SAME NOISE, BETTER GRADIENTS: POLICY OPTIMIZATION WITH COMMON RANDOM NUMBERS

Anonymous authors
Paper under double-blind review

ABSTRACT

We study common random numbers as a variance-reduction method for policy-gradient reinforcement learning in stochastic simulators. The core idea is to decouple the random number generators for action selection and state transitions, and investigate counterfactual effects of actions under the same state transition noise. For policy gradient algorithms, we implement this idea as a control-variate that depends on the state transition noise, and prove that it is unbiased and obtains lower variance in certain conditions. Further, we propose two simple modifications of the popular PPO algorithm based on either (1) resetting the random number generator, and (2) learning a value function that depends on the random seed. We demonstrate improved performance on various stochastic domains.

1 INTRODUCTION

Policy gradient algorithms are a staple of reinforcement learning (RL, (Mannor et al., 2026)), and variants such as PPO (Schulman et al., 2017b) have been used in applications ranging from computer games (Berner et al., 2019) to robotic locomotion (Tan et al., 2018) and large language models (Ouyang et al., 2022). To date, most policy gradient successes relied on extensive simulations, and techniques such as domain randomization are commonly applied to learn robust policies that transfer to the real world by injecting noise and applying variations to the simulator (Hwangbo et al., 2019).

Policy gradient algorithms learn from trial and error, using rollouts from a random policy to estimate the effect of actions on future returns. When the environment itself is noisy, the estimator mixes together the influences of the actions and the noise realizations, resulting in excess estimation variance that can adversely affect learning. To see this, consider the example in Figure 1 with two actions, A and B. Upon selecting an action, a random variable $X \sim \text{Uniform}[0, 1]$ is sampled. If A was chosen and $X > 0.5$ reward of 1 is obtained, otherwise the reward is 0. If, on the other hand, B was chosen, a non-zero reward of 1 is obtained only if $X > 0.6$. Assume that the initial policy selects A with probability 0.5. A rollout sequence here can be, for example, $\{A, 1\}$ or $\{B, 0\}$, and on average, rollouts with action A will have a higher non-zero reward count. However, there is a non-trivial chance[1] that the B samples will have a higher reward count, directing the algorithm to learn the wrong action preference.

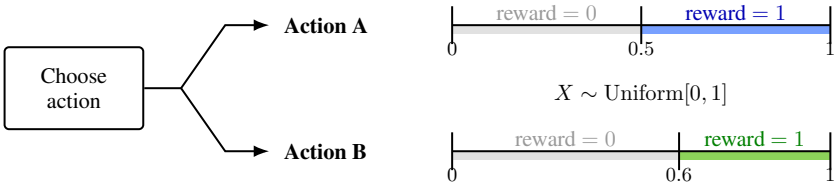


Figure 1: Action A yields reward 1 when $X > 0.5$; action B yields reward 1 only when $X > 0.6$.

We can be more clever, however, if we have access to the sampled X and ask: under this specific noise, what *would have been* the reward for the other action? In this case, only when $0.5 < X < 0.6$ we would count a difference, and B would never be preferred.

[1] For example, larger than 16% when taking 10 samples from each action.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

The insight that counterfactual action evaluation can improve policy gradient estimation was presented in several studies (Mao et al., 2018; Nota et al., 2021). However, designing effective and scalable algorithms that exploit this idea has remained a challenge. Nota et al. (2021) showed results on a small gridworld, Mao et al. (2018) required observation of the noise sequence and trained recurrent neural networks to process it, while Mesnard et al. (2021) represented future information using a hindsight network and hindsight predictor, and special loss functions. Combining these methods with off-the-shelf, general-purpose RL implementations is not trivial. In addition, all prior work required access to the specific noise instances, which is not generally available in off-the-shelf simulators. In this work we propose simple modifications of the popular PPO algorithm that exploit counterfactual actions *without* recurrent neural networks or complex loss functions and demonstrate results on standard benchmarks and comparisons with standard PPO implementations. Our key insight is that we can evaluate counterfactual actions *by controlling the simulator's random number generator*, without requiring access to the simulator's internal implementation of the stochastic dynamics. In the example above, we can evaluate the counterfactual reward without observing X directly, but just querying the reward for the alternative action with the same noise realization.

We investigate two algorithms. The first focuses on temporal difference (TD, Sutton et al. 1998) estimates – a key quantity in the generalized advantage estimation (GAE, Schulman et al. 2016) component of PPO. We propose a lower-variance TD estimate by averaging multiple queries to the simulator's next state under different actions but *the same noise realization*, effectively canceling the noise due to stochastic transitions. Our second algorithm leverages the insight that the simulator's random seed controls the complete sequence of noise variables in a rollout, and we propose to learn a different value function baseline for every seed. This method resembles the noise-conditioned value of Mao et al. (2018), but does not require observing or processing the noise sequence.

Both our algorithms involve minor modifications to PPO, and are compatible with off-the-shelf implementations such as Stable Baselines (Raffin et al., 2021). Our results demonstrate improved sample complexity and accuracy on several stochastic benchmark domains and altered Pufferlib domains. **Code will be open sourced.**

2 Related Work

The use of baselines to reduce policy-gradient variance dates back to Williams (1992). Weaver & Tao (2001) and Greensmith et al. (2004) derived variance-optimal baselines for policy-gradient estimation. Our work extends this line by conditioning the baseline on environment noise. Liu et al. (2018) proposed action-dependent control variates using Stein's identity, requiring correction terms to reduce bias.

Common random numbers (CRN) is a classical variance-reduction technique in Monte Carlo simulation (L'Ecuyer, 1994; Glasserman, 2004), and has been studied in stochastic optimization (Shapiro et al., 2021) and early RL approaches (Ng & Jordan, 2000). To our knowledge, the application of CRN to policy gradients has not been explored.

The idea of conditioning the policy gradient baseline on future events has been investigated in several studies (Mao et al., 2018; Nota et al., 2021; Harutyunyan et al., 2019; Guez et al., 2020; Mesnard et al., 2021), and our theory in Section 4 is closely related to earlier results in (Mao et al., 2018; Nota et al., 2021). Our main contribution, however, is designing novel and simple algorithms to implement these baselines. In comparison, the baselines in (Mao et al., 2018; Guez et al., 2020; Mesnard et al., 2021) required architectures that process a *sequence* of future observations, such as recurrent neural networks, and (Mao et al., 2018) requires knowledge of the noise sequence, which is not necessarily available in general. Our baselines can be directly plugged into standard RL libraries and environments, and require only the ability to reset the simulator's random number generator. Sinclair et al. (2023) recently considered RL algorithms for MDPs with *exogenous noise* that exploit specific noise sequences; our method is not specific to exogenous MDPs.

Our Seed-Conditioned Value Baseline is an instance of an asymmetric critic (Pinto et al., 2017), a technique that has been used also for multi-agent RL (Lowe et al., 2017). To our knowledge, conditioning the critic on the random seed is novel.

An especially relevant precursor is the vine sampling variant of TRPO, which evaluates multiple actions from the same state and uses common random numbers to reduce the variance of estimated

Q-value. (Schulman et al., 2017a). Our approach operates similarly by exploiting the shared simulator randomness to build a counterfactual baseline in PPO that is conditioned on the same noise.

3 Background and Problem Formulation

We consider a discounted Markov decision process (MDP, (Mannor et al., 2026)) in finite time with state space $\mathcal{S}$, action space $\mathcal{A}$, deterministic reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and discount factor $\gamma \in [0, 1)$. The usual stochastic transition kernel is $P(\cdot \mid s, a)$, and states evolve as $s_{t+1} \sim P(\cdot \mid s_t, a_t)$. A stochastic Markov policy $\pi_\theta(a \mid s)$ parameterized by $\theta \in \mathbb{R}^d$ induces an action distribution $a_t \sim \pi_\theta(\cdot \mid s_t)$.

The standard state and state-action value functions for a policy π are:

$$V^\pi(s) = \mathbb{E}^\pi \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \middle| s_t = s \right], \quad Q^\pi(s, a) = \mathbb{E}^\pi \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \middle| s_t = s, a_t = a \right],$$

and are known to satisfy the Bellman equation, $V^\pi(s) = \sum_a \pi(a|s) \left(r(s, a) + \sum_{s'} P(s' \mid s, a) V^\pi(s') \right)$, $Q^\pi(s, a) = r(s, a) + \sum_{s'} P(s' \mid s, a) V^\pi(s')$, and $V^\pi(s) = \mathbb{E}_{a \sim \pi(\cdot|s)}[Q^\pi(s, a)]$ (Mannor et al., 2026). The advantage function is their difference: $A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$.

Our objective is to maximize the expected discounted return, $J(\theta) = \mathbb{E}^{\pi_\theta} \left[\sum_{t \geq 0} \gamma^t r(s_t, a_t) \right]$. The policy gradient theorem is the basis for algorithms that estimate $\nabla_\theta J(\theta)$ from rollouts and use it to ascend on $J(\theta)$, such as REINFORCE and PPO (Mannor et al., 2026; Williams, 1992; Schulman et al., 2017b). Define the unnormalized discounted state visitation frequency: $d_\gamma^{\pi_\theta}(s) := \sum_{t=0}^{\infty} \gamma^t \Pr_{\pi_\theta}(s_t = s)$. The policy gradient theorem states that:

$$\nabla_\theta J(\theta) = \mathbb{E}_{s \sim d_\gamma^{\pi_\theta},\, a \sim \pi_\theta(\cdot|s)} [\nabla_\theta \log \pi_\theta(a \mid s) \left(Q^{\pi_\theta}(s, a) - b(s_t) \right)], \tag{1}$$

where $b(s)$ is any state-dependent baseline function. In practice, $\nabla_\theta J(\theta)$ must be estimated from samples, and the baseline (also known as a *control variate*) can be chosen to reduce estimation variance. A common choice is the value function $V^{\pi_\theta}(s)$, corresponding to $\nabla_\theta J(\theta) = \mathbb{E}_{s \sim d_\gamma^{\pi_\theta},\, a \sim \pi_\theta(\cdot|s)} [\nabla_\theta \log \pi_\theta(a \mid s) A^{\pi_\theta}(s, a)]$, and a common advantage estimator is the temporal-difference (TD):

$$\delta_t := r(s_t, a_t) + \gamma V^{\pi_\theta}(s_{t+1}) - V^{\pi_\theta}(s_t). \tag{2}$$

The generalized advantage estimate (GAE, (Schulman et al., 2016)) generalizes the TD estimator, and has become standard in PPO implementations (Raffin et al., 2021),

$$\widehat{A}_t^{\mathrm{GAE}(\lambda)} := \sum_{\ell=0}^{\infty} (\gamma\lambda)^\ell \delta_{t+\ell}, \qquad \lambda \in [0, 1]. \tag{3}$$

4 Variance Reduction using Common Random Numbers

In this section we present methods that exploit common random numbers for policy gradient variance reduction. We begin with a general formulation, and proceed with concrete implementations.

We follow the MDP problem formulation of Section 3. For our purposes it is helpful to write the state transition in its equivalent noise-explicit form,

$$s_{t+1} = f(s_t, a_t, \xi_t), \qquad \xi_t \sim P_\xi, \tag{4}$$

where P_ξ is the noise distribution, and we assume that ξ_t are sampled i.i.d from P_ξ (Mannor et al., 2026).

Similarly, we write the random action as $a_t = g_\theta(s_t, \zeta_t)$, where the action noise $\zeta_t \sim P_\zeta$ is i.i.d. and independent of ξ_t. Our subsequent formulation will exploit the separation between the random variables ξ_t and ζ_t.

For our subsequent development, we define value functions for a specific noise instances.

Definition 1. *The noise-dependent state-action value function is* $Q^\pi(s,a,\xi) = r(s,a) + V^\pi(f(s,a,\xi))$. *The noise-dependent state value function is* $V^\pi(s,\xi) = \sum_a \pi(a|s)\,(r(s,a) + V^\pi(f(s,a,\xi)))$.

Note that by (4) we have that $\mathbb{E}_\xi[V^\pi(s,\xi)] = V^\pi(s)$ and $\mathbb{E}_\xi[Q^\pi(s,a,\xi)] = Q^\pi(s,a)$.

Also note that we can write the TD update of Eq. (2) as:

$$\delta_t = Q^{\pi_\theta}(s_t,a_t,\xi_t) - V^{\pi_\theta}(s_t). \tag{5}$$

Explicitly accounting for the noise realizations in our terminology will allow us to develop a theory for exploiting noise realizations in our algorithms. Our first observation is that we can use a *noise-dependent* baseline in the policy gradient without incurring bias.

Theorem 2 (Noise-dependent baselines are unbiased)**.** *Let* $b(s,\xi)$ *be any scalar baseline function that may depend on the state and on the realized environment noise, but does not depend on the sampled action* a. *Then* $\mathbb{E}_{s\sim d_\gamma^{\pi_\theta},\, a\sim\pi_\theta(\cdot|s),\, \xi\sim P_\xi}\left[\nabla_\theta \log \pi_\theta(a \mid s)\big(Q^{\pi_\theta}(s,a,\xi) - b(s,\xi)\big)\right] = \nabla_\theta J(\theta)$.

Proof. Define the score term $w_t := \nabla_\theta \log \pi_\theta(a_t \mid s_t)$. Conditioned on s_t, the score has zero mean under the policy: $\mathbb{E}[w_t \mid s_t] = \mathbb{E}_{a_t\sim\pi_\theta(\cdot|s_t)}[\nabla_\theta \log \pi_\theta(a_t \mid s_t)] = 0$. Because the policy noise is independent of the environment noise, $b(s_t,\xi_t)$ is also independent of a_t. Therefore $\mathbb{E}_{a_t\sim\pi(\cdot|s_t)}[w_t\, b(s_t,\xi_t) \mid s_t,\xi_t] = b(s_t,\xi_t)\mathbb{E}_{a_t\sim\pi(\cdot|s_t)}[w_t \mid s_t] = 0$. □

Following the common choice of using the value function as a baseline, we define the *counterfactual value baseline*, $V^{\pi_\theta}(s,\xi)$. Intuitively, the counterfactual advantage $Q^{\pi_\theta}(s,a,\xi) - V^{\pi_\theta}(s,\xi)$ measures the difference between the expected return from s upon taking action a and the average return from s, when the noise for the transition at s *is the same*. We can also consider the counterfactual difference between $Q^{\pi_\theta}(s,a,\xi)$ and the value of an action a^* as a baseline, so long as a^* is independent of a_t in w_t. Both baselines are unbiased.

Corollary 3 (Unbiased counterfactual baselines)**.** *The baseline* $V^\pi(s,\xi_t)$ *is unbiased. For any action* a^* *that is independent of* a_t *in* w_t, *the baseline* $Q^\pi(s,a^*,\xi_t)$ *is also unbiased.*

We next propose that counterfactual value baselines *reduce estimation variance*. The next theorem establishes a key result.

Theorem 4 (Counterfactual baselines reduce variance)**.** *Fix a state* $s \in \mathcal{S}$. *Let* $a \sim \pi_\theta(\cdot \mid s)$ *and* $\xi \sim P_\xi$ *be independent, and for notational convenience, substitute* $Z := Q^{\pi_\theta}(s,a,\xi)$. *Then,*

$$\mathrm{Var}_{a\sim\pi_\theta(\cdot|s),\,\xi\sim P_\xi}\left(Z - V^{\pi_\theta}(s)\right) = \mathrm{Var}_{a\sim\pi_\theta(\cdot|s),\,\xi\sim P_\xi}\left(Z - V^{\pi_\theta}(s,\xi)\right) + \mathrm{Var}_{\xi\sim P_\xi}\left(V^{\pi_\theta}(s,\xi)\right).$$

Consequently,

$$\mathrm{Var}_{a\sim\pi_\theta(\cdot|s),\,\xi\sim P_\xi}\left(Q^{\pi_\theta}(s,a,\xi) - V^{\pi_\theta}(s,\xi)\right) \le \mathrm{Var}_{a\sim\pi_\theta(\cdot|s),\,\xi\sim P_\xi}\left(Q^{\pi_\theta}(s,a,\xi) - V^{\pi_\theta}(s)\right).$$

Proof. See Appendix E. □

Theorem 4 suggests that the counterfactual value can be used as a lower-variance advantage estimate. However, it is not clear that a reduction in the variance of the advantage translates to a reduction of the variance in the policy gradient, $\nabla_\theta \log \pi_\theta(a \mid s)\, A(s,a,\xi)$. We next investigate this question.

For notational simplicity, in the following we assume that θ is a scalar. The results naturally extend to the vector case.

Proposition 5 (Optimal baseline)**.** *The variance-minimizing baseline is* $b^*(s,\xi) = \frac{\mathbb{E}_{a_t\sim\pi_\theta(\cdot|s_t)}\left[w_t^2 Q^{\pi_\theta}(s_t,a_t,\xi_t)|s_t=s,\xi_t=\xi\right]}{\mathbb{E}_{a_t\sim\pi_\theta(\cdot|s_t)}[w_t^2|s_t=s,\xi_t=\xi]}$ *(Weaver & Tao, 2001; Greensmith et al., 2004). Define* $g_{t;b} := w_t\big(Q^{\pi_\theta}(s_t,a_t,\xi_t) - b(s_t,\xi_t)\big)$, $\sigma(s,\xi) := \mathbb{E}_{a_t\sim\pi_\theta(\cdot|s_t)}[w_t^2 \mid s_t = s, \xi_t = \xi]$. *For any baseline* $b(s,\xi)$, $\mathrm{Var}(g_{t;b}) = \mathrm{Var}(g_{t;b^*}) + \mathbb{E}\Big[\sigma(s_t,\xi_t)\big(b(s_t,\xi_t) - b^*(s_t,\xi_t)\big)^2\Big]$.

Proof. See Appendix C. □

Theorem 5 extends the optimal baseline results of (Weaver & Tao, 2001; Greensmith et al., 2004) to counterfactual baselines. The next result show a sufficient condition for $V^{\pi_\theta}(s,\xi)$ to be an optimal baseline, and evaluates its variance reduction compared to $V^{\pi_\theta}(s)$.

Corollary 6. *If for all s_t, ξ_t it holds* $\mathrm{Cov}_{a_t\sim\pi_\theta(\cdot|s_t)}\big(Q^{\pi_\theta}(s_t,a_t,\xi_t), w_t^2 \mid s_t,\xi_t\big) = 0$ *, then* $b^*(s,\xi) = V^{\pi_\theta}(s,\xi)$*, and*

$$\mathrm{Var}(g_{t;V^{\pi_\theta}(s)}) - \mathrm{Var}(g_{t;V^{\pi_\theta}(s,\xi)}) = \mathbb{E}\Big[\sigma(s_t,\xi_t)\big(V^{\pi_\theta}(s_t) - V^{\pi_\theta}(s_t,\xi_t)\big)^2\Big] \geq 0.$$

Proof. See Appendix D. □

The next result shows that when the noise has has *additive* effect on the value function, the counterfactual baseline also has reduced variance.

Proposition 7. *If there exist functions $h(s,a), \phi(\xi,s)$ such that $Q^{\pi_\theta}(s,a,\xi) = h(s,a) + \phi(\xi,s)$, we have that* $\mathrm{Var}(g_{t;V^{\pi_\theta}(s)}) - \mathrm{Var}(g_{t;V^{\pi_\theta}(s,\xi)}) = \mathbb{E}\big[w_t^2(\phi(\xi,s) - \mathbb{E}[\phi(\xi,s)|s])^2\big] \geq 0$.

Proof. See Appendix F. □

4.1 Practical Counterfactual Value Function Estimation

From the discussion above, it is clear that the counterfactual value baseline can reduce variance in policy gradient algorithms. However, practically estimating $V^\pi(s,\xi)$ is not trivial, as ξ is not typically exposed in standard simulators. In this section we show that by simply controlling the seed of the random number generator, we can estimate the counterfactual advantage without directly observing ξ. This leads to a practical "plug-in" method for modern implementations of algorithms such as PPO.

We denote by $V_\phi(s)$ a parametrized value function approximation of $V^\pi(s)$, such as a neural network. Standard methods for learning $V_\phi(s)$ include least squares and TD; in this work we are only concerned in advantage estimation, and leave the specifics of learning $V_\phi(s)$ implicit. Furthermore, common RL domains have *terminal states* that immediately end the rollout, and we denote by $D_t \in \{0,1\}$ a termination signal obtained after taking action a_t in state s_t.

Exact Counterfactual Value Function Estimator The first estimator we propose, termed Exact Counterfactual Value Function Estimator, is based on the explicit formula for $V^\pi(s,\xi)$. Given a state s_t, a realized simulator-noise instance ξ_t, and a learned value approximation $V_\phi(s)$, the counterfactual value is (cf. Definition 1)

$$\widehat{V}_\phi^{\mathrm{CFVE}}(s_t,\xi_t) := \sum_{a\in\mathcal{A}} \pi_\theta(a \mid s_t)\left[r(s_t,a) + \gamma(1 - D(s_t,a,\xi_t))V_\phi(f(s_t,a,\xi_t))\right], \tag{6}$$

where $D(s_t,a,\xi_t) \in \{0,1\}$ denotes the termination indicator induced by applying action a at state s_t under the realized noise ξ_t.

The temporal difference with the learned value and termination signal, following (5), is:

$$\delta_t^{\mathrm{CFVE}} := r_t + \gamma(1 - D(s_t,a_t,\xi_t))V_\phi(f(s_t,a_t,\xi_t)) - \widehat{V}_\phi^{\mathrm{CFVE}}(s_t,\xi_t). \tag{7}$$

The generalized advantage estimate can be formed, using the temporal difference terms:

$$\widehat{A}_t^{\mathrm{CFVE}} := \sum_{\ell=0}^{\infty} (\gamma\lambda)^\ell \delta_{t+\ell}^{\mathrm{CFVE}}. \tag{8}$$

In Algorithm 1 in Appendix A we show how to calculate the temporal difference term (7) without access to ξ_t, by resetting the state of the random number generator. Modifications to standard PPO are colored in Blue: note that our implementation only updates the calculation of the temporal difference term, and only requires to reset and step the simulator. Thus, it can easily be plugged into modern PPO implementations such as (Raffin et al., 2021).

Performance Tradeoffs: Based on our variance analysis above, we expect Algorithm 1 to produce more accurate gradient estimates from each rollout, and therefore converge with less gradient descent steps. This, however, comes at a computational cost: we require an additional simulator step for every action in $\mathcal{A}$, at each rollout step. The actual cost depends on the speed of resetting and stepping the simulator, and on the number of actions. We emphasize that Algorithm 1 does not affect the training of the policy neural network (e.g., using the PPO objective), which can be a dominating factor when the policy is large. In our experiments we investigate the variance/computation tradeoff empirically.

Sampled Counterfactual Value Function Estimators When the action space is large or continuous, enumerating all actions is not practical, and Algorithm 1 cannot be used. In this case, we approximate the noise-conditioned value function by Monte Carlo sampling counterfactual actions from the policy. Given M independently sampled actions

$$\widetilde{a}_t^{(1)}, \ldots, \widetilde{a}_t^{(M)} \overset{\text{i.i.d.}}{\sim} \pi_\theta(\cdot \mid s_t), \qquad \widetilde{a}_t^{(m)} \perp a_t \quad \text{conditional on } s_t,$$

we define the sampled counterfactual value estimator

$$\widehat{V}^{\mathrm{SCFVE}}_{\phi,M}(s_t, \xi_t) := \frac{1}{M} \sum_{m=1}^{M} \left[r(s_t, \widetilde{a}_t^{(m)}) + \gamma\big(1 - D(s_t, \widetilde{a}_t^{(m)}, \xi_t)\big) V_\phi\big(f(s_t, \widetilde{a}_t^{(m)}, \xi_t)\big) \right]. \quad (9)$$

Algorithm 2, detailed in Appendix A, replaces the $\widehat{V}^{\mathrm{CFVE}}_{\phi}(s_t, \xi_t)$ computation in Lines 3-6 of Algorithm 1 with the Monte Carlo average $\widehat{V}^{\mathrm{SCFVE}}_{\phi,M}(s_t, \xi_t)$. This advantage estimator is still unbiased, requires only M additional simulator steps, but suffers additional variance due to the Monte Carlo approximation that decreases with M.

In particular, we term the estimator with $M = 1$ a *Single-Action Counterfactual Value Function Estimator*, and note that it only requires a single additional simulator step for every rollout step. In practice, we found that instead of sampling a single action, a useful alternative is selecting the most likely action $\widetilde{a}_t^{(1)} \in \arg\max_a \pi_\theta(a \mid s_t)$. Corollary 3 shows that this baseline is also unbiased.

Threshold for Near-Deterministic Policies The sampled CFVE estimators above are most useful while the policy is still exploratory. In this phase, independently sampled counterfactual actions differ meaningfully from the factual action, and evaluating them under the same simulator noise gives an informative estimate of the noise-conditioned baseline. Empirically, however, we observed a recurring late-training failure mode.

Note that if all the sampled actions are the same as the factual action, i.e., $\widetilde{a}_t^{(i)} = a_t \quad \forall i \in \{1, \ldots, M\}$, we obtain that the advantage $r_t + \gamma(1 - D(s_t, a_t, \xi_t))V_\phi(f(s_t, a_t, \xi_t)) - \widehat{V}^{\mathrm{SCFVE}}_{\phi,M}(s_t, \xi_t) = 0$. This can happen when the policy is near-deterministic, and in this case the sampled CFVE estimator yields little benefit. In this case, we found that reverting to the standard value baseline (2) improves performance.

Algorithmically, we detect this regime by a simple threshold on the average action entropy. We term this an *entropy-gated estimator*; see Appendix A for more details.

Seed-Conditioned Value Baseline In this section we propose a simple alternative to the CFVE family of estimators that does not require additional simulator steps, and exploits the RNG in a different way. Our key observation is that by having two separate RNGs for state transitions and action selection, we can easily "fix" the state transition noise, while maintaining the action stochasticity required for estimating the policy gradient.

Let $\psi_1, \ldots, \psi_N$ denote N random seeds that control the generation of ξ_t. That is, for a fixed seed ψ_i, the sequence of noise variables is constant.

We propose to train N *seed-conditioned* value functions $V_{\phi_1}(s), \ldots, V_{\phi_N}(s)$, corresponding to the N random seeds. To sample a rollout, we first sample $i \sim \mathrm{Uniform}(1, N)$, and reset the state transition RNG to ψ_i. The action RNG is initialized randomly. We calculate the temporal differences for the rollout using $V_{\phi_i}(s)$, $\delta_t := r(s_t, a_t) + \gamma V_{\phi_i}(s_{t+1}) - V_{\phi_i}(s_t)$, and plug in these temporal differences into standard PPO.

Note that, in contrast to previous approaches such as (Mao et al., 2018), we do not explicitly condition the value on the noise variables, but only on the random seed. In practice, the value functions for different seeds can share parameters, and we empirically found that a shared network with a different head per seed works well.

A natural question is whether the policy trained on the N noise instances generalizes to different random seeds. For finite policy classes of size H, standard statistical learning theory results provide PAC bounds for the generalization gap that scale as $O(\log(H)/\sqrt{N})$ (Shalev-Shwartz & Ben-David, 2014). For neural network policies, results based on implicit bias can be derived (Soudry et al., 2018); we leave this investigation to future work. Empirically, we found that policies generalized well with a moderate number of $N \approx 20$ seeds.

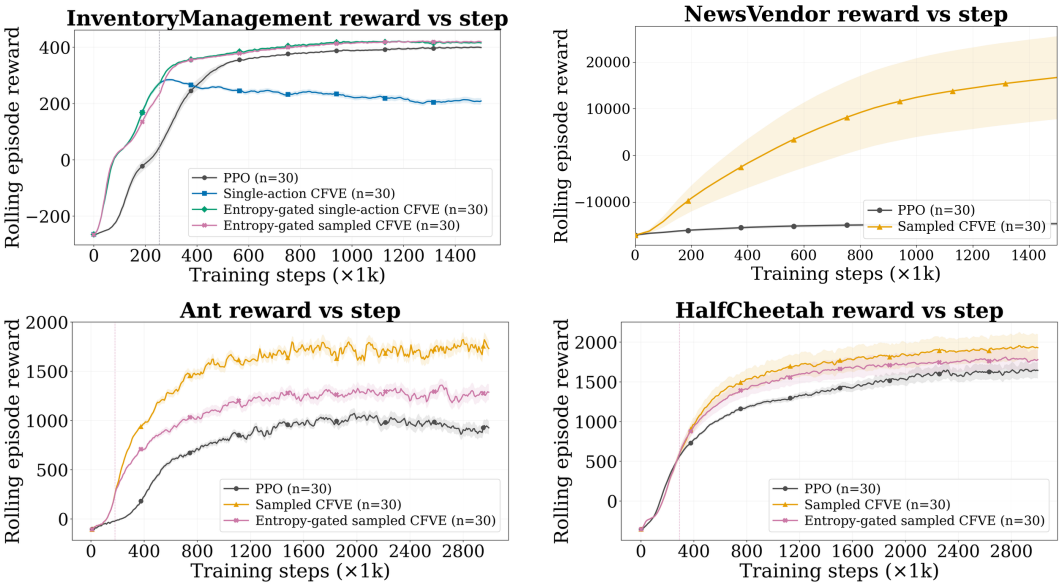


Figure 2: Results: Reward vs. steps on OR-Gym and Mujoco.

5 Experiments

Our experiments are designed to answer the following questions:

1. Does CFVE yield practical performance gains on off-the-shelf PPO?
2. What is the tradeoff between the statistical benefit of CFVE and its computational cost?
3. Do seed-conditioned baselines lead to practical performance gains?

Code for reproducing our results will be made available. Implementation details are in Appendix A.3.

Benchmarks: Based on our theory, we expect CFVE to be most relevant in environments where stochastic transitions are a dominant factor for decision making. Operations-research domains involve resource-allocation and planning problems where stochasticity plays a central role in decision quality; here we focus on Inventory-Management and Newsvendor environments from OR-Gym (Hubbs et al., 2020). Additionally, we include experiments on the popular Mujoco continuous control benchmark (Brockman et al., 2016). While the benchmark domains are deterministic, it is common to add simulation noise for training robust policies that transfer from simulation to real robots (Hwangbo et al., 2019). Here we add i.i.d. Gaussian noise to the state transitions at each time step. Additionally, we experiment on environments from pufferlib, a very fast simulator where we expect neural network optimization to dominate the training time (Suarez, 2024)

The implementation, baseline, evaluated algorithms, and evaluation metrics are described in the Appendix A.3.

In Figures 2 and 3 we show results on 2 OR-Gym domains and 2 Mujoco domains. Figure 2 shows reward vs. optimizer steps, meaning that the extra simulator steps are not counted. We show these results to validate the variance reduction our theory predicts. Figure 3 shows reward vs. wall clock time, taking into account both the simulator and policy optimization efforts. entropy-gated CFVE trains significantly faster than baseline PPO, and also converges to a better solution. The advantage is most acute in the highly stochastic Newsvendor domain. Without entropy-gating, CFVE also outperforms the baseline in 3 domains. The dashed vertical line in the plot marks the step when entropy-gating switched off CFVE; the decline in inventory management after the policy becomes almost deterministic demonstrates the importance of entropy-gating in discrete action domains. Taken together, these results demonstrate that the reduced variance of CFVE translates to significant gains in learning efficiency on these domains. In terms of reward vs. wall-clock time, the higher computational cost of CFVE is evident, and for short training budgets, the baseline PPO may be preferred. However, practitioners ultimately care about the final policy (after reward plateau) and we observe that in

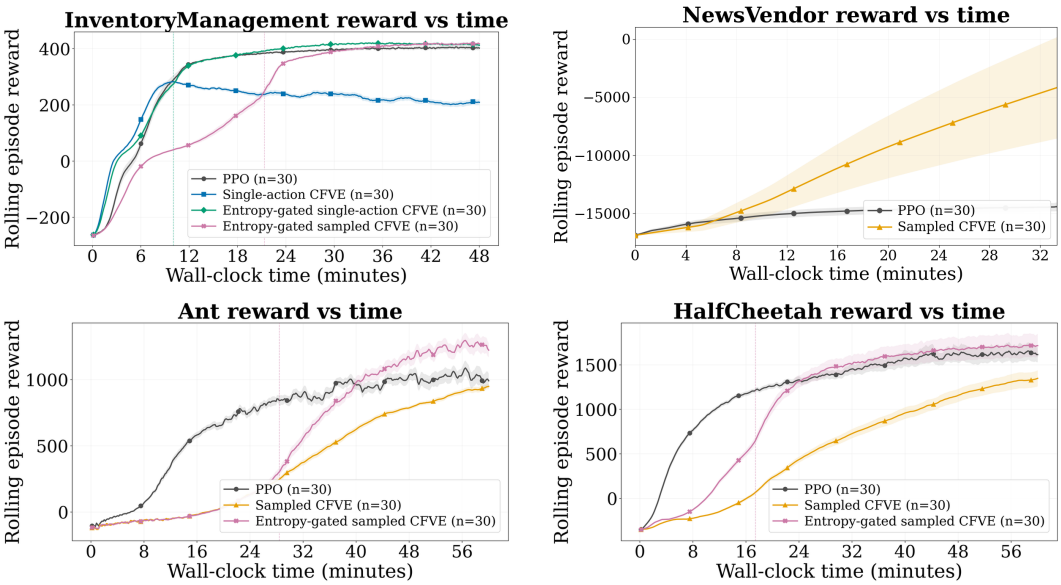
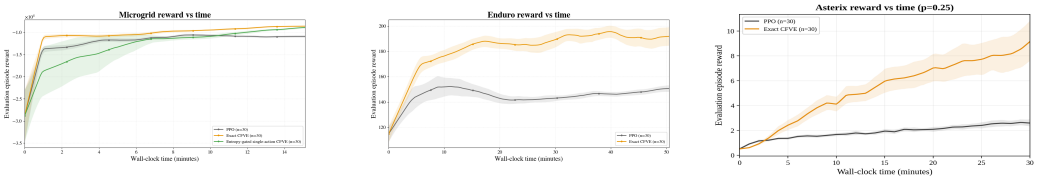


Figure 3: Results: Reward vs. wall-clock time on OR-Gym and Mujoco.



(a) Microgrid. (b) PufferLib Enduro. (c) MinAtar Asterix.

Figure 4: Equal-wall-clock reward on Microgrid, PufferLib Enduro, and MinAtar Asterix ($n = 30$ paired seeds).

all domains, CFVE ultimately leads to a better policy. The margin depends on the domain, and is statistically significant (Holm-corrected (p<0.05) in all of them).

In Appendix B we present results on two additional Mujoco domains: Reacher and Humanoid. Reacher results are similar to the Mujoco results described above – CFVE displays faster convergence in terms of steps. On Humanoid, on the other hand, we observed that CFVE underperformed the baseline PPO. We attribute this to the significantly larger action space in Humanoid, for which the action sampling is less effective. We verify that upon expanding the number of samples CFVE outperforms baseline PPO.

We additionally evaluate CFVE on Microgrid Asterix and PufferLib Enduro using 30 paired seeds in each domain. Figure 4 shows reward against training wall-clock time, including the additional cost of counterfactual evaluations. The graph show a clear improvement both in performance per time steps and per wallclock time, and indicate that our approach works significantly better than baseline ppo, even under equivalent wallclock. Interesting to note that pufferlib's case improvement was especially substantial due to the massive difference in simulator overhead. While in other environments simulation overhead accounted for as much as 75% of wallclock, in Pufferlib's Enduro, due to the nature of the fast simulator, it accounts for as little as approximately 3% of the wallclock overhead, which explains the massive advantage under equal wallclock in this case, as the simulator overhead is negligable.

Effect of Transition Noise On Wallclock Improvement We go on and examine how the amount of noise affects our results, we investigate our approach under different noise regimes in the Ant domain. We examine our approach under increasing noise regimes, and compare wallclock performance against standard ppo, which we can see in figures 5 and 9. The results show that as we introduce more noise into the transition dynamics, CFVE provides an increasingly substantial advantage under equal

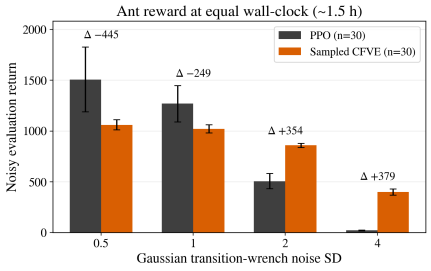


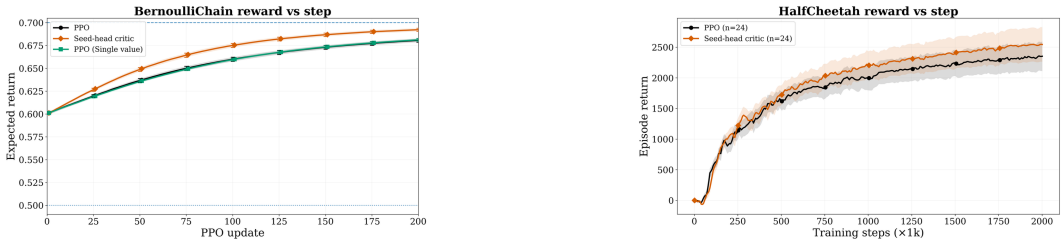
Figure 5: Ant equal-wall-clock return under increasing Gaussian transition noise ($n = 30$ paired seeds).

Table 1: Seed-conditioned critic on held-out MuJoCo evaluation seeds ($n = 40$ per environment).

Environment	PPO	Seed-head critic	p-value
Ant	975	1010	< 0.01
Hopper	2420	2483	< 0.01

wallclock budget (In our case, 90 minutes). We also show that this is robust to noise type (tested against gaussian and Laplacian noise)

Results for Seed-Conditioned Baselines For Seed-Conditioned Baselines, we show illustrative results on a simple chain domain, and results on Mujoco Half-Cheetah with noise as described above. In the chain domain, described in detail in Appendix A, only rewards are stochastic. Figure 6 shows that in both domains our method improves the baseline PPO. To further demonstrate the value of the seed conditioning, in the chain domain we show results for a method that trains only on the N sampled seeds, but uses a regular value function baseline. We additionally evaluate the seed-conditioned critic on Ant and Hopper using held-out evaluation seeds, under the same experimental methodology and structure. Table 1 shows the comparison with PPO over 40 seeds per environment.



(a) Chain domain

(b) HalfCheetah

Figure 6: Seed-conditioned baseline results. Policy evaluation is on different seeds from the training.

6 Conclusion

We studied common random numbers as a policy-gradient variance-reduction method in stochastic simulators, and proposed two simple modifications to off-the-shelf PPO implementation that build on the ability to reset the simulator's RNG. Our promising results suggest that these modifications generally improve performance on stochastic domains.

Limitations: (1) our approach requires a simulator and the ability to reset its RNG. In principle this is a minimal requirement, but for some proprietary simulators this may not be possible; (2) CFVE requires additional simulator steps. As we investigated in our experiments, this may reflect in a slower wall-clock time for some scenarios; (3) Our theoretical results show that CFVE is guaranteed to have lower gradient variance only under certain conditions. Future work may relax some of these conditions; (4) our results on Humanoid suggest that CFVE is less effective for domains with very high action spaces, and we leave the specialization of CFVE to such domains to future work.

AI USE STATEMENT

We used AI to assist with Manuscript writing, development of experimental and plotting code and formatting. The authors take full responsibility for the final manuscript.

REFERENCES

Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Alex Church et al. Tactile sim-to-real policy transfer via real-to-sim image translation. In *Proceedings of the 5th Conference on Robot Learning*, Proceedings of Machine Learning Research, 2022.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep policy gradients: A case study on ppo and trpo. *arXiv preprint arXiv:2005.12729*, 2020.

Paul Glasserman. *Monte Carlo Methods in Financial Engineering*. Springer, 2004.

Evan Greensmith, Peter L. Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in rl. *Journal of Machine Learning Research*, 5:1471–1530, 2004.

Arthur Guez, Fabio Viola, Théophane Weber, Lars Buesing, Steven Kapturowski, Doina Precup, David Silver, and Nicolas Heess. Value-driven hindsight modelling. *Advances in Neural Information Processing Systems*, 33:12499–12509, 2020.

David Ha. Slime volleyball gym environment. `https://github.com/hardmaru/slimevolleygym`, 2020. A Gym environment for benchmarking single-agent and multi-agent reinforcement learning algorithms.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. Pmlr, 2018.

Anna Harutyunyan, Will Dabney, Thomas Mesnard, Mohammad Gheshlaghi Azar, Bilal Piot, Nicolas Heess, Hado van Hasselt, Greg Wayne, Satinder Singh, Doina Precup, and Rémi Munos. Hindsight credit assignment, 2019.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Christian D. Hubbs, Hector D. Perez, Owais Sarwar, Nikolaos V. Sahinidis, Ignacio E. Grossmann, and John M. Wassick. OR-Gym: A reinforcement learning library for operations research problems, 2020.

Jemin Hwangbo, Joonho Lee, Alexey Dosovitskiy, Dario Bellicoso, Vassilios Tsounis, Vladlen Koltun, and Marco Hutter. Learning agile and dynamic motor skills for legged robots. *Science Robotics*, 4(26):eaau5872, 2019. doi: 10.1126/scirobotics.aau5872.

Pierre L'Ecuyer. Efficiency improvement and variance reduction. In *Proceedings of the Winter Simulation Conference*, pp. 122–132, 1994.

Hao Liu, Yihao Feng, Yiwei Mao, Dengyong Zhou, Jian Peng, and Qiang Liu. Action-dependent control variates for policy optimization via stein's identity. In *International Conference on Learning Representations*, 2018.

Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017.

Shie Mannor, Yishay Mansour, and Aviv Tamar. *Reinforcement Learning: Foundations*. Cambridge University Press, 2026. URL https://sites.google.com/view/rlfoundations/home.

Hongzi Mao, Shaileshh Bojja Venkatakrishnan, Malte Schwarzkopf, and Mohammad Alizadeh. Variance reduction for reinforcement learning in input-driven environments. *arXiv preprint arXiv:1807.02264*, 2018.

Thomas Mesnard, Theophane Weber, Fabio Viola, Shantanu Thakoor, Alaa Saade, Anna Harutyunyan, Will Dabney, Thomas S Stepleton, Nicolas Heess, Arthur Guez, et al. Counterfactual credit assignment in model-free reinforcement learning. In *International Conference on Machine Learning*, pp. 7654–7664. PMLR, 2021.

Andrew Y Ng and Michael I Jordan. Pegasus: A policy search method for large mdps and pomdps. *Uncertainty in Artificial Intelligence Proceedings*, 2000.

Chris Nota, Philip Thomas, and Bruno C Da Silva. Posterior value functions: Hindsight baselines for policy gradient methods. In *International Conference on Machine Learning*, pp. 8238–8247. PMLR, 2021.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Lerrel Pinto, Marcin Andrychowicz, Peter Welinder, Wojciech Zaremba, and Pieter Abbeel. Asymmetric actor critic for image-based robot learning. *arXiv preprint arXiv:1710.06542*, 2017.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

Alexander Rutherford, Benjamin Ellis, Matteo Gallici, Jonathan Cook, Andrei Lupu, Garðar Ingvarsson, Timon Willi, Ravi Hammond, Akbir Khan, Christian S de Witt, et al. Jaxmarl: Multi-agent rl environments and algorithms in jax. *Advances in Neural Information Processing Systems*, 37: 50925–50951, 2024.

John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations*, 2016.

John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust region policy optimization, 2017a. URL https://arxiv.org/abs/1502.05477.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017b.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczyński. *Lectures on Stochastic Programming*. SIAM, 3 edition, 2021.

Sean R Sinclair, Felipe Vieira Frujeri, Ching-An Cheng, Luke Marshall, Hugo De Oliveira Barbalho, Jingling Li, Jennifer Neville, Ishai Menache, and Adith Swaminathan. Hindsight learning for mdps with exogenous inputs. In *International Conference on Machine Learning*, pp. 31877–31914. PMLR, 2023.

Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *Journal of Machine Learning Research*, 19(70):1–57, 2018.

Joseph Suarez. Pufferlib: Making reinforcement learning libraries and environments play nice. *arXiv preprint arXiv:2406.12905*, 2024.

Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. *Robotics: Science and Systems XIV*, 2018.

Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012.

Lex Weaver and Nigel Tao. The optimal reward baseline for gradient-based rl. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pp. 538–545, 2001.

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

A Algorithms and Implementation Details

A.1 Algorithms

Algorithm 1 Exact CFVE: PPO with an enumerated counterfactual value baseline

Require: Policy π_θ, critic V_ϕ, finite action set $\mathcal{A}$, discount γ

for each rollout step t **do**
  Store the simulator state and RNG state (x_t, ρ_t).
  for each action $a \in \mathcal{A}$ **do**
    Step the simulator with action a, obtaining: s^a_{t+1}, r^a_t, D^a_t.
    Compute the bootstrapped counterfactual value: $q^a \leftarrow r^a_t + \gamma(1 - D^a_t)V_\phi(s^a_{t+1})$.
    Reset simulator and RNG to (x_t, ρ_t).
  end for
  Sample factual action $a_t \sim \pi_\theta(\cdot \mid s_t)$.
  Step the simulator with a_t, obtaining s_{t+1}, r_t, D_t.
  Compute temporal difference

$$\delta^{\mathrm{CFVE}}_t \leftarrow r_t + \gamma(1 - D_t)V_\phi(s_{t+1}) - \sum_{a \in \mathcal{A}} \pi_\theta(a \mid s_t) q^a.$$

  Compute GAE estimate using (8).
end for
Update θ using the standard PPO clipped objective.

Algorithm 2 Sampled CFVE: PPO with a sampled counterfactual value baseline

Require: Policy π_θ, critic V_ϕ, number of counterfactual actions M, discount γ

for each rollout step t **do**
  Store the simulator state and RNG state (x_t, ρ_t).
  Sample M distinct counterfactual actions

$$\widetilde{a}^1_t, \ldots, \widetilde{a}^M_t \sim \pi_\theta(\cdot \mid s_t).$$

  for $m = 1, \ldots, M$ **do**
    Step the simulator with action $\widetilde{a}^m_t$, obtaining

$$s^m_{t+1}, r^m_t, D^m_t.$$

    Compute the bootstrapped counterfactual value

$$q^m \leftarrow r^m_t + \gamma(1 - D^m_t)V_\phi(s^m_{t+1}).$$

    Reset simulator and RNG to (x_t, ρ_t).
  end for
  Sample factual action $a_t \sim \pi_\theta(\cdot \mid s_t)$.
  Step the simulator with a_t, obtaining s_{t+1}, r_t, D_t.
  Compute temporal difference

$$\delta^{\mathrm{SCFVE}}_t \leftarrow r_t + \gamma(1 - D_t)V_\phi(s_{t+1}) - \frac{1}{M}\sum_{m=1}^{M} q^m.$$

  Compute GAE estimate using (8).
end for
Update θ using the standard PPO clipped objective.

Algorithm 3 Entropy-gated sampled CFVE-PPO

Require: Policy π_θ, critic V_ϕ, number of counterfactual actions M, discount γ, entropy threshold η

use_cfve $\leftarrow$ true
for each rollout step t **do**
  if use_cfve **then**
    Store the simulator state and RNG state (x_t, ρ_t).
    Sample M counterfactual actions

$$\widetilde{a}_t^1, \ldots, \widetilde{a}_t^M \sim \pi_\theta(\cdot \mid s_t).$$

    for $m = 1, \ldots, M$ **do**
      Step the simulator with action $\widetilde{a}_t^m$, obtaining

$$s_{t+1}^m, r_t^m, D_t^m.$$

      Compute the bootstrapped counterfactual value

$$q^m \leftarrow r_t^m + \gamma(1 - D_t^m)V_\phi(s_{t+1}^m).$$

      Reset simulator and RNG to (x_t, ρ_t).
    end for
    Sample factual action $a_t \sim \pi_\theta(\cdot \mid s_t)$.
    Step the simulator with a_t, obtaining s_{t+1}, r_t, D_t.
    Compute temporal difference

$$\delta_t \leftarrow r_t + \gamma(1 - D_t)V_\phi(s_{t+1}) - \frac{1}{M}\sum_{m=1}^{M} q^m.$$

  else
    Sample action $a_t \sim \pi_\theta(\cdot \mid s_t)$.
    Step the simulator with a_t, obtaining s_{t+1}, r_t, D_t.
    Compute standard PPO temporal difference

$$\delta_t \leftarrow r_t + \gamma(1 - D_t)V_\phi(s_{t+1}) - V_\phi(s_t).$$

  end if
  Compute GAE estimate from the residuals $\{\delta_\ell\}_{\ell \geq t}$.
  If the normalized policy entropy is below the threshold,

$$\frac{\mathcal{H}(\pi_\theta(\cdot \mid s_t))}{\mathcal{H}_0} \leq \eta,$$

  set use_cfve $\leftarrow$ false.
end for
Update θ using the standard PPO clipped objective.

A.2 Implementation details

A.3 Experimental Setup and Evaluation Protocol

Implementation: We aim to show that CFVE improves *off-the-shelf* RL algorithms, and focus on stable-baselines-3's PPO (Raffin et al., 2021), a tried-and-true implementation that has been used in applications ranging from robotics to multi-agent games (Church et al., 2022; Ha, 2020). Hyperparamters and implementation details are known to be important in PPO (Henderson et al., 2018; Engstrom et al., 2020); we used the default stable-baselines-3 hyperparameters throughout our experiments.

Baseline: We compare with the default PPO algorithm. PPO is not uniformly state-of-the-art across all domains, and other algorithms can outperform it in settings for which they are better suited, e.g., soft actor critic (Haarnoja et al., 2018). Nevertheless, PPO remains a widely used general-purpose policy-gradient method. Since CFVE is designed as a minimal modification of PPO, our primary goal is to isolate the effect of this modification and test whether it can reliably improve the default PPO baseline. The methods in (Mao et al., 2018; Nota et al., 2021; Mesnard et al., 2021) are not compatible with GAE and cannot be easily incorporated into PPO.

Algorithms: We evaluate the entropy-gated CFVE sampled method of Section 4.1 in all domains. Hyperparameters are detailed in Appendix A.3. For the discrete-action OR-Gym domains, we also evaluated the exact CFVE, but found that it yielded approximately similar performance to the much faster single-action CFVE variant, and therefore we report results only with the latter.

Metrics: We evaluate learning curves averaged over 30 training runs with random initial weights (the same randoms seeds were used across all algorithms). The standard learning curve reports **reward vs. training steps**, where a single step means the simulator step for the *factual* action, and relates to the *statistical* accuracy of the policy gradient estimator. We expect that a lower-variance policy gradient estimator will need less steps to converge. However, since CFVE steps the simulator also for *counterfactual actions*, we also report **reward vs. wall-clock time**, relating to the computational complexity of the method. We emphasize that our wall-clock time results are *implementation specific*: they depend on the simulator speed and neural network size/training algorithm, and may look completely different with faster simulators (Rutherford et al., 2024; Suarez, 2024). Nevertheless, we include them to demonstrate performance tradeoffs.

A.3.1 PPO HYPERPARAMETERS

We use the default Stable-Baselines3 (Raffin et al., 2021) hyperparameters for PPO and for all PPO derivative methods considered and shown in the paper. Specifically, all methods use the same PPO backbone, policy architecture, optimizer settings, rollout length, minibatch construction, clipping parameters, discounting, generalized advantage estimation parameters and so on and so forth. We do not tune PPO hyperparameters separately across methods, or at all, for that matter.

Table 2: PPO hyperparameters used in all experiments. Unless otherwise stated, these are the default Stable-Baselines3 PPO hyperparameters.

Hyperparameter	**Value**
Policy class	`MlpPolicy`
Policy network	default SB3 actor–critic MLP
Activation function	`tanh`
Optimizer	Adam
Learning rate	3×10^{-4}
Rollout length n_{steps}	2048
Minibatch size	64
Number of PPO epochs	10
Discount factor γ	0.99
GAE parameter λ	0.95
Clipping parameter ϵ	0.2
Value-function clipping	none
Advantage normalization	enabled
Entropy coefficient	0.0
Value-loss coefficient	0.5
Maximum gradient norm	0.5
Target KL	none
State-dependent exploration	disabled

A.3.2 MUJOCO INTRODUCED NOISE

For the Mujoco (Todorov et al., 2012) experiments, in the CFVE, we apply gaussian noise with 0 mean and variance of 1. In the Seedoverfit experiments, we apply gaussian noise with 0 mean and variance of 0.5. In the Or-Gym (Hubbs et al., 2020) environments we use their default environments.

A.3.3 CFVE IMPLEMENTATION DETAILS

For the sampled Counterfactual Value Functionapproach, we fix the hyperparameter of sample size to be 8, as we found it most empirically successful in the current paradigm's compute to performance tradeoff.

For the Entropy-Gated Counterfactual Value Function, We end up selecting a relative entropy coefficient of 0.15, which proves to be empirically effective and outperforms the majority of the hyperparameters we've tested.

A.3.4 SEEDOVERFIT IMPLEMENTATION DETAILS

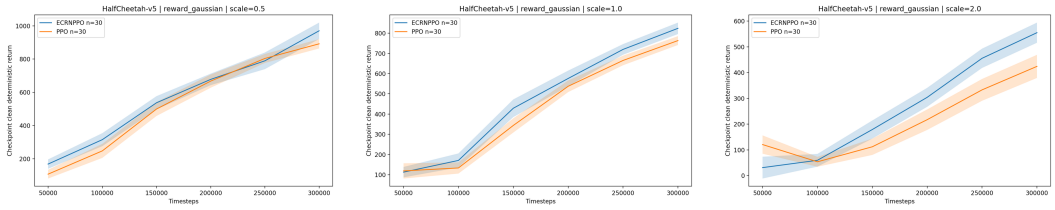
For the Seedoverfit case, generalization was empirically found to occur around 20 seeds given the noise paradigm we have injected into the Mujoco environments

A.4 BERNOULLI CHAIN ENVIRONMENT

We use a small Bernoulli chain environment as a controlled testbed for seed-conditioned value baselines. The environment is a one-dimensional chain. Each episode starts at the center of the chain. At every nonterminal state, the agent chooses one of two actions: move left or move right. Transitions are deterministic according to the action. The episode ends when the agent reaches either the left terminal state or the right terminal state. All intermediate rewards are zero.

The reward incurred at the end of the chain is either Bernoulli reward with success probability of 0.5 at the left-most-state, and 0.7 at the right most state. This is the source of stochasticity present.

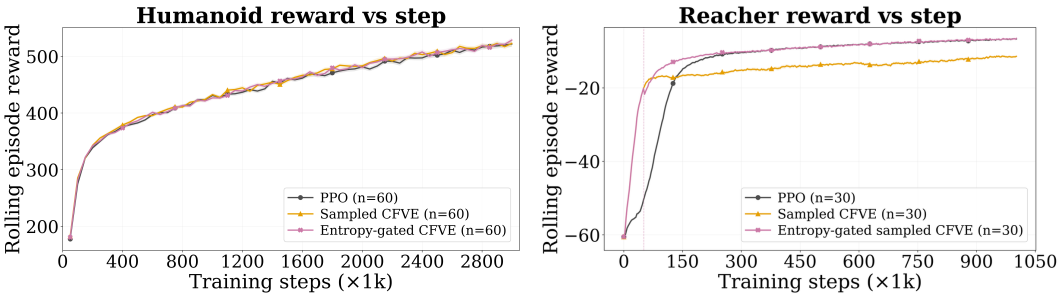
The experiment is conducted on a 3-length chain (meaning 7 total states, 3 length to the right and 3 to the left). We train on a set of pre-determined seeds that we condition on, and evaluate on a set of heldout-seeds, never seen before by the model, as to make sure that there is no way the model memorized the environment.



(a) Scale 0.5 (b) Scale 1.0 (c) Scale 2.0

Figure 8: HalfCheetah with Gaussian reward noise. Sampled CFVE improves upon ppo as the noise applied increases, with a larger improvement as the noise level increases.

B Additional MuJoCo Result Graphs



(a) Humanoid (b) Reacher

Figure 7: Additional MuJoCo learning-curve results in Humanoid and Reacher, measured as reward versus environment steps.

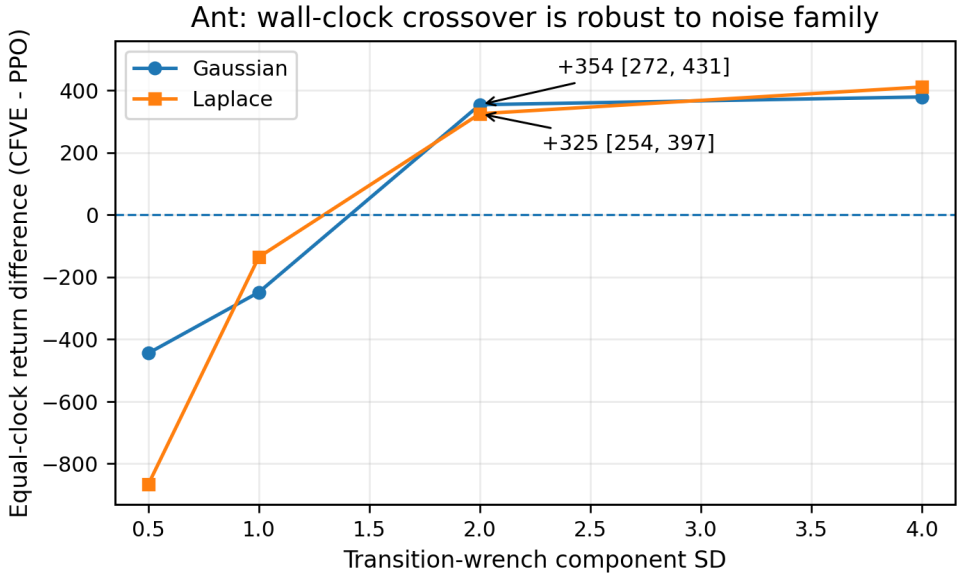


Figure 9: Ant equal wall clock return delta under Gaussian and Laplace transition noise. Positive values favor sampled CFVE over standard PPO.

C Proof of Proposition 5

Proof. From the law of conditional variance, $\mathrm{Var}(g_{t;b}) = \mathbb{E}[\mathrm{Var}(g_{t;b} \mid s_t, \xi_t)] + \mathrm{Var}[\mathbb{E}(g_{t;b} \mid s_t, \xi_t)]$, and from Theorem 2, $\mathbb{E}(g_{t;b} \mid s_t, \xi_t)$ does not depend on b. Fix (s, ξ). By Theorem 2, for any baseline $b(s, \xi)$,

$$\begin{aligned}\mathrm{Var}(g_{t;b} \mid s, \xi) &= \mathbb{E}\big[w_t^2(Q-b)^2 \mid s, \xi\big] + (\mathbb{E}[w_t(Q-b) \mid s, \xi])^2 \\ &= \mathbb{E}\big[w_t^2(Q-b)^2 \mid s, \xi\big] + (\mathbb{E}[w_t(Q) \mid s, \xi])^2\end{aligned}$$

Hence variance minimization reduces to minimizing the second moment $\mathbb{E}\big[w_t^2(Q-b)^2 \mid s, \xi\big]$. Write

$$Q - b = (Q - b^*) + (b^* - b),$$

where

$$b^*(s, \xi) = \frac{\mathbb{E}[w_t^2 Q \mid s, \xi]}{\mathbb{E}[w_t^2 \mid s, \xi]}.$$

Then

$$\begin{aligned}\mathbb{E}[w_t^2(Q-b)^2 \mid s, \xi] = \mathbb{E}[w_t^2(Q-b^*)^2 \mid s, \xi] \\ + 2(b^* - b)\mathbb{E}[w_t^2(Q-b^*) \mid s, \xi] \\ + (b^* - b)^2 \mathbb{E}[w_t^2 \mid s, \xi].\end{aligned}$$

The middle term vanishes by definition of b^*, yielding

$$\mathrm{Var}(g_{t;b}) = \mathrm{Var}(g_{t;b^*}) + \mathbb{E}\Big[\sigma(s_t, \xi_t)\big(b(s_t, \xi_t) - b^*(s_t, \xi_t)\big)^2\Big].$$

□

D Proof of Corollary 6

If

$$\mathrm{Cov}(Q(s, a, \xi), w_{t,j}^2 \mid s, \xi) = 0,$$

then

$$b_j^*(s, \xi) = \frac{\mathbb{E}[w_{t,j}^2 Q \mid s, \xi]}{\mathbb{E}[w_{t,j}^2 \mid s, \xi]} = \mathbb{E}[Q \mid s, \xi] = V^\xi(s).$$

Substituting $b = V^\pi$ and $b = V^\xi$ into Proposition 5 gives

$$\mathrm{Var}(g_{j,V^\pi}) - \mathrm{Var}(g_{j,V^\xi}) = \mathbb{E}\Big[\omega_j(s_t, \xi_t)\big(V^\pi(s_t) - V^{\xi_t}(s_t)\big)^2\Big] \geq 0.$$

□

E Proof of Theorem 4

Theorem 8 (Noise-conditioned baselines reduce advantage variance)**.** *Let*

$$Z := Q(s, a, \xi),$$

For notational convenience, let us denote

$$\mathcal{A}(s, A, \xi) := Q^\pi(s, A, \xi) - V^\pi(s), \qquad \mathcal{A}_{\mathrm{CFVE}}(s, A, \xi) := Q^\pi(s, A, \xi) - V^\pi(s, \xi).$$

(Where $\mathcal{A}$ is the standard advatage) Then

$$\mathrm{Var}_{A\sim\pi_\theta(\cdot|s),\, \xi\sim P_\xi}(\mathcal{A}_{\mathrm{std}}(s, A, \xi)) = \mathrm{Var}_{A\sim\pi_\theta(\cdot|s),\, \xi\sim P_\xi}(\mathcal{A}_{\mathrm{CFVE}}(s, A, \xi)) + \mathrm{Var}_{\xi\sim P_\xi}(V^\pi(s, \xi)).$$

Therefore,

$$\mathrm{Var}_{A\sim\pi_\theta(\cdot|s),\, \xi\sim P_\xi}(\mathcal{A}_{\mathrm{CFVE}}(s, A, \xi)) \leq \mathrm{Var}_{A\sim\pi_\theta(\cdot|s),\, \xi\sim P_\xi}(\mathcal{A}_{\mathrm{std}}(s, A, \xi)),$$

Where in non-determinstic systems we will see a strict variance reduction.

Proof. We fix $s \in \mathcal{S}$ throughout the proof and suppress the dependence on s whenever this does not create ambiguity for convenience and clarity of the proof itself.

By definition of the standard value function,

$$V^\pi(s) = \mathbb{E}_{A\sim\pi_\theta(\cdot|s),\,\xi\sim P_\xi}[Q^\pi(s,A,\xi)] = \mathbb{E}_{A\sim\pi_\theta(\cdot|s),\,\xi\sim P_\xi}[Z].$$

Therefore the standard centered advantage can be written as

$$\mathcal{A}(s,A,\xi) = Z - \mathbb{E}_{A\sim\pi_\theta(\cdot|s),\,\xi\sim P_\xi}[Z].$$

Since subtracting a deterministic constant does not change variance,

$$\mathrm{Var}_{A\sim\pi_\theta(\cdot|s),\,\xi\sim P_\xi}(\mathcal{A}_{\mathrm{std}}(s,A,\xi)) = \mathrm{Var}_{A\sim\pi_\theta(\cdot|s),\,\xi\sim P_\xi}(Z). \tag{10}$$

Now consider the noise-conditioned value. By definition,

$$V^\pi(s,\xi) = \mathbb{E}_{A\sim\pi_\theta(\cdot|s)}[Q^\pi(s,A,\xi) \mid \xi] = \mathbb{E}[Z \mid \xi],$$

where the conditional expectation $\mathbb{E}[Z \mid \xi]$ is with respect to the sampled action A. Hence the CFVE advantage is

$$\mathcal{A}_{\mathrm{CFVE}}(s,A,\xi) = Z - \mathbb{E}[Z \mid \xi].$$

Conditioned on ξ, this quantity has mean zero:

$$\mathbb{E}[\mathcal{A}_{\mathrm{CFVE}}(s,A,\xi) \mid \xi] = \mathbb{E}[Z \mid \xi] - \mathbb{E}[Z \mid \xi] = 0. \tag{11}$$

Thus, applying the law of total variance to the random variable Z with respect to the conditioning variable ξ gives

$$\begin{aligned}\mathrm{Var}_{A\sim\pi_\theta(\cdot|s),\,\xi\sim P_\xi}(Z) &= \mathbb{E}_{\xi\sim P_\xi}\big[\mathrm{Var}_{A\sim\pi_\theta(\cdot|s)}(Z \mid \xi)\big] + \mathrm{Var}_{\xi\sim P_\xi}\big(\mathbb{E}_{A\sim\pi_\theta(\cdot|s)}[Z \mid \xi]\big)\\ &= \mathbb{E}_{\xi\sim P_\xi}\big[\mathrm{Var}_{A\sim\pi_\theta(\cdot|s)}(Q^\pi(s,A,\xi) \mid \xi)\big] + \mathrm{Var}_{\xi\sim P_\xi}(V^\pi(s,\xi))\,.\end{aligned}$$

On the other hand, by the definition of $\mathcal{A}_{\mathrm{CFVE}}(s,A,\xi)$ and the conditional mean-zero property in (11),

$$\begin{aligned}\mathrm{Var}_{A\sim\pi_\theta(\cdot|s),\,\xi\sim P_\xi}(\mathcal{A}_{\mathrm{CFVE}}(s,A,\xi)) &= \mathrm{Var}_{A\sim\pi_\theta(\cdot|s),\,\xi\sim P_\xi}(Z - \mathbb{E}[Z \mid \xi])\\ &= \mathbb{E}_{\xi\sim P_\xi}\big[\mathrm{Var}_{A\sim\pi_\theta(\cdot|s)}(Z \mid \xi)\big]\,.\end{aligned}$$

Combining the last two equations yields

$$\mathrm{Var}_{A\sim\pi_\theta(\cdot|s),\,\xi\sim P_\xi}(Z) = \mathrm{Var}_{A\sim\pi_\theta(\cdot|s),\,\xi\sim P_\xi}(\mathcal{A}_{\mathrm{CFVE}}(s,A,\xi)) + \mathrm{Var}_{\xi\sim P_\xi}(V^\pi(s,\xi))\,.$$

Plugging (10) into the previous identity, we obtain

$$\mathrm{Var}_{A\sim\pi_\theta(\cdot|s),\,\xi\sim P_\xi}(\mathcal{A}_{\mathrm{std}}(s,A,\xi)) = \mathrm{Var}_{A\sim\pi_\theta(\cdot|s),\,\xi\sim P_\xi}(\mathcal{A}_{\mathrm{CFVE}}(s,A,\xi)) + \mathrm{Var}_{\xi\sim P_\xi}(V^\pi(s,\xi))\,.$$

Finally, the inequality follows because

$$\mathrm{Var}_{\xi\sim P_\xi}(V^\pi(s,\xi)) \geq 0,$$

and it is strict whenever this variance is non zero.. □

F Proof of Proposition 7

Proof. Let us consider the second moment $\mathbb{E}\big[w_t^2(Q^{\pi_\theta}(s,a,\xi) - b)^2 \mid s,\xi\big]$. We have:

$$\begin{aligned}\mathbb{E}\big[w_t^2(Q^{\pi_\theta}(s,a,\xi) - V^{\pi_\theta}(s,\xi))^2 \mid s,\xi\big] &= \mathbb{E}\left[w_t^2(h(s,a) + \phi(\xi,s) - \sum_{a'}\pi_\theta(a')h(s,a') - \phi(\xi,s))^2 \mid s,\xi\right]\\ &= \mathbb{E}\left[w_t^2(h(s,a) - \sum_{a'}\pi_\theta(a')h(s,a'))^2 \mid s,\xi\right]\end{aligned}$$

On the other hand,

$$\begin{aligned}\mathbb{E}\big[w_t^2(Q^{\pi_\theta}(s,a,\xi)-V^{\pi_\theta}(s))^2 \mid s,\xi\big] =&\mathbb{E}\left[w_t^2(h(s,a)+\phi(\xi,s)-\sum_{a'}\pi_\theta(a')h(s,a)-\sum_{\xi'}\phi(\xi',s)P(\xi',s))^2 \mid s,\xi\right]\\ =&\mathbb{E}\left[w_t^2(h(s,a)-\sum_{a}\pi_\theta(a)h(s,a))^2 \mid s,\xi\right]\\ &+\mathbb{E}\left[w_t^2(\phi(\xi,s)-\sum_{\xi'}\phi(\xi',s)P(\xi'))^2 \mid s,\xi\right]\\ &-2\left(\phi(\xi,s)-\sum_{\xi'}\phi(\xi',s)P(\xi')\right)\mathbb{E}\left[w_t^2(h(s,a)-\sum_{a}\pi_\theta(a)h(s,a)) \mid s\right].\end{aligned}$$

Since $\mathbb{E}_\xi\left(\phi(\xi,s)-\sum_{\xi'}\phi(\xi',s)P(\xi')\Big| s\right)=0$, we have that

$$\mathrm{Var}(g_{t;V^{\pi_\theta}(s)})-\mathrm{Var}(g_{t;V^{\pi_\theta}(s,\xi)})=\mathbb{E}\left[w_t^2(\phi(\xi,s)-\sum_{\xi'}\phi(\xi',s)P(\xi'))^2\right]\geq 0.$$

□