Expected Policy Gradients

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

Abstract

We propose *expected policy gradients* (EPG), which unify stochastic policy gradients (SPG) and deterministic policy gradients (DPG) for reinforcement learning. Inspired by *expected sarsa*, EPG integrates across the action when estimating the gradient, instead of relying only on the action in the sampled trajectory. We establish a new *general policy gradient theorem*, of which the stochastic and deterministic policy gradient theorems are special cases. We also prove that EPG reduces the variance of the gradient estimates without requiring deterministic policies and, for the Gaussian case, with no computational overhead. Finally, we show that it is optimal in a certain sense to explore with a Gaussian policy such that the covariance is proportional to e^H, where H is the scaled Hessian of the critic with respect to the actions. We present empirical results confirming that this new form of exploration substantially outperforms DPG with the Ornstein-Uhlenbeck heuristic in four challenging MuJoCo domains.

Introduction

Policy gradient methods (Sutton et al., 2000; Peters and Schaal, 2006, 2008b; Silver et al., 2014), which optimise policies by gradient ascent, have enjoyed great success in reinforcement learning problems with large or continuous action spaces. The archetypal algorithm optimises an *actor*, i.e., a policy, by following a policy gradient that is estimated using a *critic*, i.e., a value function.

The policy can be stochastic or deterministic, yielding *stochastic policy gradients* (SPG) (Sutton et al., 2000) or *deterministic policy gradients* (DPG) (Silver et al., 2014). The theory underpinning these methods is quite fragmented, as each approach has a separate policy gradient theorem guaranteeing the policy gradient is unbiased under certain conditions.

Furthermore, both approaches have significant shortcomings. For SPG, variance in the gradient estimates means that many trajectories are usually needed for learning. Since gathering trajectories is typically expensive, there is a great need for more sample efficient methods.

DPG's use of deterministic policies mitigates the problem of variance in the gradient but raises other difficulties. The theoretical support for DPG is limited since it assumes a critic that approximates $\nabla_a Q$ when in practice it approximates Q instead. In addition, DPG learns *off-policy*[1], which is undesirable when we want learning to take the cost of exploration into account. More importantly, learning off-policy necessitates designing a suitable *exploration policy*, which is difficult in practice. In fact, efficient exploration in DPG is an open problem and most applications simply use independent Gaussian noise or the Ornstein-Uhlenbeck heuristic (Uhlenbeck and Ornstein, 1930; Lillicrap et al., 2015).

In this paper, we propose a new approach called *expected policy gradients* (EPG) that unifies policy gradients in a way that yields both theoretical and practical insights. Inspired by *expected sarsa* (Sutton and Barto, 1998; van Seijen et al., 2009), the main idea is to integrate across the action selected by the stochastic policy when estimating the gradient, instead of relying only on the action selected during the sampled trajectory.

EPG enables two theoretical contributions. First, we establish a number of equivalences between EPG and DPG, among which is a new *general policy gradient theorem*, of which the stochastic and deterministic policy gradient theorems are special cases. Second, we prove that EPG reduces the variance of the gradient estimates without requiring deterministic policies and, for the Gaussian case, with no computational overhead over SPG.

EPG also enables a practical contribution: a principled exploration strategy for continuous problems. We show that it is optimal in a certain sense to explore with a Gaussian policy such that the covariance is proportional to e^H, where H is the scaled Hessian of the critic with respect to the actions. We present empirical results confirming that this new approach to exploration substantially outperforms DPG with Ornstein-Uhlenbeck exploration in four challenging MuJoCo domains.

Background

A *Markov decision process* is a tuple $(S, A, R, p, p_0, \gamma)$ where S is a set of states, A is a set of actions (in practice either $A = \mathbb{R}^d$ or A is finite), $R(s, a)$ is a reward function, $p(s' \mid a, s)$ is a transition kernel, p_0 is an initial state distribution, and $\gamma \in [0, 1)$ is a discount factor. A policy $\pi(a \mid s)$

[1] We show in this paper that, in certain settings, off-policy DPG is equivalent to EPG, our on-policy method.

is a distribution over actions given a state. We denote trajectories as $\tau^\pi = (s_0, a_0, r_0, s_1, a_1, r_1, \dots)$, where $s_0 \sim p_0$, $a_t \sim \pi(\cdot \mid s_{t-1})$ and r_t is a sample reward. A policy π induces a Markov process with transition kernel $p_\pi(s' \mid s) = \int_a d\pi(a \mid s)p(s' \mid a, s)$ where we use the symbol $d\pi(a \mid s)$ to denote Lebesgue integration against the measure $\pi(a \mid s)$ where s is fixed. We assume the induced Markov process is ergodic with a single invariant measure defined for the whole state space. The value function is $V^\pi = \mathbb{E}_\tau\left[\sum_i \gamma_i r_i\right]$ where actions are sampled from π. The Q-function is $Q^\pi(a \mid s) = \mathbb{E}_R\left[r \mid s, a\right] + \gamma \mathbb{E}_{p(s \mid s)}\left[V^\pi(s') \mid s\right]$ and the advantage function is $A^\pi(a \mid s) = Q^\pi(a \mid s) - V^\pi(s)$. An optimal policy maximises the total return $J = \int_s dp_0(s)V^\pi(s)$. Since we consider only on-policy learning with just one current policy, we drop the π super/subscript where it is redundant.

If π is parameterised by θ, then *stochastic policy gradients* (SPG) (Sutton et al., 2000; Peters and Schaal, 2006, 2008b) perform gradient ascent on ∇J, the gradient of J with respect to θ (gradients without a subscript are always with respect to θ). For stochastic policies, we have:

$$\nabla J = \int_s d\rho(s) \int_a d\pi(a \mid s)\nabla \log \pi(a \mid s)(Q(a, s) + b(s)), \quad (1)$$

where ρ is the discounted-ergodic occupancy measure, defined in the supplement, and $b(s)$ is a baseline, which can be any function that depends on the state but not the action, since $\int_a d\pi(a \mid s)\nabla \log \pi(a \mid s)b(s) = 0$. Typically, (1) is approximated from samples from a trajectory τ of length T:

$$\hat{\nabla} J = \sum_{t=0}^{T} \gamma^t \nabla \log \pi(a_t \mid s_t)(\hat{Q}(s_t, a_t) + b(s_t)). \quad (2)$$

If the policy is deterministic (we denote it $\pi(s)$), we can use *deterministic policy gradients* (Silver et al., 2014) instead:

$$\nabla J = \int_s d\rho(s)\nabla\pi(s)\nabla_a Q(a = \pi(s), s). \quad (3)$$

This update is then approximated using samples:

$$\hat{\nabla} J = \sum_{t=0}^{T} \gamma^t \nabla\pi(s)\nabla_a \hat{Q}(a = \pi(s_t), s_t). \quad (4)$$

Since the policy is deterministic, the problem of exploration is addressed using an external source of noise, typically modeled using a zero-mean Ornstein-Uhlenbeck (OU) process (Uhlenbeck and Ornstein, 1930; Lillicrap et al., 2015) parametrized by ψ and σ:

$$n_i \leftarrow -n_{i-1}\psi + \mathcal{N}(0, \sigma I) \quad a \sim \pi(s) + n_i. \quad (5)$$

In (2) and (4), $\hat{Q}$ is a *critic* that approximates Q and can be learned by *sarsa* (Rummery and Niranjan, 1994; Sutton, 1996):

$$\hat{Q}(s_t, a_t) \leftarrow \hat{Q}(s_t, a_t) + \alpha\big[r_{t+1} + \gamma\hat{Q}(s_{t+1}, a_{t+1}) - \hat{Q}(s_t, a_t)\big]. \quad (6)$$

Alternatively, we can use *expected sarsa* (Sutton and Barto, 1998; van Seijen et al., 2009), which marginalises out a_{t+1}, the distribution over which is specified by the known policy, to reduce the variance in the update:

$$\hat{Q}(s_t, a_t) \leftarrow \hat{Q}(s_t, a_t) + \alpha\big[r_{t+1} + \gamma \int_a d\pi(a \mid s)\hat{Q}(s_{t+1}, a) - \hat{Q}(s_t, a_t)\big]. \quad (7)$$

We could also use advantage learning (Baird and others, 1995) or LSTDQ (Lagoudakis and Parr, 2003). If the critic's function approximator is *compatible*, then the actor, i.e., π, converges (Sutton et al., 2000).

Instead of learning $\hat{Q}$, we can set $b(s) = -V(s)$ so that $Q(a, s) + b(s) = A(s, a)$ and then use the TD error $\delta(r, s', s) = r + \gamma V(s') - V(s)$ as an estimate of $A(s, a)$ (Bhatnagar et al., 2008):

$$\hat{\nabla} J = \sum_{t=0}^{T} \gamma^t \nabla \log \pi(a_t \mid s_t)(r + \gamma\hat{V}(s') - \hat{V}(s)), \quad (8)$$

where $\hat{V}(s)$ is an approximate value function learned using any policy evaluation algorithm. (8) works because $\mathbb{E}\left[\delta(r, s', s) \mid a, s\right] = A(s, a)$, i.e., the TD error is an unbiased estimate of the advantage function. The benefit of this approach is that it is sometimes easier to approximate V than Q and that the return in the TD error is unprojected, i.e., it is not distorted by function approximation. However, the TD error is noisy, introducing variance in the gradient.

To cope with this variance, we can reduce the learning rate when the variance of the gradient would otherwise explode, using, e.g., *Adam* (Kingma and Ba, 2014), *natural policy gradients* (Kakade, 2002; Amari, 1998; Peters and Schaal, 2008a), the adaptive step size method (Pirotta, Restelli, and Bascetta, 2013) or *Newton's method* (Furmston and Barber, 2012; Parisi, Pirotta, and Restelli, 2016). However, this results in slow learning when the variance is high. One can also use PGPE, which replaces the stochastic policy with a distribution over deterministic policies (Sehnke et al., 2010). However, PGPE precludes updating the current policy during the episode and makes it difficult to explore efficiently.

We can also eliminate all variance caused by the policy at the cost of making the policy deterministic and using the DPG update, which usually necessitates performing off-policy exploration. EPG, presented below, reduces to DPG in many useful cases, while providing a principled way to explore and also allowing for stochastic policies.

Yet another way to eliminate variance in the actor is not to have an actor at all, instead selecting actions soft-greedily with respect to $\hat{Q}$ learned using sarsa. This is trivial for discrete actions and can also be done with a one-step Newton's method for Q-functions that are quadric in the actions (Gu et al., 2016b).

Expected Policy Gradients

In this section, we propose *expected policy gradients* (EPG).

Main Algorithm

First, we introduce $I_\pi^Q(s)$ to denote the inner integral in (1):

$$\nabla J = \int_s d\rho(s) \underbrace{\int_a d\pi(a \mid s)\nabla \log \pi(a \mid s)(Q(a, s) + b(s))}_{I_\pi^Q(s)}$$

$$= \int_s d\rho(s) I_\pi^Q(s). \quad (9)$$

This suggests a new way to write the approximate gradient:

$$\hat{\nabla} J = \sum_{t=0}^{T} \underbrace{\gamma^t \hat{I}_{\pi}^{\hat{Q}}(s_t)}_{g_t}, \qquad (10)$$

where $\hat{I}_{\pi}^{\hat{Q}}(s)$ is some approximation to $I_{\pi}^{\hat{Q}}(s) = \int_a d\pi(a \mid s) \nabla \log \pi(a \mid s)(\hat{Q}(a, s) + b(s))$. This approach makes explicit that one step in estimating the gradient is to evaluate an integral to estimate $I_{\pi}^{\hat{Q}}(s)$. The main insight behind EPG is that, given a state, $I_{\pi}^{\hat{Q}}(s)$ is expressed fully in terms of known quantities. Hence we can manipulate it analytically to obtain a formula or we can just compute the integral using any numerical quadrature if an analytical solution is impossible.

SPG as given in (2) performs this quadrature using a simple one-sample Monte Carlo method. However, relying on such a method is unnecessary. In fact, the actions used to interact with the environment need not be used at all in the evaluation of $\hat{I}_{\pi}^{Q}(s)$ since a is a bound variable in the definition of $I_{\pi}^{Q}(s)$. The motivation is thus similar to that of expected sarsa but applied to the actor's gradient estimate instead of the critic's update rule. EPG, shown in Algorithm 1, uses (10) to form a policy gradient algorithm that repeatedly estimates $\hat{I}_{\pi}^{Q}(s)$ with an integration subroutine.

Algorithm 1 Expected Policy Gradients

```
$s \leftarrow s_0$, $t \leftarrow 0$
initialise optimiser, initialise policy $\pi$ parametrised by $\theta$
while not converged do
    $g_t \leftarrow \gamma^t$ DO-INTEGRAL($\hat{Q}, s, \pi_\theta$)
        ▷ $g_t$ is the estimated policy gradient as per (10)
    $\theta \leftarrow \theta +$ optimiser.UPDATE($g_t$)
    $a \sim \pi(\cdot, s)$
    $s', r \leftarrow$ simulator.PERFORM-ACTION(a)
    $\hat{Q}$.UPDATE($s, a, r, s'$)
    $t \leftarrow t + 1$
    $s \leftarrow s'$
end while
```

EPG has benefits even when an analytical solution is not possible: if the action space is low dimensional, numerical quadrature is cheap; if it is high dimensional, it is still often worthwhile to balance the expense of simulating the system with the cost of quadrature. Actually, even in the extreme case of expensive quadrature but cheap simulation, the limited resources available for quadrature could still be better spent on EPG with smart quadrature than SPG with simple Monte Carlo. One of the motivations of DPG was precisely that the simple one-sample Monte-Carlo quadrature implicitly used by SPG often yields high variance gradient estimates, even with a good baseline. To see why, consider Figure 1 (left). A simple Monte Carlo method evaluates the integral by sampling one or more times from $\pi(a \mid s)$ (blue) and evaluating $\nabla_\mu \log \pi(a \mid s) Q(a, s)$ (red) as a function of a. A baseline can decrease the variance by adding a multiple of $\nabla_\mu \log \pi(a \mid s)$ to the red curve, but the problem remains that the red curve has high values where the blue curve is almost zero. Consequently, substantial variance persists, whatever the baseline, even with a simple linear Q-function, as shown in Figure 1 (right). DPG addressed this problem for deterministic policies but EPG extends it to stochastic ones.

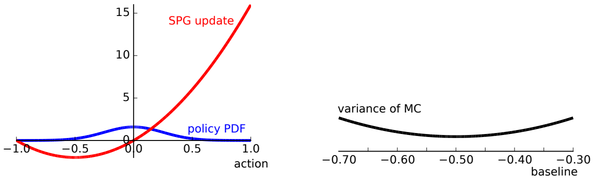

Figure 1: At left, $\pi(a \mid s)$ for a Gaussian policy with $\mu = \theta = 0$ at a given state and constant σ^2 (blue) and the SPG update $\nabla_\theta \log \pi(a \mid s) Q(a, s)$ (in red), obtained for $Q = \frac{1}{2} + \frac{1}{2}a$. At right, the variance of a simple single-sample Monte Carlo estimator as a function of the baseline. In a simple multi-sample Monte Carlo method, the variance would go down as the number of samples.

Relationship to Other Methods

EPG has some similarities with VINE sampling (Schulman et al., 2015), which uses an (intrinsically noisy) Monte Carlo quadrature with many samples.[2] However, the example in Figure 1 shows that even with a computationally expensive many-sample Monte Carlo method, the problem of variance remains, regardless of the baseline.

EPG is also related to variance minimisation techniques that interpolate between two estimators, e.g., (Gu et al., 2016a, Eq. 7) is similar to Corollary 4. However, EPG uses a quadric (not linear) approximation to the critic, which is crucial for exploration. Furthermore, it completely eliminates variance in the inner integral, as opposed to just reducing it.

The idea behind EPG was also independently and concurrently developed as Mean Actor Critic (Asadi et al., 2017), though only for discrete actions and without a supporting theoretical analysis.

Gaussian Policies

EPG is particularly useful when we make the common assumption of a Gaussian policy: we can then perform the integration analytically under reasonable conditions. We show below (see Lemma 3) that the update to the policy mean computed by EPG is equivalent to the DPG update. Moreover, a simple formula for the covariance can be derived (see Lemma 2). Algorithms 2 and 3 show the resulting special case of EPG, which we call *Gaussian policy gradients* (GPG).

Surprisingly, GPG is on-policy but nonetheless fully equivalent to DPG, an off-policy method, with a particular form of exploration. Hence, GPG, by specifying the policy's covariance, can be seen as a derivation of an exploration strategy for DPG. In this way, GPG addresses an important open question. As we show later, this leads to improved performance in practice.

[2] VINE sampling also differs from EPG by performing independent rollouts of Q, requiring a simulator with reset.

Algorithm 2 Gaussian Policy Gradients

```
$s \leftarrow s_0, t \leftarrow 0$
initialise optimiser
while not converged do
    $g_t \leftarrow \gamma^t$ DO-INTEGRAL-GAUSS($\hat{Q}, s, \pi_\theta$)
    $\theta \leftarrow \theta +$ optimiser.UPDATE($g_t$)
        ▷ policy parameters $\theta$ are updated using gradient
    $\Sigma_s \leftarrow$ GET-COVARIANCE($\hat{Q}, s, \pi_\theta$)
                                        ▷ $\Sigma_s$ computed from scratch
    $a \sim \pi(\cdot \mid s)$                    ▷ $\pi(\cdot \mid s) = N(\mu_s, \Sigma_s)$
    $s', r \leftarrow$ simulator.PERFORM-ACTION(a)
    $\hat{Q}$.UPDATE($s, a, r, s'$)
    $t \leftarrow t + 1$
    $s \leftarrow s'$
end while
```

Algorithm 3 Gaussian Integrals

```
function DO-INTEGRAL-GAUSS($\hat{Q}, s, \pi_\theta$)
    $I^Q_{\pi(s),\mu_s} \leftarrow (\nabla\mu_s)\nabla_a\hat{Q}(a = \mu_s, s)$  ▷ Use Lemma 1
    return $I^Q_{\pi(s),\mu_s}$
end function

function GET-COVARIANCE($\hat{Q}, s, \pi_\theta$)
    $H \leftarrow$ COMPUTE-HESSIAN($\hat{Q}(\mu_s, s)$)
    return $\sigma_0^2 e^{cH}$                          ▷ Use Lemma 2
end function
```

The computational cost of GPG is small: while it must store a Hessian matrix $H(a, s) = \nabla_a^2\hat{Q}(a, s)$, its size is only $d \times d$, where $A = \mathbb{R}^d$, which is typically small, e.g., $d = 6$ for HalfCheetah-v1. This Hessian is the same size as the policy's covariance matrix, which any policy gradient must store anyway, and should not be confused with the Hessian with respect to the parameters of the neural network, as used with Newton's or natural gradient methods (Peters and Schaal, 2008a; Furmston, Lever, and Barber, 2016), which can easily have thousands of entries. Hence, GPG obtains EPG's variance reduction essentially for free.

Analysis

In this section, we analyse EPG, showing that it unifies SPG and DPG, that $\hat{I}^Q_\pi(s)$ can often be computed analytically, and that EPG has lower variance than SPG.

General Policy Gradient Theorem

We begin by stating our most general result, showing that EPG can be seen as a generalisation of both SPG and DPG. To do this, we first state a new general policy gradient theorem. We use the shorthand ∇ without a subscript to denote the gradient with respect to policy parameters θ.

Theorem 1 (General Policy Gradient Theorem)**.** *If $\pi(\cdot, s)$ is a normalised Lebesgue measure for all s, then*

$$\nabla J = \int_s d\rho(s) \underbrace{\left[\nabla V(s) - \int_a d\pi(a, s)\nabla Q(a, s)\right]}_{I_G(s)}.$$

Proof. We begin by expanding the following expression.

$$\begin{aligned}
&\textstyle\int_s d\rho(s) \int_a d\pi(a, s)\nabla Q(a, s) \\
&= \textstyle\int_s d\rho(s) \int_a d\pi(a,s)\nabla(R(a,s) + \gamma \int_{s'} dp(s'|s,a)V(s')) \\
&= \textstyle\int_s d\rho(s) \int_a d\pi(a,s)(\underbrace{\nabla R(a,s)}_{0} + \gamma \int_{s'} dp(s'|s,a)\nabla V(s')) \\
&= \gamma \textstyle\int_s d\rho(s) \int_{s'} dp_\pi(s' \mid s)\nabla V(s') \\
&= \textstyle\int_s d\rho(s)\nabla V(s) - \underbrace{\textstyle\int_s dp_0(s)\nabla V(s)}_{\nabla J} \\
&= \textstyle\int_s d\rho(s)\nabla V(s) - \nabla J.
\end{aligned}$$

The first equality follows by expanding the definition of Q and the penultimate one follows from Lemma B (in the supplement). Then the theorem follows by rearranging terms. □

The crucial benefit of Theorem 1 is that it works for all policies, both stochastic and deterministic, unifying previously separate derivations for the two settings. To show this, in the following two corollaries, we use Theorem 1 to recover the *stochastic policy gradient theorem* (Sutton et al., 2000) and the *deterministic policy gradient theorem* (Silver et al., 2014), in each case by introducing additional assumptions to obtain a formula for $I_G(s)$ expressible in terms of known quantities.

Corollary 1 (Stochastic Policy Gradient Theorem)**.** *If $\pi(\cdot \mid s)$ is differentiable, then*

$$\begin{aligned}
\nabla J &= \textstyle\int_s d\rho(s) I_G(s) \\
&= \textstyle\int_s d\rho(s) \int_a d\pi(a \mid s)\nabla \log \pi(a \mid s)Q(a, s).
\end{aligned}$$

Proof. We obtain the following by expanding ∇V.

$$\begin{aligned}
\nabla V &= \nabla \textstyle\int_a d\pi(a, s)Q(a, s) = \\
&\textstyle\int_a da(\nabla\pi(a, s))Q(a, s) + \int_a d\pi(a, s)(\nabla Q(a, s))
\end{aligned}$$

We obtain $I_G(s) = \int_a d\pi(a \mid s)\nabla \log \pi(a \mid s)Q(a, s) = I^Q_\pi(s)$ by plugging this into the definition of $I_G(s)$. We obtain ∇J by invoking Theorem 1 and plugging in the above expression for $I_G(s)$. □

We now recover the DPG update introduced in (3).

Corollary 2 (Deterministic Policy Gradient Theorem)**.** *If $\pi(\cdot \mid s)$ is a Dirac-delta measure (i.e., a deterministic policy) and $Q(\cdot, s)$ is differentiable, then*

$$\nabla J = \textstyle\int_s d\rho(s) I_G(s) = \int_s d\rho(s)\nabla\pi(s)\nabla_a Q(a, s).$$

Proof. We begin by obtaining an expression for $I_G(s)$.

$$\begin{aligned}
I_G(s) &= \nabla V(s) - \textstyle\int_a d\pi(a, s)\nabla Q(a, s) \\
&= \nabla V(s) - \gamma \textstyle\int_{s'} dp_\pi(s' \mid s)\nabla V(s') \\
&= \nabla\pi(s)\nabla_a Q(a, s).
\end{aligned}$$

Here, the second equality follows by expanding the definition of Q and the third follows from an established deterministic policy gradient result (Silver et al., 2014, Supplement, Eq. 1). We can then obtain ∇J by invoking Theorem 1 and plugging in the above expression for $I_G(s)$. □

These corollaries show that the choice between deterministic and stochastic policy gradients is fundamentally a choice of quadrature method. Hence, the empirical success of DPG relative to SPG (Silver et al., 2014; Lillicrap et al., 2015) can be understood in a new light. In particular, it can be attributed, not to a fundamental limitation of stochastic policies (indeed, stochastic policies are sometimes preferred), but instead to superior quadrature. DPG integrates over Dirac-delta measures, which is known to be easy, while SPG typically relies on simple Monte Carlo integration. Thanks to EPG, a deterministic approach is no longer required to obtain a method with low variance.

We add as a sideline that since Theorem 1 can be written as $I_G(s) = \nabla V(s) - \gamma \int_{s'} dp_\pi(s' \mid s) \nabla V(s')$, which involves the derivatives of value functions, GPG resembles *value gradients* (Heess et al., 2015). However, in our case, we are learning ∇J directly and do not perform recursive estimation of ∇V as value gradient methods do.

Analytical Quadrature - Gaussian Policy

We now derive a lemma supporting GPG.

Lemma 1 (Gaussian Policy Gradients)**.** *If the policy is Gaussian, i.e.* $\pi(\cdot|s) \sim \mathcal{N}(\mu_s, \Sigma_s)$ *with* μ_s *and* $\Sigma_s^{1/2}$ *parametrised by* θ*, where* $\Sigma_s^{1/2}$ *is symmetric and* $\Sigma_s^{1/2}\Sigma_s^{1/2} = \Sigma_s$ *and the critic is of the form* $Q(a, s) = a^\top A(s)a + a^\top B(s) + const$ *where* $A(s)$ *is symmetric for every* s*, then* $I_\pi^Q(s) = I^Q_{\pi(s),\mu_s} + I^Q_{\pi(s),\Sigma_s^{1/2}}$*, where the mean and covariance components are given by* $I^Q_{\pi(s),\mu_s} = (\nabla\mu_s)(2A(s)\mu + B(s))$ *and* $I^Q_{\pi(s),\Sigma_s^{1/2}} = (\nabla\Sigma_s^{1/2})2A(s)\Sigma_s^{1/2}$.

See Lemma 1 in the supplement for proof of this result. While Lemma 1 requires the critic to be quadric in the actions, this assumption is not very restrictive since the coefficients $B(s)$ and $A(s)$ can be arbitrary continuous functions of the state, e.g., a neural network.

Arbitrary Critics

If Q does not meet the conditions of Lemma 1, we can approximate Q with a quadric function in the neighbourhood of the policy mean. This approximation is motivated by two arguments. First, in MDPs that model physical systems with reasonable reward functions, Q is fairly smooth. Second, policy gradients are a local, incremental method anyway – since the policy mean changes slowly, the values of Q for actions far from the policy mean are usually not relevant for the current update.

Corollary 3 (Approximate Gaussian Policy Gradients with an Arbitrary Critic)**.** *If the policy is Gaussian, i.e.* $\pi(\cdot|s) \sim \mathcal{N}(\mu_s, \Sigma_s)$ *with* μ_s *and* $\Sigma_s^{1/2}$ *parametrised by* θ *as in Lemma 1 and any critic* $Q(a, s)$ *doubly differentiable with respect to actions for each state, then* $I^Q_{\pi(s),\mu_s} \approx (\nabla\mu_s)\nabla_a Q(a = \mu_s, s)$ *and* $I^Q_{\pi(s),\Sigma_s^{1/2}} \approx (\nabla\Sigma_s^{1/2})H(\mu_s, s)\Sigma_s^{1/2}$*, where* $H(\mu_s, s)$ *is the Hessian of* Q *with respect to* a*, evaluated at* μ_s *for a fixed* s*.*

Proof. We begin by approximating the critic (for a given s) using the first two terms of the Taylor expansion of Q in μ_s.

$$\begin{aligned} Q(a, s) \approx Q(\mu_s, s) + (a - \mu_s)^\top \nabla_a Q(a = \mu_s, s) \\ + \tfrac{1}{2}(a - \mu_s)^\top H(\mu_s, s)(a - \mu_s) \\ = \tfrac{1}{2} a^\top H(\mu_s,s)a + a^\top(\nabla_a Q(a{=}\mu_s,s) - H(\mu_s,s)\mu_s) + \text{const.} \end{aligned}$$

Because of the series truncation, the function on the righthand side is quadric and we can then use Lemma 1:

$$\begin{aligned} I^Q_{\pi(s),\mu_s} &= \nabla\mu_s(2\tfrac{1}{2}H(\mu_s,s)\mu_s + \nabla_a Q(a{=}\mu_s,s) - H(\mu_s,s)\mu_s) \\ &= \nabla\mu_s \nabla_a Q(a{=}\mu_s,s) \\ I^Q_{\pi(s),\Sigma_s^{1/2}} &= (\nabla\Sigma_s^{1/2})(\tfrac{1}{2}2H(\mu_s,s)\Sigma_s^{1/2}) = (\nabla\Sigma_s^{1/2})H(\mu_s,s)\Sigma_s^{1/2}. \end{aligned}$$

□

To actually obtain the Hessian, we could use automatic differentiation to compute it analytically. Alternatively, we can observe that, if the critic really is quadric, we can just read off the coefficients of the quadric term directly. Therefore, we can approximate the Hessian by generating a number of random action-values around μ_s, computing the Q values, and (locally) fitting a quadric. This process is typically more computationally expensive than automatic differentiation but has the advantage of working with ReLU networks (where the true Hessian is zero but we still have a kind of global curvature after smoothing) and leveraging more information from the critic (since the evaluation is at more than one point).

Linear GPG

We now state a consequence of Lemma 1 for the case when the critic Q is linear in the actions, i.e., the quadric term is always zero.

Corollary 4 (Linear Gaussian Policy Gradients)**.** *If the policy is Gaussian, i.e.,* $\pi(\cdot|s) \sim \mathcal{N}(\mu_s, \Sigma_s)$ *with* μ_s *parametrised by* θ *and the critic is of the form* $Q(a \mid s) = a^\top B(s) + const$*, then* $I_\pi^Q(s) = (\nabla\mu_s)B(s)$*. Moreover, it is unnecessary to parameterise* $\Sigma_s^{1/2}$ *since the policy gradient w.r.t. to* $\Sigma_s^{1/2}$ *is zero (i.e., a linear Q-function does not give any information about the exploration covariance).*

We make Corollary 4 explicit for two reasons. First, it is useful for showing an equivalence between DPG and EPG (see below). Second, it may actually be useful for a non-trivial class of physical systems: if the time-sampling frequency is high enough (which implies acting in small steps), the critic is effectively only used to say if a small step one way is preferable to small step the other way – a linear property.

Equivalences between EPG and DPG

The update for the policy mean obtained in Corollary 3 is the same as the DPG update, linking the two methods:

$$I_\pi^Q(s) = (\nabla\mu_s)\nabla_a Q(a = \mu_s, s).$$

We now formalise the equivalences between EPG and DPG. First, on-policy GPG with a linear critic (or an arbitrary critic approximated by the first term in the Taylor expansion) is equivalent to DPG with a Gaussian exploration policy where the covariance stays the same. This follows from Corollary 4. Second, on-policy GPG with a quadric critic (or an arbitrary critic approximated by the first two terms in the Taylor expansion) is equivalent to DPG with a Gaussian exploration policy where the covariance is computed using the update (where α_n is a sequence of step-sizes):

$$\Sigma_s^{1/2} \leftarrow \Sigma_s^{1/2} + \alpha_n H(s) \Sigma_s^{1/2}. \tag{11}$$

This follows from Corollary 3. Third, and most generally, for any critic at all (not necessarily quadric), DPG is a kind of EPG for a particular choice of quadrature (using a Dirac measure). This follows from Theorem 1.

Surprisingly, this means that DPG, normally considered to be off-policy, can also be seen as on-policy when exploring with Gaussian noise. Furthermore, the compatible critic for DPG (Silver et al., 2014) is indeed linear in the actions. Hence, this relationship holds whenever DPG uses a compatible critic.[3] Furthermore, Lemma 1 lends new legitimacy to the common practice of replacing the critic required by the DPG theory, which approximates $\nabla_a Q$, with one that approximates Q itself, as done in SPG and EPG.

Exploration using the Hessian

The second equivalence given above suggests that we can include the covariance in the actor network and learn it along with the mean. However, another option is to compute it from scratch at each iteration by analytically computing the result of applying (11) infinitely many times.

Lemma 2 (Exploration Limit). *The iterative procedure defined by the equation $\Sigma_s^{1/2} \leftarrow \Sigma_s^{1/2} + \alpha H(s) \Sigma_s^{1/2}$ applied n times using the diminishing learning rate $\alpha = 1/n$ converges to $\Sigma_s^{1/2} \propto e^{H(s)}$ as $n \to \infty$.*

Proof. Consider the sequence $(\Sigma_s^{1/2})_0 = \sigma_0 I$, $(\Sigma_s^{1/2})_n = (\Sigma_s^{1/2})_{n-1} + \alpha H(s) (\Sigma_s^{1/2})_{n-1}$. Expanding out the recursion, the n-th element of the sequence is given as:

$$(\Sigma_s^{1/2})_n = (I + \alpha H(s))^n (\Sigma_s^{1/2})_0.$$

We diagonalise the Hessian as $H(s) = U \Lambda U^\top$ for some orthonormal matrix U and obtain the following expression for $(\Sigma_s^{1/2})_n$.

$$(\Sigma_s^{1/2})_n = (I + \alpha U \Lambda U^\top)^n (\Sigma_s^{1/2})_0 = U (I + \alpha \Lambda)^n U^\top (\Sigma_s^{1/2})_0$$

Since we have $\lim_{n\to\infty} (1 + \frac{1}{n} \lambda)^n = e^\lambda$ for each diagonal entry of Λ, we plug $\alpha = \frac{1}{n}$ and obtain the identity:

$$\lim_{n\to\infty} (\Sigma_s^{1/2})_n = U e^\Lambda U^\top (\Sigma_s^{1/2})_0 = \sigma_0 e^{H(s)}.$$

□

The practical implication of Lemma 2 is that, in a policy gradient method, it is justified to use Gaussian exploration with covariance proportional to e^{cH} for some reward scaling constant c. Thus by exploring with (scaled) covariance e^{cH}, we obtain a principled alternative to the Ornstein-Uhlenbeck heuristic defined in (5). Our results below show that it also performs much better in practice.

Lemma 2 has an intuitive interpretation. If $H(s)$ has a large positive eigenvalue λ, then $\hat{Q}(s, \cdot)$ has a sharp minimum along the corresponding eigenvector, and the corresponding eigenvalue of Σ is e^λ, i.e., also large. The result is a large exploration bonus along that direction, enabling the algorithm to leave local minima. Conversely, if λ is negative, then $\hat{Q}(s, \cdot)$ has a maximum and so e^λ is small, since exploration is not needed.

Variance Analysis

We now prove that for any policy, the EPG estimator of (10) has lower variance than the SPG estimator of (2).

Lemma 3. *If for all $s \in S$, the random variable $\nabla \log \pi(a \mid s) \hat{Q}(s, a)$ where $a \sim \pi(\cdot|s)$ has nonzero variance, then*

$$\mathbb{V}_\tau \left[\sum_{t=0}^{\infty} \gamma^t \nabla \log \pi(a_t|s_t) (\hat{Q}(s_t, a_t) + b(s_t)) \right] > \mathbb{V}_\tau \left[\sum_{t=0}^{\infty} \gamma^t I_\pi^{\hat{Q}}(s_t) \right].$$

The proof is deferred to the supplement (see Lemma 3 there). Lemma 3's assumption is reasonable since the only way a random variable $\nabla \log \pi(a \mid s) \hat{Q}(s, a)$ could have zero variance is if it were the same for all actions in the policy's support (except for sets of measure zero), in which case optimising the policy would be unnecessary. Since we know that both the estimators of (2) and (10) are unbiased, the estimator with lower variance has lower MSE.

Extension to Entropy Regularisation

On-policy SPG sometimes includes an entropy term in the gradient in order to aid exploration by making the policy more stochastic. The gradient of the differential entropy[4] $\mathcal{H}(s)$ of the policy at state s is defined as follows.

$$\begin{aligned}
-\nabla \mathcal{H}(s) &= \nabla \int_a d\pi(a|s) \log \pi(a|s) \\
&= \int_a da \nabla \pi(a|s) \log \pi(a|s) + \int_a d\pi(a|s) \nabla \log \pi(a|s) \\
&= \int_a da \nabla \pi(a|s) \log \pi(a|s) + \int_a d\pi(a|s) \tfrac{1}{\pi(a|s)} \nabla \pi(a|s) \\
&= \int_a da \nabla \pi(a|s) \log \pi(a|s) + \nabla \underbrace{\int_a d\pi(a|s)}_{1} \\
&= \int_a da \nabla \pi(a|s) \log \pi(a|s) = \int_a d\pi(a|s) \nabla \log \pi(a|s) \log \pi(a|s).
\end{aligned}$$

Typically, we weight the entropy update with the policy gradient update:

$$\begin{aligned}
I_G^E(s) &= I_G(s) + \alpha \nabla \mathcal{H}(s) \\
&= \int_a d\pi(a|s) \nabla \log \pi(a|s) (Q(a, s) - \alpha \log \pi(a|s)).
\end{aligned}$$

This equation makes clear that performing entropy regularisation is equivalent to using a different critic with Q-values shifted by $\alpha \log \pi(a|s)$; this holds for both SPG and EPG.

[3] The notion of compatibility of a critic is different for stochastic and deterministic policy gradients.

[4] For discrete action spaces, the same derivation with integrals replaced by sums holds for the entropy.

Domain	$\hat{\sigma}_{\text{DPG}}$	$\hat{\sigma}_{\text{EPG}}$
HalfCheetah-v1	1336.39 [1107.85, 1614.51]	1056.15 [875.54, 1275.94]
InvertedPendulum-v1	291.26 [241.45, 351.88]	0.00 n/a
Reacher2d-v1	1.22 [0.63, 2.31]	0.13 [0.07, 0.26]
Walker2d-1	543.54 [450.58, 656.65]	762.35 [631.98, 921.00]

Table 1: Estimated standard deviation (mean and 90% interval) across runs after learning.

Experiments

While EPG has many potential uses, we focus on empirically evaluating one particular application: exploration driven by the Hessian exponential (as introduced in Algorithm 2 and Lemma 2), replacing the standard Ornstein-Uhlenbeck (OU) exploration in continuous action domains. To this end, we applied EPG to four domains modelled with the MuJoCo physics simulator (Todorov, Erez, and Tassa, 2012): HalfCheetah-v1, InvertedPendulum-v1, Reacher2d-v1 and Walker2d-v1 and compared its performance to DPG and SPG.

In practice, EPG differed from deep DPG (Lillicrap et al., 2015; Silver et al., 2014) only in the exploration strategy, though their theoretical underpinnings are different. The hyperparameters for DPG and those of EPG that are not related to exploration were taken from an existing benchmark (Islam et al., 2017; Brockman et al., 2016). The exploration hyperparameters for EPG were $\sigma_0^2 = 0.2$ and $c = 1.0$ where the exploration covariance is $\sigma_0^2 e^{cH}$. These values were obtained using a grid search from the set $\{0.2, 0.5, 1\}$ for σ_0^2 and $\{0.5, 1.0, 2.0\}$ for c over the HalfCheetah-v1 domain. Since c is just a constant scaling the rewards, it is reasonable to set it to 1.0 whenever reward scaling is already used. Hence, our exploration strategy has just one hyperparameter σ_0^2 as opposed specifying a pair of parameters (standard deviation and mean reversion constant) for OU. We used the same learning parameters for the other domains. For SPG[5], we used OU exploration and a constant diagonal covariance of 0.2 in the actor update (this approximately corresponds to the average variance of the OU process over time). The other parameters for SPG are the same as for the rest of the algorithm. For the learning curves, we obtained 90% confidence intervals around the learning curves. The learning curves show results of independent evaluation runs which used actions generated by the policy mean without any exploration noise.

The results (Figure 2) show that EPG's exploration strategy yields much better performance than DPG with OU. Furthermore, SPG does poorly, solving only the easiest domain (InvertedPendulum-v1) reasonably quickly, achieving slow progress on HalfCheetah-v1, and failing entirely on the other domains. This is not surprising DPG was introduced precisely to solve the problem of high variance SPG estimates on this type of problem. In InvertedPendulum-v1, SPG initially learns quickly, outperforming the other methods. This is because noisy gradient updates provide a crude, indirect form of exploration that happens to suit this problem. Clearly, this is inadequate for more complex domains: even for this simple domain it leads to subpar performance late in learning.

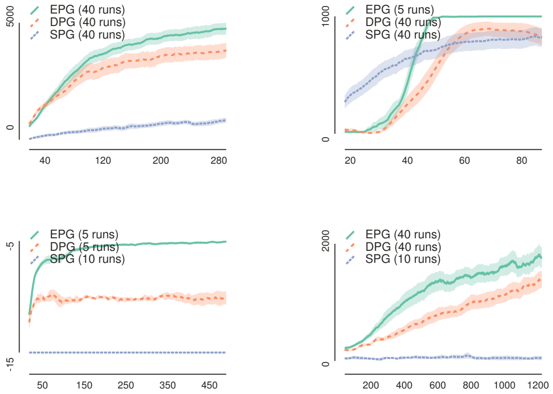

Figure 2: Learning curves (mean and 90% interval) for HalfCheetah-v1 (top left), InvertedPendulum-v1 (top right), Reacher2d-v1 (bottom left, clipped at -14) and Walker2d-v1 (bottom right). The number of independent training runs is in parentheses. Horizontal axis is scaled in thousands of steps.

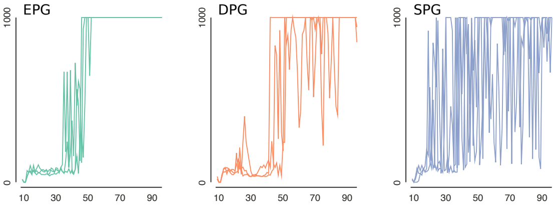

Figure 3: Three runs for EPG (left), DPG (middle) and SPG (right) for the InvertedPendulum-v1 domain, demonstrating that EPG shows much less unlearning.

In addition, EPG typically learns more consistently than DPG with OU. In two tasks, the empirical standard deviation across runs of EPG ($\hat{\sigma}_{\text{EPG}}$) was substantially lower than that of DPG ($\hat{\sigma}_{\text{DPG}}$) at the end of learning, as shown in Table 1. For the other two domains, the confidence intervals around the empirical standard deviations for DPG and EPG were too wide to draw conclusions.

Surprisingly, for InvertedPendulum-v1, DPG's learning curve declines late in learning. The reason can be seen in the individual runs shown in Figure 3: both DPG and SPG suffer from severe unlearning. This unlearning cannot be explained by exploration noise since the evaluation runs just use the mean action, without exploring. Instead, OU exploration in DPG may be too coarse, causing the optimiser to exit good optima, while SPG unlearns due to noise in the gradients. The noise also helps speed initial learning, as described above, but this does not transfer to other domains. EPG avoids this problem by automatically reducing the noise when it finds a good optimum, i.e., a Hessian with large negative eigenvalues.

[5] We tried learning the covariance for SPG but the covariance estimate was unstable; no regularisation hyperparameters we tested matched SPG's performance with OU even on the simplest domain.

Conclusions

This paper proposed a new policy gradient method called *expected policy gradients* (EPG), that integrates across the action selected by the stochastic policy. We used EPG to prove a new general policy gradient theorem subsuming the stochastic and deterministic policy gradient theorems. We also showed that, under certain realistic conditions, the quadrature required by EPG can be performed analytically, allowing DPG with principled exploration. We presented empirical results confirming that this application of EPG outperforms DPG and SPG on four domains.

Acknowledgements

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement number 637713).

References

Amari, S.-I. 1998. Natural gradient works efficiently in learning. *Neural computation* 10(2):251–276.

Asadi, K.; Allen, C.; Roderick, M.; Mohamed, A.-r.; Konidaris, G.; and Littman, M. 2017. Mean Actor Critic. *ArXiv e-prints*.

Baird, L., et al. 1995. Residual algorithms: Reinforcement learning with function approximation. In *Proceedings of the twelfth international conference on machine learning*, 30–37.

Bhatnagar, S.; Ghavamzadeh, M.; Lee, M.; and Sutton, R. S. 2008. Incremental natural actor-critic algorithms. In *Advances in neural information processing systems*, 105–112.

Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. Openai gym. *arXiv preprint arXiv:1606.01540*.

Furmston, T., and Barber, D. 2012. A unifying perspective of parametric policy search methods for markov decision processes. In *Advances in neural information processing systems*, 2717–2725.

Furmston, T.; Lever, G.; and Barber, D. 2016. Approximate newton methods for policy search in markov decision processes. *Journal of Machine Learning Research* 17(227):1–51.

Gu, S.; Lillicrap, T.; Ghahramani, Z.; Turner, R. E.; and Levine, S. 2016a. Q-prop: Sample-efficient policy gradient with an off-policy critic. *arXiv preprint arXiv:1611.02247*.

Gu, S.; Lillicrap, T.; Sutskever, I.; and Levine, S. 2016b. Continuous deep q-learning with model-based acceleration. In *International Conference on Machine Learning*, 2829–2838.

Heess, N.; Wayne, G.; Silver, D.; Lillicrap, T.; Erez, T.; and Tassa, Y. 2015. Learning continuous control policies by stochastic value gradients. In *Advances in Neural Information Processing Systems*, 2944–2952.

Islam, R.; Henderson, P.; Gomrokchi, M.; and Precup, D. 2017. Reproducibility of benchmarked deep reinforcement learning tasks for continuous control. *arXiv preprint arXiv:1708.04133*.

Kakade, S. M. 2002. A natural policy gradient. In *Advances in neural information processing systems*, 1531–1538.

Kingma, D., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lagoudakis, M. G., and Parr, R. 2003. Least-squares policy iteration. *Journal of machine learning research* 4(Dec):1107–1149.

Lillicrap, T. P.; Hunt, J. J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; and Wierstra, D. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*.

Parisi, S.; Pirotta, M.; and Restelli, M. 2016. Multi-objective reinforcement learning through continuous pareto manifold approximation. *Journal of Artificial Intelligence Research* 57:187–227.

Peters, J., and Schaal, S. 2006. Policy gradient methods for robotics. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, 2219–2225. IEEE.

Peters, J., and Schaal, S. 2008a. Natural actor-critic. *Neurocomputing* 71(7):1180–1190.

Peters, J., and Schaal, S. 2008b. Reinforcement learning of motor skills with policy gradients. *Neural networks* 21(4):682–697.

Pirotta, M.; Restelli, M.; and Bascetta, L. 2013. Adaptive step-size for policy gradient methods. In *Advances in Neural Information Processing Systems*, 1394–1402.

Rummery, G. A., and Niranjan, M. 1994. *On-line Q-learning using connectionist systems*. University of Cambridge, Department of Engineering.

Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; and Moritz, P. 2015. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 1889–1897.

Sehnke, F.; Osendorfer, C.; Rückstieß, T.; Graves, A.; Peters, J.; and Schmidhuber, J. 2010. Parameter-exploring policy gradients. *Neural Networks* 23(4):551–559.

Silver, D.; Lever, G.; Heess, N.; Degris, T.; Wierstra, D.; and Riedmiller, M. 2014. Deterministic policy gradient algorithms. In *ICML*.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.

Sutton, R. S.; McAllester, D. A.; Singh, S. P.; and Mansour, Y. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, 1057–1063.

Sutton, R. S. 1996. Generalization in reinforcement learning: Successful examples using sparse coarse coding. *Advances in neural information processing systems* 1038–1044.

Todorov, E.; Erez, T.; and Tassa, Y. 2012. Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, 5026–5033. IEEE.

Uhlenbeck, G. E., and Ornstein, L. S. 1930. On the theory of the brownian motion. *Physical review* 36(5):823.

van Seijen, H.; van Hasselt, H.; Whiteson, S.; and Wiering, M. 2009. A theoretical and empirical analysis of expected sarsa. In *ADPRL 2009: Proceedings of the IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, 177–184.

Supplement

We first provide formal proofs for certain statements invoked by our paper. We then provide a brief discussion of the use of a learning rate that diminished in the trajectory length in the computation of the covariance.

Proofs

First, we prove two lemmas concerning the discounted-ergodic measure $\rho(s)$ which have been implicitly realised for some time but as far as we could find, never proved explicitly.

Definition 1 (Time-dependent occupancy)**.**

$$p(s \mid t=0) = p_0(s)$$
$$p(s' \mid t=i+1) = \int_s p(s' \mid s) p(s \mid t=i) \quad \textit{for} \quad i \geq 0$$

Definition 2 (Truncated trajectory)**.** *Define the trajectory truncated after N steps as $\tau^N = (s_0, a_0, r_0, s_1, a_1, r_1, \ldots, s_N)$.*

Observation 1 (Expectation wrt. truncated trajectory)**.** *Since $\tau_N = (s_0, s_1, s_2, \ldots, s_N)$ is associated with the density $\prod_{i=0}^{N-1} p(s_{i+1} \mid s_i) p_0(s_0)$, we have that*

$$\begin{aligned} &\mathbb{E}_{\tau_N}\left[\textstyle\sum_{i=0}^{N} \gamma^i f(s_i)\right] = \\ &= \textstyle\int_{s_0, s_1, \ldots, s_N} \left(\prod_{i=0}^{N-1} p(s_{i+1} \mid s_i)\right) p_0(s_0) \left(\sum_{i=0}^{N} \gamma^i f(s_i)\right) ds_0 ds_1 \ldots ds_N = \\ &= \textstyle\sum_{i=0}^{N} \int_{s_0, s_1, \ldots, s_N} \left(p_0(s_0) \prod_{i=0}^{N-1} p(s_{i+1} \mid s_i)\right) \gamma^i f(s_i) ds_0 ds_1 \ldots ds_N = \\ &= \textstyle\sum_{i=0}^{N} \int_s p(s \mid t=i) \gamma^i f(s) ds \end{aligned}$$

for any function f.

Definition 3 (Expectation with respect to infinte trajectory)**.** *For any bounded function f, we have*

$$\mathbb{E}_\tau\left[\sum_{i=0}^{\infty} \gamma^i f(s_i)\right] \triangleq \lim_{N \to \infty} \mathbb{E}_{\tau_N}\left[\sum_{i=0}^{N} \gamma^i f(s_i)\right].$$

Here, the sum on the left-hand side is part of the symbol being defined.

Observation 2 (Property of expectation with respect to infinte trajectory)**.**

$$\begin{aligned} \mathbb{E}_\tau\left[\textstyle\sum_{i=0}^{\infty} \gamma^i f(s_i)\right] &= \textstyle\lim_{N \to \infty} \mathbb{E}_{\tau_N}\left[\sum_{i=0}^{N} \gamma^i f(s_i)\right] = \\ &= \textstyle\lim_{N \to \infty} \sum_{i=0}^{N} \int_s p(s \mid t=i) \gamma^i f(s) ds = \\ &= \sum_{i=0}^{\infty} \int_s dp(s \mid t=i) \gamma^i f(s) \end{aligned}$$

for any bounded function f.

Definition 4 (Discounted-ergodic occupancy measure ρ)**.**

$$\rho(s) = \sum_{i=0}^{\infty} \gamma^i p(s \mid t=i)$$

The measure ρ is not normalised in general. Intuitively, it can be thought of as 'marginalising out' the time in the system dynamics.

Lemma 4 (Discounted-ergodic property)**.** *For any bounded function f:*

$$\int_s \rho(s) f(s) = \mathbb{E}_\tau\left[\sum_{i=0}^{\infty} \gamma^i f(s_i)\right].$$

Proof.

$$\mathbb{E}_\tau\left[\sum_{i=0}^{\infty} \gamma^i f(s_i)\right] = \sum_{i=0}^{\infty} \gamma^i \int_s p(s \mid t=i) f(s) ds = \int_s \underbrace{\left[\sum_{i=0}^{\infty} \gamma^i p(s \mid t=i)\right]}_{\rho(s)} f(s) ds$$

Here, the first equality follows from Observation 2. □

This property is useful since the expression on the left can be easily manipulated while the expression on the right can be estimated from samples using Monte Carlo.

Lemma 5 (Generalised eigenfunction property)**.** *For any bounded function f:*

$$\gamma \int_s d\rho(s) \int_{s'} dp(s' \mid s) f(s') = \left(\int_s d\rho(s) f(s) \right) - \left(\int_s dp_0(s) f(s) \right)$$

Proof.

$$\begin{aligned}
\gamma \textstyle\int_s d\rho(s) \int_{s'} dp(s' \mid s) f(s') &= \gamma \textstyle\sum_{i=0}^{\infty} \gamma^i \int_{s,s'} p(s \mid t=i) p(s' \mid s) f(s') ds ds' = \\
&= \textstyle\sum_{i=0}^{\infty} \gamma^{i+1} \int_{s'} dp(s' \mid t=i+1) f(s') \\
&= \textstyle\sum_{i=1}^{\infty} \gamma^{i} \int_{s'} dp(s' \mid t=i) f(s') \\
&= \left(\textstyle\sum_{i=0}^{\infty} \gamma^{i} \int_{s'} dp(s' \mid t=i) f(s') \right) - \left(\int_s dp_0(s) f(s) \right) \\
&= \left(\textstyle\int_s d\rho(s) f(s) \right) - \left(\int_s dp_0(s) f(s) \right)
\end{aligned}$$

Here, the first equality follows form definition 4, the second one from definition 1. The last equality follows again from definition 4. □

Definition 5 (Markov Reward Process)**.** *A Markov Reward Process is a tuple (p, p_0, R, γ), where $p(s'|s)$ is a transition kernel, p_0 is the distribution over initial states, $R(\cdot|s)$ is a reward distribution conditioned on the state and γ is the discount constant.*

An MRP can be thought of as an MDP with a fixed policy and dynamics given by marginalising out the actions $p_\pi(s' \mid s) = \int_a d\pi(a \mid s) p(s' \mid a, s)$. Since this paper considers the case of one policy, we abuse notation slightly by using the same symbol τ to denote trajectories including actions, i.e. $(s_0, a_0, r_0, s_1, a_1, r_1, \dots)$ and without them $(s_0, r_0, s_1, r_1, \dots)$.

Lemma 6 (Second Moment Bellman Equation)**.** *Consider a Markov Reward Process (p, p_0, X, γ) where $p(s' \mid s)$ is a Markov process and $X(\cdot \mid s)$ is some probability density function*[6]*. Denote the value function of the MRP as V. Denote the second moment function S as*

$$S(s) = \mathbb{E}_\tau \left[\left(\sum_{t=0}^{\infty} \gamma^t x_t \right)^2 \middle| s_0 = s \right] \quad x_t \sim X(\cdot \mid s_t).$$

Then S is the value function of the MRP: (p, p_0, u, γ^2), where $u(s)$ is a deterministic random variable given by

$$u(s) = \mathbb{V}_{X(x|s)}[x] + \left(\mathbb{E}_{X(x|s)}[x] \right)^2 + 2\gamma \mathbb{E}_{X(x|s)}[x] \, \mathbb{E}_{p(s'|s)}[V(s')].$$

Proof.

$$\begin{aligned}
S(s) &= \mathbb{E}_\tau \left[\left(x_0 + \textstyle\sum_{t=1}^{\infty} \gamma^t x_t \right)^2 \middle| s_0 = s \right] \\
&= \mathbb{E}_\tau \left[x_0^2 + 2x_0 \left(\textstyle\sum_{t=1}^{\infty} \gamma^t x_t \right) + \left(\textstyle\sum_{t=1}^{\infty} \gamma^t x_t \right)^2 \middle| s_0 = s \right] \\
&= \underbrace{\mathbb{E}_\tau \left[x_0^2 \mid s_0 = s \right] + \mathbb{E}_\tau \left[2x_0 \left(\textstyle\sum_{t=1}^{\infty} \gamma^t x_t \right) \mid s_0 = s \right]}_{u(s)} + \underbrace{\mathbb{E}_\tau \left[\left(\textstyle\sum_{t=1}^{\infty} \gamma^t x_t \right)^2 \middle| s_0 = s \right]}_{\gamma^2 \mathbb{E}_{p(s'|s)}[S(s')]}
\end{aligned}$$

This is exactly the Bellman equation of the MRP (p, p_0, u, γ^2). The theorem follows since the Bellman equation uniquely determines the value function. □

Observation 3 (Dominated Value Functions)**.** *Consider two Markov Reward Processes (p, p_0, X_1, γ) and (p, p_0, X_2, γ), where $p(s' \mid s)$ is a Markov process (common to both MRPs) and $X_1(s)$, $X_2(s)$ are some deterministic random variables meeting the condition $X_1(s) \leq X_2(s)$ for every s. Then the value functions V_1 and V_2 of the respective MRPs satisfy $V_1(s) \leq V_2(s)$ for every s. Moreover, if we have that $X_1(s) < X_2(s)$ for all states, then the inequality between value functions is strict.*

Proof. Follows trivially by expanding the value function as a series and comparing series elementwise. □

We now move our attention to prove the Gaussian Policy Gradients lemma.

[6] Note that while X occupies a place in the definition of the MRP usually called 'reward distribution', we are using the symbol X, not R since we shall apply the lemma to Xes which are constructions distinct from the reward of the MDP we are solving.

Lemma 1 (Gaussian Policy Gradients). *If the policy is Gaussian, i.e.* $\pi(\cdot|s) \sim \mathcal{N}(\mu_s, \Sigma_s)$ *with* μ_s *and* $\Sigma_s^{1/2}$ *parametrised by* θ*, where* $\Sigma_s^{1/2}$ *is symmetric and* $\Sigma_s^{1/2}\Sigma_s^{1/2} = \Sigma_s$ *and the critic is of the form* $Q(a, s) = a^\top A(s)a + a^\top B(s) + const$ *where* $A(s)$ *is symmetric for every* s*, then* $I_\pi^Q(s) = I^Q_{\pi(s),\mu_s} + I^Q_{\pi(s),\Sigma_s^{1/2}}$*, where the mean and covariance components are given by* $I^Q_{\pi(s),\mu_s} = (\nabla\mu_s)(2A(s)\mu + B(s))$ *and* $I^Q_{\pi(s),\Sigma_s^{1/2}} = (\nabla\Sigma_s^{1/2})2A(s)\Sigma_s^{1/2}$.

Proof. First, we observe that the critic Q defined in the statement of the lemma does not depend on the policy parameters θ. This is because Q is an approximation to the Q-function maintained by the algorithm as opposed to the true Q-function, which is defined with respect to the policy and does depend on it.

We can hence move the differentiation outside of the integral, as follows.

$$I_\pi^Q(s) = \nabla \int_a \pi(a|s)Q(a, s)da = \nabla\mathbb{E}_\pi\left[Q(a, s)\right]$$

We now expand the expectation using a known expression for the expectation of a a quadratic form:

$$\mathbb{E}_\pi\left[Q(a, s)\right] = \text{trace}(A(s)\Sigma) + \mu^\top A(s)\mu + B(s)^\top\mu.$$

This gives way to the following derivatives.

$$\begin{aligned}\nabla_{\Sigma^{1/2}}\mathbb{E}_\pi\left[Q(a, s)\right] &= \nabla_{\Sigma^{1/2}}(\text{trace}(A(s)\Sigma) + \mu^\top A(s)\mu + B(s)^\top\mu) = 2A(s)\Sigma^{1/2}\\ \nabla_\mu\mathbb{E}_\pi\left[Q(a, s)\right] &= \nabla_\mu(\text{trace}(A(s)\Sigma) + \mu^\top A(s)\mu + B(s)^\top\mu) = 2A(s)\mu + B(s)\end{aligned}$$

.

We now obtain the result by applying chain rule.

$$I_\pi^Q(s) = I^Q_{\pi(s),\mu_s} + I^Q_{\pi(s),\Sigma_s^{1/2}} = (\nabla\mu)(2A(s)\mu + B(s)) + (\nabla\Sigma^{1/2})(2A(s)\Sigma^{1/2})$$

□

Lemma 3. *If for all* $s \in S$*, the random variable* $\nabla\log\pi(a \mid s)\hat{Q}(s, a)$ *where* $a \sim \pi(a|s)$ *has nonzero variance, then*

$$\begin{aligned}&\mathbb{V}_\tau\left[\textstyle\sum_{t=0}^\infty \gamma^t\nabla\log\pi(a_t \mid s_t)(\hat{Q}(s_t, a_t) + b(s_t))\quad\right] >\\ &\mathbb{V}_\tau\left[\textstyle\sum_{t=0}^\infty \gamma^t I_\pi^{\hat{Q}}(s_t)\right].\end{aligned}$$

Proof. Both random variables have the same mean so we need only show that:

$$\begin{aligned}&\mathbb{E}_\tau\left[\left(\textstyle\sum_{t=0}^\infty \gamma^t\nabla\log\pi(a_t \mid s_t)(\hat{Q}(s_t, a_t) + b(s_t))\right)^2\right] >\\ &\mathbb{E}_\tau\left[\textstyle\sum_{t=0}^\infty \left(\gamma^t I_\pi^{\hat{Q}}(s_t)\right)^2\right].\end{aligned}$$

We start by applying Lemma 6 to the lefthand side and setting $X = X_1(s_t) = \gamma^t\nabla\log\pi(a_t \mid s_t)(\hat{Q}(s_t, a_t) + b(s_t))$ where $a_t \sim \pi(a_t|s_t)$. This shows that $\mathbb{E}_\tau\left[\left(\sum_{t=0}^\infty \gamma^t\nabla\log\pi(a_t \mid s_t)(\hat{Q}(s_t, a_t) + b(s_t))\right)^2\right]$ is the total return of the MRP (p, p_0, u_1, γ^2), where

$$u_1 = \mathbb{V}_{X_1(x|s)}\left[x\right] + \left(\mathbb{E}_{X_1(x|s)}\left[x\right]\right)^2 + 2\gamma\mathbb{E}_{X_1(x|s)}\left[x\right]\mathbb{E}_{p(s'|s)}\left[V(s')\right].$$

Likewise, applying Lemma 6 again to the righthand side, instantiating X as a deterministic random variable $X_2(s_t) = I_\pi^{\hat{Q}}(s_t)$, we have that $\mathbb{E}_\tau\left[\sum_{t=0}^\infty \left(\gamma^t I_\pi^{\hat{Q}}(s_t)\right)^2\right]$ is the total return of the MRP (p, p_0, u_2, γ^2), where

$$u_2 = \left(\mathbb{E}_{X_2(x|s)}\left[x\right]\right)^2 + 2\gamma\mathbb{E}_{X_2(x|s)}\left[x\right]\mathbb{E}_{p(s'|s)}\left[V(s')\right].$$

Note that $\mathbb{E}_{X_1(x|s)}\left[x\right] = \mathbb{E}_{X_2(x|s)}\left[x\right]$ and therefore $u_1 \geq u_2$. Furthermore, by assumption of the lemma, the inequality is strict. The lemma then follows by applying Observation 3. □

For convenience, Lemma 3 also assumes infinite length trajectories. However, this is not a practical limitation since all policy gradient methods implicitly assume trajectories are long enough to be modelled as infinite. Furthermore, a finite trajectory variant also holds, though the proof is messier.

Remarks on the covariance limit

When we obtain e^H as the limiting covariance matrix in Lemma 2 of the main paper, there is a slight modelling difficulty: is it justified to use the learning rate of $\frac{1}{n}$, which diminished in the length of the trajectory, as opposed to a small finite number? We observe that the problem of choosing step sizes is, in general, not specific to our method since all policy gradient methods rely on stochastic optimisation and hence work with a diminishing learning rate of some sort. We do note; however, that the step size we use, which is $\frac{1}{n}$ for every point in the trajectory, is different from the step size typically used with Robbins-Monro procedure, which is different at each time step. This means that the sum of our step sizes is finite while the sum of the Robbins-Monro step-sizes diverges. Hence our choice of step size does not give the guarantees typically associated with stochastic optimisation. We use the step sequence since it serves as a useful intermediate stage between simply taking *one* PG step of equation (11) and using a finite step-sizes, which would mean that the covariance would converge either to zero or diverge to infinity.