Off-Environment RL with Rare Events

Kamil Ciosek **Shimon Whiteson**
Department of Computer Science
University of Oxford
Wolfson Building, Parks Rd, Oxford OX1 3QD
kamil.ciosek,shimon.whiteson@cs.ox.ac.uk

Abstract

Policy gradient methods have been widely applied in reinforcement learning. For reasons of safety and cost, learning is often conducted using a simulator. However, learning in simulation does not traditionally utilise the opportunity to improve learning by adjusting certain *environment variables* – state features that are randomly determined by the environment in a physical setting but controllable in a simulator. Exploiting environment variables is crucial in domains containing *significant rare events* (SREs), e.g., unusual wind conditions that can crash a helicopter, which are rarely observed under random sampling but have a considerable impact on expected return. We propose *off environment reinforcement learning* (OFFER), which addresses such cases by simultaneously optimising the policy and a *proposal distribution* over environment variables. We prove that OFFER converges to a locally optimal policy and show experimentally that it learns better and faster than a policy gradient baseline.

1 Introduction

When applying *reinforcement learning* (RL) to physical systems, a major issue is the cost and risk of running trials, e.g., when learning a control policy for a robot. Hence, learning is often performed using a simulator, to which off-line RL (i.e., sample-based planning) can be applied. Although this is cheaper and safer than physical trials, the computational cost of each trial can still be considerable. It is therefore important to develop algorithms that minimise this cost. *Policy gradient* methods [25] are popular in such settings as they cope well with continuous action spaces, which often occur in physical systems such as robots.

However, existing policy gradient methods do not exploit an important opportunity presented by simulators: the chance to adjust certain *environment variables*, i.e., state features that cannot be controlled in a physical setting but are (stochastically) determined by the environment. For example, if we learn to fly a helicopter under varying wind conditions [15], we cannot control the wind in physical trials but can easily do so in simulation.

A conventional application of policy gradient methods to such settings is not robust to *significant rare events* (SREs), i.e., it fails any time there are rare events that substantially affect expected performance. For example, some rare wind conditions may increase the risk of crashing the helicopter. Since crashes are so catastrophic, avoiding them is key to maximising expected performance, even though the wind conditions contributing to the crash occur only rarely. In such cases, the conventional approach does not see such rare events often enough to learn an appropriate response.

In this paper, we propose a new policy gradient method called *off-environment reinforcement learning* (OFFER) that aims to address this deficiency. The main idea is to couple the *primary optimisation* of the policy with the *secondary optimisation* of a proposal distribution governing the environment variables. Since environment variables can be controlled in simulation, we are free to sample from the proposal distribution when generating data for the primary optimisation. Thanks to importance

The work was presented at the NIPS workshop on Optimizing the Optimizers. Barcelona, Spain, 2016.

sampling, the primary optimisation can retain unbiased gradient estimates. Just as *off-policy* RL learns about a target policy while using a behaviour policy, *off-environment* RL learns about a target environment (the true distribution over environment variables) while generating data from another environment (the proposal distribution). By learning a proposal distribution that minimises the variance in the gradient estimate used during primary optimisation, OFFER can automatically discover and focus on the SREs that any robust policy must address.

We show that OFFER is guaranteed to converge to a locally optimal policy. In addition, we present empirical results on several tasks showing that it greatly outperforms existing policy gradient methods in the presence of significant rare events.

Our approach is related to existing work on *adaptive importance sampling* [1, 8, 7], which also seeks to optimise proposal distributions. but mostly focus on *Markov reward processes*, not MDPs. It has been attempted [8] to consider a full MDP and aim to learn using a proposal distribution that takes rare events into account; however, the authors assume prior knowledge of what the significant rare events are as well as access to a simulator with environment variables that directly control the probability of such rare events. By contrast, our work seeks to automatically discover significant rare events and the proposal distributions that generate them. One can also [19] consider a similar setting to ours but in the context of Bayesian optimisation, in which case environment variables must be marginalised out using Bayesian quadrature. Here, we consider a policy gradient approach, which is typically more effective in high-dimensional tasks.

2 Problem Setting

We model the decision-making task as a *Markov decision process* (MDP), in which taking an action $a_t \in A$ in state $s_t \in S$ at time t generates a reward whose expected value is $r(s_t, a_t)$ and a transition to a next state $s_{t+1} \sim p(s_{t+1}|s_t, a_t)$. We assume access to an MDP simulator in the form of a *trajectory model* that generates sequences of samples, obtained from the distribution over initial states $p_1(s_1)$ and each action a_t is sampled from a stochastic policy $\pi_\theta(a_t|\phi(s_t))$ parameterised by a vector θ; $\phi(s_t)$ is a function mapping s_t to a vector of real-valued features. In the sequel, we write $\pi_\theta(a_t|s_t)$ for brevity. We assume π_θ is a twice differentiable function of θ.

In addition, we assume access to a vector of *environment variables* ψ (e.g., coefficients of friction, wind velocities) that affect state transition probabilities. Note that, while we can control ψ when running the simulator, the policy we ultimately deploy cannot.

In this paper, we model environment variables by supposing that states in the simulator are sampled, not from $p_1(s_1)$ and $p(s_{t+1}|s_t, a_t)$, but from *proposal distributions* $f_1^\psi(s_1)$ and $f^\psi(s_{t+1}|s_1, a_t)$, which are parameterised by ψ. The goal in this setting is find a θ that maximises the total expected return $J_\theta = \int_S \rho^\pi(s) \int_A \pi_\theta(a|s) r(s,a) da ds$ where the improper distribution $\rho^\pi(s') = \int_S \sum_{n=1}^\infty \gamma^{n-1} p_1(s) p(s \to s', n, \pi) ds$ and $p(s \to s', n, \pi) = \sum_{\tau \in \text{Traj}(n,s,s')} \prod_{t=1}^n \int p(s_{t+1}|s_t, a_t) \pi(a_t|s_t) da_t$, where $\text{Traj}(n, s, s')$ is the set of all possible trajectories of length n beginning with s and ending with s'. In the next section, we propose an algorithm that learns both θ and ψ in parallel.

3 Off-Environment Reinforcement Learning

We propose *off-environment reinforcement learning*, or OFFER (Algorithm 1) for coping with significant rare events by exploiting environment variables. OFFER interleaves two optimisation steps. The *primary optimisation* step performs an importance-weighted policy gradient update, adjusting θ to improve the policy's expected return. The *secondary optimisation* performs a gradient descent step on the proposal distribution, adjusting ψ to reduce the variance of the gradient estimate used during primary optimisation.

Algorithm 1 OFFER()

while not converged **do**
  $\tau \leftarrow$ sample trajectory using π_θ and f_Ψ
  $\triangleright\ \tau = (s_1, a_1, r_1, s_2, a_2, r_2, \ldots, s_N, a_N, r_N)$
  $\Delta\theta \leftarrow$ PRIMARY-OPTIMISATION(τ, θ, ψ)
  $\theta \leftarrow \theta + \Delta\theta$
  $\Delta\psi \leftarrow$ SECONDARY-OPTIMISATION(τ, ψ)
  $\psi \leftarrow \psi + \Delta\psi$
end while

3.1 Primary Optimisation

The primary optimisation performs a policy gradient update that adjusts θ to improve expected return. To estimate the gradient, we apply importance sampling to the policy gradient update [12]:

$$\nabla_\theta J_\theta = \mathrm{E}_{\tau\sim f_\pi(\tau)}\left[\frac{p_\pi(\tau)}{f_\pi(\tau)}\sum_{t=1}^{N}\gamma^t\nabla_\theta\log\pi_\theta(a_t|s_t)Q^\pi(s_t,a_t)\right], \quad \text{where } \begin{cases} f_\pi(\tau) = f_1(s_1)\prod_{t=1}^{N} f(s_{t+1}|s_t,a_t)\pi(a_t|s_t), \\ p(\tau) = p_1(s_1)\prod_{t=1}^{N} p(s_{t+1}|s_t,a_t)\pi(a_t|s_t). \end{cases}$$

We now consider how to update θ. Since the magnitude of the update in stochastic gradient descent depends on the magnitude of $\hat{u}$ and hence on the magnitude of the rewards, the optimal learning rate depends on the scale of the rewards. Furthermore, in the presence of SREs we have to deal with different reward scales (this is what makes the rare event significant). Hence, the ability to adaptively set the learning rate separately for each policy feature becomes an essential, not merely desirable, characteristic of the learning algorithm. To meet this requirement, we employ a stochastic variant of Newton's method [9, 23] because it has already been shown to work well with policy gradient methods [9]. Furthermore, we maintain a mean of the Hessians estimated over time. To avoid the prohibitive cost of inverting the full Hessian, only its diagonal is used, which corresponds to performing Newton's method on each coordinate separately. Algorithm 2 summarises the resulting primary optimisation algorithm, where CRITIC-REINFORCE and CRITIC-AC correspond to the critic part for actor-critic methods for computing a baseline-adjusted approximation to the Q value [20].

Algorithm 2 PRIMARY-OPTIMISATION(τ, θ,ψ)

$\triangleright$ Traj. $\tau = (s_1, a_1, r_1, s_2, a_2, r_2, \ldots, s_N, a_N, r_N)$
$\triangleright$ Critic
$\hat{u} \leftarrow$ CRITIC-REINFORCE(τ) or CRITIC-AC(τ) $\quad\triangleright$ Baseline-adjusted estimate of Q function

$\triangleright$ Actor
$w \leftarrow \frac{p_1(s_1)}{f_1(s_1)}\prod_{t=1}^{N}\frac{p(s_{t+1}|s_t,a_t)}{f(s_{t+1}|s_t,a_t)}$ $\quad\triangleright$ Importance sampling
$\hat{\nabla}_\theta J_\theta \leftarrow w\sum_{i=1}^{N}\gamma^i\nabla_\theta\log\pi_\theta(a_i|s_i)\hat{u}_i$
$\hat{H} \leftarrow w\sum_{i=1}^{N}\gamma^i\mathrm{diag}(\nabla_\theta\nabla_\theta^\top\log\pi_\theta(a_i|s_i))\hat{u}_i$
$\hat{H}_M \leftarrow \frac{i-1}{i}\hat{H}_M + \frac{1}{i}\hat{H}$

return $\alpha_i\hat{H}_M^{-1}\hat{\nabla}_\theta J_\theta$

3.2 Secondary Optimisation

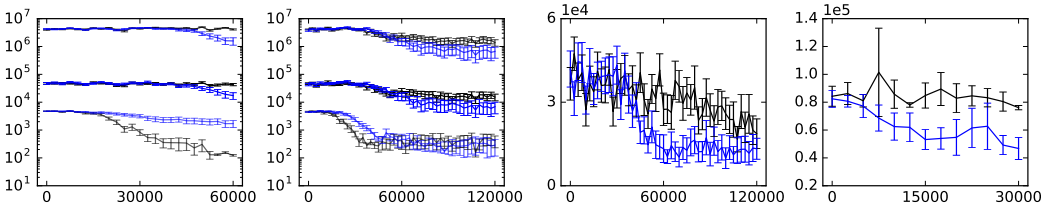
The goal of secondary optimisation is to find a proposal distribution that best facilitates primary optimisation. To this end, we propose to minimise the variance of the gradient estimate followed during primary optimisation. Such variance is known to be a key contributor to slow learning in policy gradient methods [20, 26, 12, 17]. By minimising this variance, we expect OFFER to discover proposal distributions that generate significant rare events more often than in the original task. Since such events contribute substantially to expected return, doing so makes it easier to estimate the gradient of the expected return.

We start by defining the covariance $C = \mathrm{Cov}_{\tau\sim f^\psi}\left[\frac{1}{f^\psi(\tau)}h(\tau)\right]$, where $h(\tau) = p(\tau)\sum_{t=1}^{N}\gamma^t\nabla_\theta\log\pi_\theta(a_t|s_t)u_t$. Since we are interested in minimising our uncertainty about each

Algorithm 3 SECONDARY-OPTIMISATION(τ, ψ)

▷ Traj. $\tau = (s_1, a_1, r_1, s_2, a_2, r_2, \ldots, s_N, a_N, r_N)$
▷ ADAM algorithm
$\hat{\nabla}_{\psi_j} \operatorname{trace}(C) \leftarrow \frac{1}{m} \sum_{\tau \in S} \sum_i -\frac{1}{f^\psi(\tau)^3} \left(\nabla_{\psi_j} f(\tau)\right) h_i(\tau)^2$
$m \leftarrow \beta_1 m + (1 - \beta_1)\hat{\nabla}_{\psi_j} \operatorname{trace}(C)$
▷ $\odot$ is an element-wise product
$v \leftarrow \beta_2 v + (1 - \beta_2)\hat{\nabla}_{\psi_j} \operatorname{trace}(C) \odot \hat{\nabla}_{\psi_j} \operatorname{trace}(C)$
$\hat{m} \leftarrow m/(1 - \beta_1^i)$
$\hat{v} \leftarrow v/(1 - \beta_2^i)$

▷ Division and square root are element-wise
return $\alpha' \hat{m}/\sqrt{\hat{v} + \epsilon}$



(a) Unaliased MC (b) Aliased MC (c) 4-Featured MC (d) Robotic Arm

Figure 1: *Average per-episode penalty (lower is better) for OFFER (blue) and a policy gradient baseline (black) in three variants of mountain car, as well as a robotic arm task.*

of the partial derivatives that make up the gradient, rather than the correlations between them, we consider only the diagonal of C. Furthermore, since we have no a priori reason to think one partial derivative is more important than another, we define the trace of C as our objective, i.e., we solve the optimisation $\min_\psi \operatorname{trace}(C)$. If we treat $\hat{H}_M$ as a constant (which is approximately true for large N), then minimising the covariance of $\hat{\nabla}_\theta J_\theta$ also minimises the covariance of $\Delta\theta$. To solve the minimisation by gradient descent, we must evaluate the gradient of the trace with respect to ϕ. It can be shown that $\nabla_{\psi_j} \operatorname{trace}(C) = \mathrm{E}_{\tau \sim f^\psi}\left[-\sum_i \frac{1}{f^\psi(\tau)^3} \left(\nabla_{\psi_j} f(\tau)\right) h_i(\tau)^2\right]$, which can be estimated from a trajectory τ as follows:

$$\hat{\nabla}_{\psi_j} \operatorname{trace}(C) = \sum_i -\frac{1}{f^\psi(\tau)^3} \left(\nabla_{\psi_j} f(\tau)\right) \hat{h}_i(\tau)^2, \quad \text{where} \quad \hat{h}_i(\tau) = p(\tau) \sum_{t=1}^{N} \gamma^t \nabla_\theta \log \pi_\theta(a_t|s_t) \hat{u}_t.$$

The secondary optimisation can be performed using any variant of stochastic gradient descent (we use ADAM [14]). Algorithm 3 summarises the secondary optimisation.

The following corollary establishes a convergence guarantee for OFFER.

Corollary 1. *Under mild technical conditions, OFFER (Algorithm 1) converges almost surely to the local minimum $\theta^\star$. Moreover, ψ converges to a local minimum. The proof, which we omit due to lack of space, extends a result established by [23].*

4 Experiments

We empirically compare OFFER to a policy gradient baseline (primary optimisation only) on variants of the mountain car task, as well as a simulated robot arm. Both domains have penalties rather than rewards, so lower is better on all plots. All results are averaged over 48 runs. Blue curves represent our algorithm and black curves represent vanilla policy gradients. The lines at the top and bottom of the upper two plots show the performance of the same learning runs split out into just rare events and just normal events, respectively, i.e., the middle lines are weighted averages of the top and bottom lines. In every case, OFFER substantially outperforms vanilla policy gradients.

References

[1] TP Imthias Ahamed, Vivek S Borkar, and S Juneja. Adaptive importance sampling technique for markov chains using stochastic approximation. *Operations Research*, 54(3):489–504, 2006.

[2] Dimitri P Bertsekas and Ian B Rhodes. On the minimax feedback control of uncertain dynamic systems. In *Decision and Control, 1971 IEEE Conference on*, pages 451–455. IEEE, 1971.

[3] Shalabh Bhatnagar, Richard Sutton, Mohammad Ghavamzadeh, and Mark Lee. Natural actor–critic algorithms. *Automatica*, 2009.

[4] Antoine Cully, Jeff Clune, Danesh Tarapore, and Jean-Baptiste Mouret. Robots that can adapt like animals. *Nature*, 521(7553):503–507, 2015.

[5] Thomas Degris, Patrick M Pilarski, and Richard S Sutton. Model-free reinforcement learning with continuous action in practice. In *American Control Conference (ACC), 2012*, pages 2177–2182. IEEE, 2012.

[6] Marc Peter Deisenroth, Gerhard Neumann, Jan Peters, et al. A survey on policy search for robotics. *Foundations and Trends in Robotics*, 2(1-2):1–142, 2013.

[7] Paritosh Y Desai and Peter W Glynn. Simulation in optimization and optimization in simulation: a markov chain perspective on adaptive monte carlo algorithms. In *Proceedings of the 33nd conference on Winter simulation*, pages 379–384. IEEE Computer Society, 2001.

[8] Jordan Frank, Shie Mannor, and Doina Precup. Reinforcement learning in the presence of rare events. In *Proceedings of the 25th international conference on Machine learning*, pages 336–343. ACM, 2008.

[9] Thomas Furmston and David Barber. A unifying perspective of parametric policy search methods for markov decision processes. In *Advances in Neural Information Processing Systems*, pages 2717–2725, 2012.

[10] Javier García and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16:1437–1480, 2015.

[11] Clement Gehring and Doina Precup. Smart exploration in reinforcement learning using absolute temporal difference errors. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pages 1037–1044. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

[12] Peter W. Glynn. Likelihood ratio gradient estimation for stochastic systems. *Commun. ACM*, 33(10):75–84, October 1990.

[13] Matthias Heger. Consideration of risk in reinforcement learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 105–111, 1994.

[14] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[15] Rogier Koppejan and Shimon Whiteson. Neuroevolutionary reinforcement learning for generalized control of simulated helicopters. *Evolutionary intelligence*, 4(4):219–241, 2011.

[16] María Malfaz and Miguel A Salichs. Learning to avoid risky actions. *Cybernetics and Systems*, 42(8):636–658, 2011.

[17] Marvin K Nakayama, Ambuj Goyal, and Peter W Glynn. Likelihood ratio sensitivity analysis for markovian models of highly dependable systems. *Operations Research*, 42(1):137–157, 1994.

[18] Art B. Owen. *Monte Carlo theory, methods and examples*. 2013.

[19] Supratik Paul, Kamil Ciosek, Michael A. Osborne, and Shimon Whiteson. Alternating optimisation and quadrature for robust reinforcement learning. *CoRR*, abs/1605.07496, 2016.

[20] Jan Peters and Stefan Schaal. Policy gradient methods for robotics. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 2219–2225. IEEE, 2006.

[21] Jan Peters and Stefan Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697, 2008.

[22] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *Proceedings of The 31st International Conference on Machine Learning*, pages 387–395, 2014.

[23] James C Spall. Adaptive stochastic approximation by the simultaneous perturbation method. *IEEE transactions on automatic control*, 45(10):1839–1853, 2000.

[24] Richard Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988.

[25] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, pages 1057–1063, 2000.

[26] Ronald Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.