in Sect. 4, we use a learned value function as a baseline (Greensmith et al. 2004; Schulman et al. 2016) for both the SEC policy gradient estimator and the Monte Carlo policy gradient estimator. In other cases, such as online value function learning, further work may be needed to apply SEC and RIS.

8 Discussion of limitations

In this section we discuss the results we have presented and limitations of the SEC and RIS estimator.

Our theoretical and empirical studies have focused on the statistical properties of the SEC and RIS estimators. The gain in statistical efficiency comes at a cost of increased computational complexity. Both SEC and all RIS estimators have an additional step of estimating the empirical behavior policy compared to the Monte Carlo estimator. Furthermore, in the on-policy setting, the Monte Carlo estimator avoids computing importance ratios while SEC and RIS estimators must always compute the ratios. The trade-off between computational and statistical efficiency is a trade-off that must be made by practitioners.

Our theoretical analysis compared the asymptotic properties of our new estimators to that of the Monte Carlo estimator. This analysis proves the statistical benefit of using our new estimators when the sample size is very large. However, our empirical results show a statistical benefit to using the new estimators even for smaller sample sizes. Currently, we lack a theoretical explanation for small sample size variance reduction. We also know that SEC and RIS estimators are introducing bias but we lack theoretical analysis as to how much bias is introduced and how fast this bias goes to zero.

The SEC and RIS estimators are related to the use of importance sampling for off-policy reinforcement learning where the behavior policy is unknown and thus must be estimated before it can be used to form the importance weights. In practice, behavior policy estimation can be challenging when the distribution class of the true behavior policy is unknown (Raghu et al. 2018). However, in the settings we studied, we have complete access to the behavior policy and can specify the policy set Π to include π (thus ensuring consistency of the SEC and RIS estimators). We can even simplify the policy set Π by estimating a policy that conditions on intermediate representations of the behavior policy. For example

if the behavior policy, π_b, is a convolutional neural network mapping states to a softmax distribution over actions, we can use all but the last layer of π_b as a feature extractor and then model Π as all linear functions mapping these features to a softmax distribution over actions. Such a technique can significantly simplify estimating $\hat{\pi}$ while maintaining consistency guarantees when the behavior policy is a complex function. Our CartPole experiment in Sect. 4 shows evidence of the benefit of this approach.

9 Future work

In this section, we outline directions for future work to further develop the SEC and RIS estimators for correcting sampling error in reinforcement learning. As an overarching direction, we note that this work assumed an episodic and fully observable environment. Future work should consider how to best correct sampling error in continuing or partially observable environments.

9.1 Behavior policy search for regression importance sampling

The methods introduced in this article are methods that lower variance *post* data collection. That is, data is collected in the same way that a Monte Carlo estimator would collect data, and only then do our new methods re-weight data to lower variance. One direction for future work would be to answer the question, "how should we collect data for the most accurate SEC or RIS estimate?"

Hanna et al. (2017) introduce the idea of adapting the behavior policy to lower the variance of Monte Carlo policy evaluation. However, after collecting data, their policy value estimate remains a Monte Carlo estimate. A straightforward additional study would be to use their behavior policy gradient algorithm to learn how collect data but then use regression importance sampling to lower sampling error in the observed data.

Though straightforward, this proposed approach may be sub-optimal and we illustrate this fact by considering the bandit setting. Consider a k-armed bandit with deterministic rewards on each arm. After all k arms have been observed, the RIS estimate will have both zero bias and zero variance.[4] Thus the optimal behavior policy for RIS should increase the probability of unobserved actions; it is a non-stationary policy that depends on all of the past actions. In contrast, an optimal behavior policy for the Monte Carlo estimator would take actions in proportion to $\pi(a)r(a)$ (Hanna et al. 2017). Thus behavior policy search, as introduced in prior work, may yield a behavior policy that is sub-optimal for the RIS estimator.

9.2 Finite-sample analysis

In Sects. 3.2 and 5.2 we proved SEC and RIS have asymptotically variance at most that of the Monte Carlo estimator. Further theoretical analysis should examine the finite-sample bias and variance of SEC and RIS compared to the Monte Carlo estimator. A starting point

[4] This statement follows from having deterministic rewards and the observation of Li et al. (2015) that importance sampling with an estimated behavior policy is equivalent to an analytic expectation over the estimated reward function.

for this work could be the results of Li et al. (2015) who provide bounds on these finite-sample quantities in the bandit setting. Extending these results to MDPs would give us a deeper understanding of when RIS and SEC are lower error estimators than Monte Carlo. The empirical results in Sect. 6 provide strong evidence that RIS is always preferable to ois. However, theoretical analysis would strengthen this claim.

The theoretical analysis in Sect. 5.2 did *not* distinguish different RIS methods according to how much history they conditioned on (the estimator parameter n). Theoretical analysis of the finite-sample bias-variance trade-off and asymptotic variance for different RIS methods would deepen our understanding of how to choose n. Empirical results on the Singlepath domain (Fig. 12) suggest that small n have lower small-sample MSE while large n have asymptotically lower MSE. Verifying this finding formally is an interesting direction for future work.

9.3 Value function learning

Finally, we have only considered estimating scalar or vector-valued expectations that arise in the RL literature. Another important problem that arises in the RL literature is how to efficiently learn the value function that gives the expected return of a policy from any state. Many value function learning algorithms rely on leveraging intermediate value estimates to avoid variance due to sampling many consecutive actions (Sutton 1984). However, these methods still tend to require some amount of action sampling and thus have some amount of sampling error to be corrected. Pavse et al. (2020) have shown that correcting sampling error with a method like SEC or the RIS estimators leads to lower value function error compared to standard temporal difference learning when learning from a fixed batch of data. Future work should consider whether a similar advantage can be shown in *online* value function learning where the learning agent processes a single transition tuple (s, a, r, s') at a time.

9.4 Regression importance sampling for high confidence off-policy evaluation

Empirical results in Sect. 6 showed that regression importance sampling leads to lower mean squared error off-policy evaluation. It remains to be seen if RIS also leads to tighter confidence intervals for high confidence off-policy evaluation. One way to tackle this problem would be to simply use RIS with a bootstrap confidence interval as done by Thomas et al. (2015) and Hanna et al. (2017). Given that RIS has been empirically shown to have lower variance than ordinary importance sampling, we could expect such a method to produce tighter confidence intervals.

A more challenging direction for future work would be to obtain true confidence intervals with an estimated behavior policy. While the data efficiency of bootstrapping is desirable, it only provides approximate confidence bounds. In order to determine exact confidence intervals for RIS, we would need to develop *concentration inequalities* for RIS in the same way that one can use Hoeffding's inequality to establish confidence intervals for ois. One possible direction is to explore use of the Dvoretzky-Kiefer-Wolfowitz inequality which bounds how far the empirical distribution of samples is from the true distribution (Dvoretzky et al. 1956). Regardless of the exact approach, exact confidence bounds for importance sampling with an estimated behavior policy would be of great value to providing provable guarantees of safety in real world settings where the true behavior policy is unknown.

10 Conclusion

This article introduces and describes a general method for reducing the variance of Monte Carlo estimation in reinforcement learning: estimate the empirical action probabilities, $\hat{\pi}(a|s)$, from observed data and then use importance sampling with the ratio $\frac{\pi(a|s)}{\hat{\pi}(a|s)}$. This general approach lowers variance by correcting sampling error—error due to stochasticity in the agent's action selection. Following this general approach, we first introduce the sampling error corrected (SEC) estimator and present theoretical analysis showing that the SEC estimator has asymptotic variance at most that of the Monte Carlo estimator. We use the SEC estimator to lower the variance of policy gradient estimates in two batch policy gradient algorithms and demonstrate this approach leads to more data efficient RL compared to a Monte Carlo approach.

We next introduce a family of regression importance sampling (RIS) estimators for settings where the desired expectation to estimate is written as a distribution over trajectories. Like the SEC estimator, RIS estimators first estimate the behavior policy before importance sampling. Unlike the SEC estimator, the family of RIS estimators contains methods that estimate non-Markovian behavior policies before importance sampling and corrects for sampling error due to action selection along the entire trajectory. We show that all RIS estimators have asymptotic variance at most that of the Monte Carlo estimator. We further apply RIS to the problem of off-policy policy evaluation and show that RIS estimators lead to lower mean squared error policy value estimates than Monte Carlo importance sampling variants.

Appendix 1: Consistency proof

In this appendix we show that, assuming we use a consistent estimator of the behavior policy, the SEC estimator and RIS estimators are consistent estimators of $\bar{\phi}$ and $\bar{\chi}$ respectively.

Assumption 3 (Consistent estimation of $\hat{\pi}$)

$$\underset{\pi \in \Pi}{\operatorname{argmax}} \sum_{j=1}^{k} \log \pi(A_j|S_j) \xrightarrow{a.s.} \pi$$

where $\xrightarrow{a.s.}$ denotes almost sure convergence.

Proposition 1 *Under Assumption 3, the SEC estimator is a consistent estimator of* $\bar{\phi}$:

$$\text{SEC}(D) \xrightarrow{a.s.} \bar{\phi}.$$

Proof We have assumed that as the amount of data increases, the behavior policy estimated by SEC will almost surely converge to the true behavior policy:

$$\hat{\pi} \xrightarrow{a.s.} \pi_b.$$

Almost sure convergence to the true behavior policy means that SEC almost surely converges to the Monte Carlo estimate. Consider the difference, $\text{SEC}(D) - \text{MC}(D)$. Since $\hat{\pi} \xrightarrow{a.s.} \pi_b$, we have that:

$$\text{SEC}(D) - \text{MC}(D) \xrightarrow{a.s.} 0.$$

Thus, with probability 1, SEC and Monte Carlo converge to the same value. Since the Monte Carlo estimator is a consistent estimator of $\bar{\phi}$, then with probability 1 we have that $\text{OIS}(\pi_e, D)$ converges to $\bar{\phi}$. Thus $\text{SEC}(\mathcal{D}) \xrightarrow{a.s.} \bar{\phi}$.

Similarly, for $\text{RIS}(n)$:

Proposition 3 *Under Assumption 3, $\forall n$, $\text{RIS}(n)$ is a consistent estimator of $\bar{\chi}$:* $\text{RIS}(n)(\pi, D) \xrightarrow{a.s.} \bar{\chi}$.

Proof The proof is identical to that for Proposition 3 with $\text{RIS}(n)$ taking the place of SEC, $\bar{\chi}$ taking the place of $\bar{\phi}$, and the off-policy ordinary importance sampling estimator taking the place of the Monte Carlo estimator.

Appendix 2: Consistent behavior policy estimation

The previous section proves the SEC and RIS estimators are consistent as long as they use consistent estimators of the true behavior policy. In this section we give more precise assumptions under which we can prove consistent behavior policy estimation.

The main intuition for the proofs is that SEC and RIS estimators are performing policy search on an estimate of the log-likelihood, $\widehat{\mathcal{L}}(\pi|D)$, as a surrogate objective for the true log-likelihood, $\mathcal{L}(\pi)$. Since π_b has generated our data, π_b is the optimal solution to this policy search. As long as, for all π, $\widehat{\mathcal{L}}(\pi|D)$ is a consistent estimator of $\mathcal{L}(\pi)$ then selecting $\hat{\pi} = \text{argmax}_{\pi \in \Pi} \widehat{\mathcal{L}}(\pi|D)$ will converge probabilistically to π_b. If the set of policies we search over, Π, is countable then this argument is almost enough to show a consistent behavior policy estimator. The difficulty (as we explain below) arises when Π is *not* countable.

Our proof takes inspiration from Thomas and Brunskill who show that their magical policy search algorithm converges to the optimal policy by maximizing a surrogate estimate of policy value 2016b. They show that performing policy search on a policy value estimate, $\hat{v}(\pi)$, will almost surely return the policy that maximizes $v(\pi)$ if $\hat{v}(\pi)$ is a consistent estimator of $v(\pi)$. The proof is almost identical; the notable difference is substituting the log-likelihood, $\mathcal{L}(\pi)$, and a consistent estimator of the log-likelihood, $\widehat{\mathcal{L}}(\pi|D)$, in place of $v(\pi)$ and $\hat{v}(\pi)$.

Appendix 2.1: Definitions and assumptions

Let $\mathcal{H}_n$ be the set of all possible state-action trajectory segments with n states and $n-1$ actions:

$$\mathcal{H}_n = \mathcal{S}^n \times \mathcal{A}^{n-1}.$$

We will denote elements of $\mathcal{H}_n$ as h_n and random variables that take values from $\mathcal{H}_n$ as H_n. Let $d_{\pi_b,\mathcal{H}_n} : \mathcal{H}_n \to [0,1]$ be the distribution over elements of $\mathcal{H}_n$ induced by running π_b. Previously, we defined the behavior policy, π_b, to be a function mapping state-action pairs to probabilities. We re-define $\pi_b : \mathcal{H}_n \times \mathcal{A} \to [0,1]$, i.e., a policy that conditions the distribution over actions on the preceding length n trajectory segment. These definitions are equivalent provided for any $h_{n,i} = (s_i, a_i, ...s_{i+n-1})$ and $h_{n,j} = (s_j, a_j, ...s_{j+n-1})$, if $s_{i+n-1} = s_{j+n-1}$ then $\forall a\ \pi_b(a|h_{n,i}) = \pi_b(a|h_{n,j})$.

Let $(\Omega, \mathcal{F}, \mu)$ be a probability space and $D_m : \Omega \to \mathcal{D}$ be a random variable. $D_m(\omega)$ is a sample of m trajectories with $\omega \in \Omega$. Let d_{π_b} be the distribution of length n trajectory segments under π_b. Define the expected log-likelihood:

$$\mathcal{L}(\pi) = \mathbf{E}\left[\log \pi(A|H_n) \;\middle|\; H_n \sim d_{\pi_b,\mathcal{H}_n}, A \sim \pi_b\right]$$

and its sample estimate from samples in $D_m(\omega)$:

$$\widehat{\mathcal{L}}(\pi|D_m(\omega)) = \frac{1}{ml}\sum_{j=1}^{m}\sum_{t=0}^{l-1}\log \pi(A_t^j|H_{t-n,t}^j).$$

Note that:

$$\pi_b = \underset{\pi\in\Pi}{\operatorname{argmax}}\ \mathcal{L}(\pi)$$

and

$$\pi_D{}^{(n)} = \underset{\pi\in\Pi}{\operatorname{argmax}}\ \widehat{\mathcal{L}}(\pi|D_m(\omega)).$$

Define the KL-divergence (D_{KL}) between π_b and π_D after segment h_n as:

$$\delta_{\mathrm{KL}}(h_n) = D_{\mathrm{KL}}(\pi_b(\cdot|h_n), \pi_D(\cdot|h_n)).$$

Assuming for all h_n and a the variance of $\log \pi(a|h_n)$ is bounded, $\widehat{\mathcal{L}}(\pi|D_m(\omega))$ is a consistent estimator of $\mathcal{L}(\pi)$. We make this assumption explicit:

Assumption 8 (*Consistent Estimation of Log likelihood*). For all $\pi \in \Pi$, $\widehat{\mathcal{L}}(\pi|D_m(\omega)) \xrightarrow{a.s.} \mathcal{L}(\pi)$.

This assumption will hold when the support of π_b is a subset of the support of π for all $\pi \in \Pi$, i.e., no $\pi \in \Pi$ places zero probability measure on an action that π_b might take. We can ensure this assumption is satisfied by only considering $\pi \in \Pi$ that place non-zero probability on any action that π_b has taken.

We also make an additional assumption about the piece-wise continuity of the log-likelihood, $\mathcal{L}$, and the estimate of the log-likelihood, $\widehat{\mathcal{L}}$. First we present two necessary definitions as given by Thomas and Brunskill (2016b):

Definition 3 (*Piecewise Lipschitz continuity*). We say that a function $f : M \to \mathbb{R}$ on a metric space (M, d) is piecewise Lipschitz continuous with respect to Lipschitz constant K and with respect to a countable partition, $\{M_1, M_2, ...\}$ if f is Lipschitz continuous with Lipschitz constant K on all metric spaces in $\{(M_i, d_i)\}_{i=1}^{\infty}$.

Definition 4 (*δ-covering*). If (M, d) is a metric space, a set $X \subset M$ is a δ-covering of (M, d) if and only if $\max_{y \in M} \min_{x \in X} d(x, y) \leq \delta$.

Assumption 9 (*Piecewise Lipschitz objectives*). Our policy class, Π, is equipped with a metric, d_Π, such that for all $D_m(\omega)$ there exist countable partition of Π, $\Pi^{\mathcal{L}} := \{\Pi_1^{\mathcal{L}}, \Pi_2^{\mathcal{L}}, ...\}$ and $\Pi^{\widehat{\mathcal{L}}} := \{\Pi_1^{\widehat{\mathcal{L}}}, \Pi_2^{\widehat{\mathcal{L}}}, ...\}$, where $\mathcal{L}$ and $\widehat{\mathcal{L}}(\cdot|D_m(\omega))$ are piecewise Lipschitz continuous with respect to $\Pi^{\mathcal{L}}$ and $\Pi^{\widehat{\mathcal{L}}}$ with Lipschitz constants K and $\widehat{K}$ respectively. Furthermore, for all $i \in \mathbb{N}_{>0}$ and all $\delta > 0$ there exist countable δ-covers of $\Pi_i^{\mathcal{L}}$ and $\Pi_i^{\widehat{\mathcal{L}}}$.

As pointed out by Thomas and Brunskill, this assumption holds for the most commonly considered policy classes but is also general enough to hold for other settings (see Thomas and Brunskill 2016b for further discussion of Assumption 9 and the related definitions).

Appendix 2.2: Consistent behavior policy estimation proof

We now show that SEC and RIS estimators use consistent behavior policy estimation by showing that the expected KL-divergence between the true behavior policy and estimted behavior policy almost surely goes to zero.

Lemma 1 *If Assumptions 8 and 9 hold then* $\mathbf{E}[\delta_{\mathrm{KL}}(H_n)|H_n \sim d_{\pi_b, \mathcal{H}_n}] \xrightarrow{a.s.} 0$.

Proof Define $\Delta(\pi, \omega) = |\widehat{\mathcal{L}}(\pi|D_m(\omega)) - \mathcal{L}(\pi)|$. From Assumption 8 and one definition of almost sure convergence, for all $\pi \in \Pi$ and for all $\epsilon > 0$:

$$\Pr\left(\liminf_{m \to \infty} \{\omega \in \Omega : \Delta(\pi, \omega) < \epsilon\}\right) = 1. \tag{21}$$

Thomas and Brunskill point out that because Π may not be countable, (21) may not hold at the same time for all $\pi \in \Pi$. More precisely, it does *not* immediately follow that for all $\epsilon > 0$:

$$\Pr\left(\liminf_{m \to \infty} \{\omega \in \Omega : \forall \pi \in \Pi, \Delta(\pi, \omega) < \epsilon\}\right) = 1. \tag{22}$$

Let $C(\delta)$ denote the union of all of the policies in the δ-covers of the countable partitions of Π assumed to exist by Assumption 2. Since the partitions are countable and the δ-covers for each region are assumed to be countable, we have that $C(\delta)$ is countable for all δ. Thus, for all $\pi \in C(\delta)$, (21) holds simulatenously. More precisely, for all $\delta > 0$ and for all $\epsilon > 0$:

$$\Pr\left(\liminf_{m \to \infty} \{\omega \in \Omega : \forall \pi \in C(\delta), \Delta(\pi, \omega) < \epsilon\}\right) = 1. \tag{23}$$

Consider a $\pi \notin C(\delta)$. By the definition of a δ-cover and Assumption 9, we have that $\exists \pi' \in \Pi_i^{\mathcal{L}}, d(\pi, \pi') \leq \delta$. Since Assumption 9 requires $\mathcal{L}$ to be Lipschitz continuous on $\Pi_i^{\mathcal{L}}$, we have that $|\mathcal{L}(\pi) - \mathcal{L}(\pi')| \leq K\delta$. Similarly $|\widehat{\mathcal{L}}(\pi|D_m(\omega)) - \widehat{\mathcal{L}}(\pi'|D_m(\omega))| \leq \widehat{K}\delta$. So, $|\widehat{\mathcal{L}}(\pi|D_m(\omega)) - \mathcal{L}(\pi)| \leq |\widehat{\mathcal{L}}(\pi|D_m(\omega)) - \mathcal{L}(\pi')| + K\delta \leq |\widehat{\mathcal{L}}(\pi'|D_m(\omega)) - \mathcal{L}(\pi')| + (\widehat{K} + K)\delta$ Then it follows that for all $\delta > 0$:

$$(\forall \pi \in C(\delta), \Delta(\pi, \omega) \leq \epsilon) \rightarrow \left(\forall \pi \in \Pi, \Delta(\pi, \omega) < \epsilon + (K + \widehat{K})\delta\right).$$

Substituting this into (23) we have that for all $\delta > 0$ and for all $\epsilon > 0$:

$$\Pr\left(\liminf_{m\to\infty}\{\omega \in \Omega : \forall \pi \in \Pi, \Delta(\pi, \omega) < \epsilon + (K + \widehat{K})\delta\}\right) = 1.$$

The next part of the proof massages (23) into a statement of the same form as (22). Consider the choice of $\delta := \epsilon/(K + \widehat{K})$. Define $\epsilon' = 2\epsilon$. Then for all $\epsilon' > 0$:

$$\Pr\left(\liminf_{m\to\infty}\{\omega \in \Omega : \forall \pi \in \Pi, \Delta(\pi, \omega) < \epsilon'\}\right) = 1. \tag{24}$$

Since $\forall \pi \in \Pi, \Delta(\pi, \omega) < \epsilon'$, we obtain:

$$\Delta(\pi_b, \omega) < \epsilon' \tag{25}$$

$$\Delta(\pi_D, \omega) < \epsilon' \tag{26}$$

and then applying the definition of Δ:

$$\mathcal{L}(\pi_D) \overset{(a)}{\leq} \mathcal{L}(\pi_b) \tag{27}$$

$$\overset{(b)}{<} \widehat{\mathcal{L}}(\pi_b | D_m(\omega)) + \epsilon' \tag{28}$$

$$\overset{(c)}{\leq} \widehat{\mathcal{L}}(\pi_D | D_m(\omega)) + \epsilon' \tag{29}$$

$$\overset{(d)}{\leq} \mathcal{L}(\pi_D) + 2\epsilon' \tag{30}$$

where (a) comes from the fact that π_b maximizes $\mathcal{L}$, (b) comes from (25), (c) comes from the fact that π_D maximizes $\widehat{\mathcal{L}}(\cdot|D_m(\omega))$, and (d) comes from (26). Considering (27) and (30), it follows that $|\mathcal{L}(\pi_D) - \mathcal{L}(\pi_b)| < 2\epsilon'$. Thus, (24) implies that:

$$\forall \epsilon' > 0, \Pr\left(\liminf_{m\to\infty}\{\omega \in \Omega : |\mathcal{L}(\pi_D) - \mathcal{L}(\pi_b)| < 2\epsilon'\}\right) = 1.$$

Using $\epsilon'' := 2\epsilon'$ we obtain:

$$\forall \epsilon'' > 0, \Pr\left(\liminf_{m\to\infty}\{\omega \in \Omega : |\mathcal{L}(\pi_D) - \mathcal{L}(\pi_b)| < \epsilon''\}\right) = 1$$

From the definition of the KL-Divergence,

$$\mathcal{L}(\pi_D) - \mathcal{L}(\pi_b) = \mathbf{E}[\delta_{\mathrm{KL}}(H_n) | H_n \sim d_{\pi_b, \mathcal{H}_n}]$$

and we obtain that:

$$\forall \epsilon > 0, \Pr\left(\liminf_{n\to\infty}\{\omega \in \Omega : |-\mathbf{E}[\delta_{\mathrm{KL}}(H_n) | H_n \sim d_{\pi_b, \mathcal{H}_n}]| < \epsilon\}\right) = 1$$

And finally, since the KL-Divergence is non-negative:

$$\forall \epsilon > 0, \Pr\left(\liminf_{m\to\infty}\{\omega \in \Omega : \mathbf{E}[\delta_{\mathrm{KL}}(H_n) | H_n \sim d_{\pi_b, \mathcal{H}_n}]| < \epsilon\}\right) = 1,$$

which, by the definition of almost sure convergence, means that

$$\mathbf{E}[\delta_{\mathrm{KL}}(H_n)|H_n \sim d_{\pi_b,\mathcal{H}_n}] \xrightarrow{a.s.} 0.$$

Appendix 3: Asymptotic variance of RIS and SEC

In this section we prove that the SEC estimator and, $\forall n$, RIS(n) has asymptotic variance at most that of the Monte Carlo estimator. These results are corollaries of Theorem 1 in Henmi et al. (2007) that holds for general Monte Carlo integration. Consider estimating $v = \mathbf{E}[f(X)|X \sim p]$ for probability mass function p and real-valued function f with domain $\mathcal{X}$. Note that while we define distributions as probability mass functions, this result can be applied to continuous-valued state and action spaces by replacing probability mass functions with density functions. Given parameterized and twice differentiable probability mass function $q(\cdot|\tilde{\boldsymbol{\theta}})$, the Monte Carlo estimator of v is $\tilde{v} := \frac{1}{m}\sum_{i=1}^{m} \frac{p(X_i)}{q(X_i,\tilde{\boldsymbol{\theta}})} f(X_i)$. Similarly, define $\hat{v} := \frac{1}{m}\sum_{i=1}^{m} \frac{p(X_i)}{q(X_i,\hat{\boldsymbol{\theta}})} f(X_i)$ where $\hat{\boldsymbol{\theta}}$ is the maximum likelihood estimate of $\tilde{\boldsymbol{\theta}}$ given samples from $q(\cdot|\tilde{\boldsymbol{\theta}})$. The following theorem relates the asymptotic variance of $\hat{v}$ to that of $\tilde{v}$.

Theorem 1

$$\mathrm{Var}_{\mathrm{A}}(\hat{v}) \leq \mathrm{Var}_{\mathrm{A}}(\tilde{v})$$

where $\mathrm{Var}_{\mathrm{A}}$ *denotes the asymptotic variance.*

Proof See Theorem 1 of Henmi et al. (2007).

Theorem 1 shows that an importance sampling estimate using the maximum likelihood estimate of the sampling distribution parameters yields an asymptotically lower variance estimate than using the true parameters, $\tilde{\boldsymbol{\theta}}$. To specialize this theorem to our setting, we show that the maximum likelihood behavior policy parameters are also the maximum likelihood parameters for the state-action distribution (for SEC) and the trajectory distribution (for RIS methods). We first need to specify the parameterized class of the sampling distribution. For SEC, the sampling distribution is $\Pr(S = s, A = a;\boldsymbol{\theta}) = d_{\pi}(s)\pi_{\boldsymbol{\theta}}(a|s)$. Note that the state distribution d_{π} is *not* parameterized by $\boldsymbol{\theta}$—only the policy, $\pi_{\boldsymbol{\theta}}$. This parameterization means that changing $\boldsymbol{\theta}$ leaves the distribution of states unchanged and is justified because we are only concerned with weighting already sampled data and not with collecting additional data. For RIS(n), the sampling distribution is $\Pr(H = h;\boldsymbol{\theta}) = p(h)w_{\pi_{\boldsymbol{\theta}}}(h)$ where $p(h) := d_0(s_0)\prod_{t=1}^{l-1} P(s_t|s_{t-1}, a_{t-1})$ and $w_{\pi_{\boldsymbol{\theta}}}(h) = \prod_{t=0}^{l-1} \pi_{\boldsymbol{\theta}}(a_t|s_{t-n}, a_{t-n}, \ldots, s_t)$.

We next present two lemmas that show that maximum likelihood estimation of the behavior policy is equivalent to maximum likelihood estimation of the specified sampling distributions. For SEC, we give the following lemma:

Lemma 2

$$\operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{i=1}^{k} \log \pi_{\boldsymbol{\theta}}(A_i|S_i) = \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{i=1}^{k} \log \Pr(S_k, A_k;\boldsymbol{\theta})$$

$$\operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{i=1}^{k} \log \pi_{\boldsymbol{\theta}}(A_i|S_i) = \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{i=1}^{k} \log \pi_{\boldsymbol{\theta}}(A_i|S_i) + \underbrace{\log d_{\pi}(S_i)}_{\text{const w.r.t.}} \boldsymbol{\theta}$$

$$= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{i=1}^{k} \log \Pr(S_i, A_i; \boldsymbol{\theta})$$

Proof

And for all RIS(n):

Lemma 3

$$\operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{i=1}^{m} \sum_{t=0}^{l-1} \log \pi_{\boldsymbol{\theta}}(a_t^i|s_{t-n}^i, a_{t-n}^i, \ldots, s_t^i) = \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{i=1}^{m} \log \Pr(h_i; \boldsymbol{\theta})$$

$$\operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{i=1}^{m} \sum_{t=0}^{l-1} \log \pi_{\boldsymbol{\theta}}(a_t^i|s_{t-n}^i, a_{t-n}^i, \ldots, s_t^i) = \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{i=1}^{m} \sum_{t=0}^{l-1} \log \pi_{\boldsymbol{\theta}}(a_t^i|s_{t-n}^i, a_{t-n}^i, \ldots, s_t^i)$$

$$+ \underbrace{\log d(s_0^i) + \sum_{t=1}^{l-1} \log P(s_t^i|s_{t-1}^i, a_{t-1}^i)}_{\text{const w.r.t.}} \boldsymbol{\theta}$$

$$= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{i=1}^{m} \log w_{\pi_{\boldsymbol{\theta}}}(h_i) + \log p(h_i)$$

$$= \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{i=1}^{m} \log \Pr(h_i; \boldsymbol{\theta})$$

Proof

Combining each of these lemmas in turn with Theorem 1 allows us to prove Corollaries 1 and 2 respectively.

Corollary 1 *Let* $\mathrm{Var}_{\mathrm{A}}(\mathrm{EST})$ *denote the asymptotic variance of estimator* EST. *Under Assumptions 4 and 5,*

$$\mathrm{Var}_{\mathrm{A}}(\mathrm{SEC}) \leq \mathrm{Var}_{\mathrm{A}}(\mathrm{MC}).$$

Proof Define $\mathcal{X} := \mathcal{S} \times \mathcal{A}$, $f(x) := \phi(s, a)$, $p(x) := \Pr(s, a|\pi)$ and $q(s, a|\boldsymbol{\theta}) := \Pr(s, a|\pi_{\boldsymbol{\theta}})$. Lemma 2 implies that:

$$\hat{\boldsymbol{\theta}} = \operatorname*{argmax}_{\boldsymbol{\theta} \in \Pi_{\boldsymbol{\theta}}} \sum_{i=1}^{k} \sum_{t=0}^{l-1} \log \pi_{\boldsymbol{\theta}}(a_j|s_j)$$

is the maximum likelihood estimate of $\tilde{\boldsymbol{\theta}}$ (where $\pi_{\hat{\boldsymbol{\theta}}} = \pi$ and $\Pr(s, a|\tilde{\boldsymbol{\theta}})$ is the probability of (s, a) under π) and then Corollary 1 follows directly from Theorem 1.

Corollary 2 *Under Assumptions 4 and 5,*$\forall n$,

$$\mathrm{Var}_{\mathrm{A}}(\mathrm{RIS}(n)(\pi, D)) \leq \mathrm{Var}_{\mathrm{A}}(\mathrm{OIS}(\pi, D, \pi_b))$$

where $\mathrm{Var}_{\mathrm{A}}$ *denotes the asymptotic variance.*

Proof Define $f(x) = g(h)$, $p(h) = \Pr(h|\pi_e)$ and $q(h|\boldsymbol{\theta}) = \Pr(h|\pi_{\boldsymbol{\theta}})$. Lemma 3 implies that:

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta} \in \boldsymbol{\Pi}_{\boldsymbol{\theta}}}{\operatorname{argmax}} \sum_{i=1}^{m} \sum_{t=0}^{l-1} \log \pi_{\boldsymbol{\theta}}(a_t^i | s_t^i)$$

is the maximum likelihood estimate of $\tilde{\boldsymbol{\theta}}$ (where $\pi_{\tilde{\boldsymbol{\theta}}} = \pi_b$ and $\Pr(h|\tilde{\boldsymbol{\theta}})$ is the probability of h under π_b) and then Corollary 2 follows directly from Theorem 1.

Note that for RIS(n) with $n > 0$, the condition that $\pi_{\tilde{\boldsymbol{\theta}}} \in \boldsymbol{\Pi}^n$ can hold even if the distribution of $A_t \sim \pi_{\tilde{\boldsymbol{\theta}}}$ (i.e., $A_t \sim \pi_b$) is only conditioned on s_t. This condition holds when $\exists \pi_{\boldsymbol{\theta}} \in \boldsymbol{\Pi}^n$ such that $\forall s_{t-n}, a_{t-n}, \ldots a_{t-1}$:

$$\pi_{\tilde{\boldsymbol{\theta}}}(a_t | s_t) = \pi_{\boldsymbol{\theta}}(a_t | s_{t-n}, a_{t-n}, \ldots, s_t),$$

i.e., the action probabilities only vary with respect to the immediate preceding state.

Appendix 4: SEC variance proof

In this appendix we prove Proposition 2 from Sect. 3.2:

Proposition 4 *Let* Var(EST) *denote the variance of estimator* EST. *Under Assumptions* 6 *and* 7, *for the Monte Carlo estimator*, MC, *and the SEC estimator*, SEC:

$$\text{Var}(\text{SEC}(B)) \leq \text{Var}(\text{MC}(B))$$

Recall that B is a set of state-action pairs collected by running the current policy π. Let X be the random variable representing the states observed in B and let U be the random variable representing the actions observed in B. We will sometimes write $\{X, U\}$ in place of B to make the composition of B explicit. Let $\text{Var}_X(\text{EST}(\{X, U\}))$ denote the variance of estimator EST with respect to the state set X. Let $\text{Var}_U(\text{EST}(\{X, U\})|X = \mathcal{X})$ denote the variance of estimator EST with respect to the action set U given $X = \mathcal{X}$

Under Assumptions 6 and 7, we make two claims about the SEC estimator, EST.

Claim 1 $\text{Var}_U(\text{SEC}(\{X, U\})|X = \mathcal{X})) = 0.$

Proof We can write either SEC or MC as:

$$\text{EST}(\{X, U\}) = \sum_{s \in \mathcal{S}} d_{\mathcal{B}}(s) \sum_{a \in \mathcal{A}} \pi_{\mathcal{B}}(a|s) w(s, a) \phi(s, a) \tag{31}$$

where $w(s, a) = \frac{\pi(a|s)}{\pi_{\mathcal{B}}(a|s)}$ for SEC and $w(s, a) = 1$ for MC. In Claim 1, the sampled states are fixed and variance only arises from $\pi_{\mathcal{B}}$ and $w(s, a)$ which vary for different realizations of $\mathbb{A}$. When we choose $w(s, a) = \frac{\pi_{\boldsymbol{\theta}}(a|s)}{\pi_{\mathcal{B}}(a|s)}$ (as SEC does) the $\pi_{\mathcal{B}}(a|s)$ factors cancel in Eq. 31. Since $\pi_{\mathcal{B}}$ is the only part of SEC that depends on the random variable U, using $w(s, a)$ eliminates variance due to action selection in the estimator. This proves Claim 1.

Claim 2 $\mathbf{E}_U \left[SEC(\{X, U\}) \,\middle|\, X \right] = \mathbf{E}_U \left[MC(\{X, U\}) \,\middle|\, X \right].$

Proof Claim 2 also follows from the same logic as Claim 1. The cancellation of the $\pi_{\mathcal{B}}(a|s)$ factors converts the inner summation over actions into an exact expectation under π. Since the Monte Carlo estimator is an unbiased estimator, the inner summation over actions must be equal to the exact expectation under π *in expectation*. Thus the expectation of both estimators *conditioned on X* is:

$$\mathbf{E}_U\left[\text{EST}(\{X,U\}) \;\middle|\; X\right] = \sum_{s\in\mathcal{S}} d_{\mathcal{B}}(s) \sum_{a\in\mathcal{A}} \pi(a|s) w(s,a) \phi(s,a). \tag{32}$$

This proves Claim 2.

We can now prove Proposition 2.

Proposition 5 *Let* Var(EST) *denote the variance of estimator* EST. *Under Assumptions 6 and 7, for the Monte Carlo estimator*, MC, *and the SEC estimator*, SEC:

$$\text{Var}(\text{SEC}(B)) \leq \text{Var}(\text{MC}(B))$$

Proof Using the law of total variance, the variance of the general estimator given by (31) can be decomposed as:

$$\text{Var}_{X,U}(\text{EST}) = \underbrace{\mathbf{E}\left[\text{Var}_U(\text{EST}(\{X,U\})) \;\middle|\; X \sim \pi\right]}_{\Sigma_U} + \underbrace{\text{Var}_X\left(\mathbf{E}\left[\text{EST}(\{X,U\}) \;\middle|\; U \sim \pi\right]\right)}_{\Sigma_X}$$

The first term, Σ_U, is the variance due to stochasticity in the action selection. From Claim 1, we know that for SEC this term is zero while in general it is not zero for MC.[5] The second term, Σ_X, is the variance due to only visiting a finite number of states before computing the estimate. Claim 2 shows that this term is equal for both SEC and MC. Thus the variance of SEC is at most that of MC.

Appendix 5: Connection to the REG estimator

In this section we show that SEC and RIS can be viewed as approximations of the REG estimator studied by Li et al. (2015). This connection is notable because Li et al. showed REG has asymptotically minimax optimal MSE, however, in MDPs, REG requires knowledge of the environment's state transition probabilities and initial state distribution probabilities 2015 while SEC and RIS do not.

Li et al. introduce the regression estimator (REG) for policy evaluation in multi-armed bandit problems 2015. We present it here as a general estimator for any function f. REG uses the available data to estimate the mean reward for each action as $f_D(a)$ and then computes the estimate:

[5] The Monte Carlo estimator has zero variance with respect to the sampled actions only when $\phi(s,a)$ is equal for all actions in any state.

$$\text{REG}(\pi, D) := \sum_{a \in \mathcal{A}} \pi(a) f_D(a).$$

In multi-armed bandit problems (MDPs with a single state and length one horizon), REG is identical to SEC and RIS(0) with f being either the function ϕ or χ respectively.

To apply REG to state-action expectations, one first estimates the mean ϕ value over (s, a) pairs as ϕ_D and then computes the estimate:

$$\text{REG}(\pi, D) = \sum_{S,A \in B} d_\pi(S)\pi(A|S)\phi_D(S, A)$$

This estimate requires knowledge of d_π and is thus inapplicable to general RL tasks. To apply REG to trajectory expectations, one first estimates the mean χ value for each observed trajectory as $\chi_D(H)$ and then computes the estimate:

$$\text{REG}(\pi, D) = \sum_{H \in D} \Pr(H|\pi)\chi_D(H)$$

This estimate requires knowledge of d_0 and P and is thus also inapplicable to general RL tasks.

We now elucidate a relationship between RIS($l-1$) and REG even though they are different estimators. Let $c(h)$ denote the number of times that trajectory h appears in D. We can rewrite REG as an importance sampling method:

$$\text{REG}(\pi, D) = \sum_{h \in \mathcal{H}} \Pr(h|\pi)\chi_D(h) \tag{33}$$

$$= \frac{1}{m} \sum_{h \in \mathcal{H}} c(h) \frac{\Pr(h|\pi)}{c(h)/m} \chi_D(h) \tag{34}$$

$$= \frac{1}{m} \sum_{i=1}^{m} \frac{\Pr(h_i|\pi)}{c(h_i)/m} \chi(h_i) \tag{35}$$

The denominator in (35) can be re-written as a telescoping product to obtain an estimator that is similar to RIS($l-1$):

$$\begin{aligned}
\text{REG}(\pi, D) &= \frac{1}{m} \sum_{i=1}^{m} \frac{\Pr(h_i|\pi)}{c(h_i)/m} \chi(h_i) \\
&= \frac{1}{m} \sum_{i=1}^{m} \frac{\Pr(h_i|\pi)}{\frac{c(s_0)}{m} \frac{c(s_0,a_0)}{c(s_0)} \cdots \frac{c(h_i)}{c(h_i/a_{l-1})}} \chi(h_i) \\
&= \frac{1}{m} \sum_{i=1}^{m} \frac{d_0(s_0)\pi(a_0|s_0)P(s_1|s_0,a_0)\cdots}{\hat{d}(s_0)\pi_D(a_0|s_0)\hat{P}(s_1|s_0,a_0)\cdots} \\
&\quad \frac{\cdots P(s_{l-1}|s_{l-2},a_{l-2})\pi(a_{l-1}|s_{l-1})}{\cdots \hat{P}(s_{l-1}|h_{0:l-1})\pi_D(a_{l-1}|h_{i:j})} \chi(h_i).
\end{aligned}$$

This expression differs from RIS($l-1$) in two ways:

1. The numerator includes the initial state distribution and transition probabilities of the environment.
2. The denominator includes count-based estimates of the initial state distribution and transition probabilities of the environment where the transition probabilities are conditioned on all past states and actions.

If we assume that the empirical estimates of the environment probabilities in the denominator are equal to the true environment probabilities then these factors cancel and we obtain the RIS($l-1$) estimate. This assumption will almost always be false except in deterministic environments. However, showing that RIS($l-1$) is approximating REG suggests that RIS($l-1$) may have similar theoretical properties to those derived for REG by Li et al. (2015). Our SinglePath experiment (See Fig. 10 in Sect. 6) supports this conjecture: RIS($l-1$) has high bias in the low to medium sample size but have asymptotically lower MSE compared to other methods. REG has even higher bias in the low to medium sample size range but has asymptotically lower MSE compared to RIS($l-1$). RIS with smaller n appear to decrease the initial bias but have larger MSE as the sample size grows. The asymptotic benefit of RIS for all n is also corroborated by Corollary 2 in "Appendix 3" though Corollary 2 does *not* tell us anything about how different RIS methods compare. The asymptotic benefit of REG compared to RIS methods can be understood as REG correcting for sampling error in both the action selection and state transitions. Similar conclusions can be drawn for a comparison between SEC and REG.

Acknowledgements We would like to thank Garrett Warnell, Ishan Durugkar, Philip Thomas, Qiang Liu, Faraz Torabi, Leno Felipe da Silva, Marc Bellemare, Finale Doshi-Velez, Brahma Pavse, Rich Sutton and the anonymous reviewers for insightful comments that suggested new directions to study and improved the final presentation of the work.

Funding This work has taken place in the Learning Agents Research Group (LARG) and the Personal Autonomous Robotics Lab (PeARL) at the Artificial Intelligence Laboratory, The University of Texas at Austin. LARG research is supported in part by grants from the National Science Foundation (CPS-1739964, IIS-1724157, NRI-1925082), the Office of Naval Research (N00014-18-2243), Future of Life Institute (RFP2-000), Army Research Office (W911NF-19-2-0333), DARPA, Lockheed Martin, General Motors, and Bosch. PeARL research is supported in part by the NSF (IIS-1724157, IIS-1638107, IIS-1749204, IIS-1925082) and ONR (N00014-18-2243). The views and conclusions contained in this document are those of the authors alone.

Compliance with ethical standards

Conflict of interest Peter Stone serves as the Executive Director of Sony AI America and receives financial compensation for this work. The terms of this arrangement have been reviewed and approved by the University of Texas at Austin in accordance with its policy on objectivity in research.

References

Asadi, K., Allen, C., Roderick, M., Mohamed, A.-R., Konidaris, G., & Littman, M. (2017). Mean actor critic. arXiv preprint arXiv:1709.00503v1.

Asis, K. D., Hernandez-Garcia, J. F., Holland, G. Z., & Sutton, R. S. (2018). Multi-step reinforcement learning: A unifying algorithm. In *Proceedings of the 32nd AAAI conference on artificial intelligence (AAAI)*.

Austin, P. C. (2011). An introduction to propensity score methods for reducing the effects of confounding in observational studies. *Multivariate Behavioral Research*, *46*(3), 399–424.

Bellman, R. (1966). Dynamic programming. *Science*, *153*(3731), 34–37.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). OpenAI gym. arXiv preprint arXiv:1606.01540.

Ciosek, K., & Whiteson, S. (2017). OFFER: Off-environment reinforcement learning. In *Proceedings of the 31st AAAI conference on artificial intelligence (AAAI)*.

Ciosek, K., & Whiteson, S. (2018). Expected policy gradients. In *Proceedings of the 32nd AAAI conference on artificial intelligence (AAAI)*.

Delyon, B., & Portier, F. (2016). Integral approximation by kernel smoothing. *Bernoulli*, *22*(4), 2177–2208.

Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., & Wu, Y. (2017). OpenAI baselines. https://github.com/openai/baselines.

Doroudi, S., Thomas, P. S., & Brunskill, E. (2017). Importance sampling for fair policy selection. In *Proceedings of uncertainty in artificial intelligence (UAI)*.

Dudík, M., Langford, J., & Li, L. (2011). Doubly robust policy evaluation and learning. In *Proceedings of the 28th international conference on machine learning (ICML)*.

Dvoretzky, A., Kiefer, J., & Wolfowitz, J. (1956). Asymptotic minimax character of the sample distribution function and of the classical multinomial estimator. *The Annals of Mathematical Statistics*, *27*(3), 642–669.

Farajtabar, M., Chow, Y., & Ghavamzadeh, M. (2018). More robust doubly robust off-policy evaluation. In *Proceedings of the 35th international conference on machine learning (ICML)*.

Fellows, M., Ciosek, K., & Whiteson, S. (2018). Fourier policy gradients. arXiv preprint arXiv:1802.06891.

Frank, J., Mannor, S., & Precup, D. (2008). Reinforcement learning in the presence of rare events. In *Proceedings of the 25th international conference on machine learning*, ACM, pp. 336–343.

Gelada, C., & Bellemare, M. G. (2019). Off-policy deep reinforcement learning by bootstrapping the covariate shift. *Proceedings of the AAAI Conference on Artificial Intelligence*, *33*, 3647–3655.

Greensmith, E., Bartlett, P. L., & Baxter, J. (2004). Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, *5*, 1471–1530.

Hallak, A., & Mannor, S. (2017). Consistent on-line off-policy evaluation. In *Proceedings of the 34th international conference on machine learning*, pp. 1372–1383.

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

Hanna, J. P., & Stone, P. (2019). Reducing sampling error in the monte carlo policy gradient estimator. In *Proceedings of the 19th international conference on autonomous agents and multi-agent systems (AAMAS)*.

Hanna, J. P., Thomas, P. S., Stone, P., & Niekum, S. (2017). Data-efficient policy evaluation through behavior policy search. In *Proceedings of the 34th international conference on machine learning (ICML)*.

Hanna, J. P., Niekum, S., & Stone, P. (2019). Importance sampling with an estimated behavior policy. In *Proceedings of the 36th international conference on machine learning (ICML)*.

Henmi, M., Yoshida, R., & Eguchi, S. (2007). Importance sampling via the estimated sampler. *Biometrika*, *94*(4), 985–991.

Hirano, K., Imbens, G. W., & Ridder, G. (2003). Efficient estimation of average treatment effects using the estimated propensity score. *Econometrica*, *71*(4), 1161–1189.

Jiang, N., & Li, L. (2016). Doubly robust off-policy evaluation for reinforcement learning. In *Proceedings of the 33rd international conference on machine learning (ICML)*.

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *Proceedings of the international conference on learning representations (ICLR)*.

Li, L., Munos, R., & Szepesvári, C. (2015). Toward minimax off-policy value estimation. In *Proceedings of the 18th international conference on artificial intelligence and statistics*.

Liu, Q., & Lee, J. D. (2017). Black-box importance sampling. In *Proceedings of the 20th international conference on artificial intelligence and statistics*.

Liu, Q., Li, L., Tang, Z., & Zhou, D. (2018). Breaking the curse of horizon: Infinite-horizon off-policy estimation. *Advances in Neural Information Processing Systems (NeurIPS)*, *31*, 5356–5366.

Mahadevan, S. (1996). Average reward reinforcement learning: Foundations, algorithms, and empirical results. *Machine Learning*, *1*, 159–196.
Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. Cambridge: Cambridge University Press.
Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533.
Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., & Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. In *Proceedings of the 33rd international conference on machine learning (ICML)*, pp. 1928–1937.
Moore, A (1990). Efficient Memory-based learning for robot control. PhD thesis, University of Cambridge.
Mousavi, A., Li, L., Liu, Q., & Zhou, D. (2020). Black-box off-policy estimation for infinite-horizon reinforcement learning. In *International conference on learning representations (ICLR)*
Narita, Y., Yasui, S., & Yata, K. (2019). Efficient counterfactual learning from bandit feedback. In *Proceedings of the 35th AAAI conference on artificial intelligence (AAAI)*.
Oates, C. J., Girolami, M., & Chopin, N. (2017). Control functionals for monte carlo integration. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *79*(3), 695–718.
Pavse, B. S., Durugkar, I., Hanna, J. P., & Stone, P. (2020). Reducing sampling error in batch temporal difference learning. In *Proceedings of the 37th international conference on machine learning (ICML)*.
Peters, J., & Schaal, S. (2008). Reinforcement learning of motor skills with policy gradients. *Neural Networks, 21*(4), 682–697.
Petit, B., Amdahl-Culleton, L., Liu, Y., Smith, J., & Bacon, P. L. (2019). All-action policy gradient methods: A numerical integration approach. arXiv preprint arXiv:1910.09093.
Precup, D., Sutton, R. S., & Singh, S. (2000). Eligibility traces for off-policy policy evaluation. In *Proceedings of the 17th international conference on machine learning (ICML)*, pp. 759–766.
Puterman, M. L. (2014). *Markov decision processes: Discrete stochastic dynamic programming*. New York: Wiley.
Raghu, A., Gottesman, O., Liu, Y., Komorowski, M., Faisal, A., Doshi-Velez, F., & Brunskill, F. (2018). Behaviour policy estimation in off-policy policy evaluation: Calibration matters. In *Proceedings of the ICML workshop on causal inference, counterfactual prediction, and autonomous action*.
Rosenbaum, P. R. (1987). Model-based direct adjustment. *Journal of the American Statistical Association*, *82*(398), 387–394.
Rubinstein, R. Y., & Kroese, D. P. (2013). *The cross-entropy method: a unified approach to combinatorial optimization Monte Carlo simulation and machine learning*. Berlin: Springer.
Rummery, G. A., & Niranjan, M. (1994). *On-line Q-learning using connectionist systems*, Vol. 37. Department of Engineering Cambridge, England: University of Cambridge.
Schulman, J., Levine, S., Moritz, P., Jordan, M., & Abbeel, P. (2015). Trust region policy optimization. In *Proceedings of the 32nd international conference on machine learning (ICML)*. URL http://jmlr.csail.mit.edu/proceedings/papers/v37/schulman15.html.
Schulman, J., Moritz, P., Levine, S., Jordan, M. I., & Abbeel, P. (2016). High-dimensional continuous control using generalized advantage estimation. In *Proceedings of the international conference on learning representations (ICLR)*.
Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347.
Schwartz, A. (1993). A reinforcement learning method for maximizing undiscounted rewards. In *Proceedings of the 10th international conference on machine learning (ICML)*.
Shi, L., Li, S., Cao, L., Yang, L., & Pan, G. (2019). TBQ (σ): Improving efficiency of trace utilization for off-policy reinforcement learning. In *Proceedings of the 18th international conference on autonomous agents and multiagent systems (AAMAS)*, pp. 1025–1032.
Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., et al. (2016). Mastering the game of go with deep neural networks and tree search. *Nature*, *529*(7587), 484–489.
Singh, S. P., & Sutton, R. S. (1996). Reinforcement learning with replacing eligibility traces. *Machine Learning*, *22*, 123–158.
Sutton, R. S. (1984). Temporal credit assignment in reinforcement learning. PhD thesis, University of Massachusetts, Amherst.
Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction*. Cambridge: MIT Press.
Sutton, R. S., Singh, S., & McAllester, D. (2000). Comparing policy-gradient algorithms.
Thomas, P. S. (2015). Safe reinforcement learning. PhD thesis, University of Massachusetts Amherst.
Thomas, P. S., & Brunskill, E. (2016a). Data-efficient off-policy policy evaluation for reinforcement learning. In *Proceedings of the 33rd international conference on machine learning (ICML)*.

Thomas, P. S., & Brunskill, E. (2016b). Magical policy search: Data efficient reinforcement learning with guarantees of global optimality. In *European workshop on reinforcement learning*.

Thomas, P. S., & Brunskill, E. (2017). Importance sampling with unequal support. In *Thirty-first AAAI conference on artificial intelligence*.

Thomas, P. S., Theocharous, G., & Ghavamzadeh, M. (2015). High confidence policy improvement. In *Proceedings of the 32nd international conference on machine learning (ICML)*.

Van Seijen, H., Van Hasselt, H., Whiteson, S., & Wiering, M. (2009). A theoretical and empirical analysis of expected SARSA. In *Proceedings of the IEEE symposium on adaptive dynamic programming and reinforcement learning*, IEEE, pp. 177–184.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, *8*(3–4), 229–256.

Xie, Y., Liu, B., Liu, Q., Wang, Z., Zhou, Y., & Peng, J. (2018). Off-policy evaluation and learning from logged bandit feedback: Error reduction via surrogate policy. In *Proceedings of the international conference on learning representations (ICLR)*.

Yang, L., Shi, M., Zheng, Q., Meng, W., & Pan, G. (2018). A unified approach for multi-step temporal-difference learning with eligibility traces in reinforcement learning. In *Proceedings of the 27th international joint conference on artificial intelligence (IJCAI)*.

Yang, M., Nachum, O., Dai, B., Li, L., & Schuurmans, D. (2020). Off-policy evaluation via the regularized lagrangian. In *Advances in neural information processing systems (NeurIPS)*, Vol. 33.