PPO Pt. 1: Actor-Critic Methods

Catherine Ji

March 18, 2025

1 Summary

- Bias-Variance tradeoff: REINFORCE is high-variance but bias-free, Q-Learning is high bias.
- To reduce variance in REINFORCE, we can use a critic-estimate $Q^\pi(s_t, a_t)$ and advantage estimation.
- Importance sampling allows us to use a replay buffer in policy gradient methods – making our Vanilla AC now off-policy!
- PPO combines all of these methods to reduce variance in policy gradient methods!

2 Introduction

As a refresh, what algorithms have we learned in the class so far?

1. REINFORCE: We learned about the REINFORCE algorithm, which is a policy gradient method. Given a bunch of sampled trajectories from policy π, we calculate $\text{grad}_\theta\, \mathbb{E}_{\tau\sim\pi(\tau)}[R(\tau)]$ to push policy parameters in the direction of higher trajectory returns. Then, we must resample trajectories from this new policy, and repeat – REINFORCE is an on-policy method. Doing log-likelihood trick and noting that only future returns should influence action at timestep t, recall that we can rewrite REINFORCE as

$$\text{grad}_\theta\, \mathbb{E}_{\tau\sim\pi(\cdot)}[R(\tau)] = \mathbb{E}_{\tau\sim\pi_\theta(\cdot)}[\sum_{t'=t}^{T} \gamma^{t'} r(s_{t'}, a_{t'}) \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t)]$$

$$\theta_{i+1} \leftarrow \theta_i + \xi \mathbb{E}_{\tau\sim\pi_\theta(\cdot)}[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t) \sum_{t'=t}^{T} \gamma^{t'} r(s_{t'}, a_{t'})].$$

   We can also do the expectation over the trajectory-induced probability distribution over states, with γ weightings to "simulate" absorbing states (probability of "death" is $1-\gamma$ per timestep). Below, we relabel $\sum_{t'=t}^{T} \gamma^{t'-t} r(s_{t'}, a_{t'})$ as $G_t = G(s_t, a_t)$).

   Effectively, we're saying to forget about iterating along the trajectories – if we follow the policy for a ton of samples, we will effectively sample from a state distribution *induced* by the policy called the **dicounted state occupancy measure**, and can abstract away annoyances of computing policy-gradient transition-by-transition:

$$\theta_{i+1} \leftarrow \theta_i + \xi \mathbb{E}_{s_t\sim P_\gamma^\pi(\cdot), a_t\sim\pi(\cdot\mid s_t)}[\nabla_\theta \log \pi_\theta(a \mid s) G(s, a)].$$

Explicitly, the **discounted state occupancy measure** is as follows (prefactor is just for normalization)

$$P_\gamma^\pi(s) \triangleq \frac{1-\gamma}{1-\gamma^{T+1}} \sum_{t=0}^{T} \gamma^t p^\pi(s_t = s),$$

where we define (written out explicitly for clarity):

$$\begin{aligned}
p^\pi(s_t = s) &= (\text{Probability that we're in state s at time t by following policy from start distribution}) \\
&= \sum_{(s_0, a_0, \cdots s_{t-1}, a_{t-1}) \in (\mathcal{S}\times\mathcal{A})^t} p^\pi(s_t = s \mid s_{t-1}, a_{t-1}, s_{t-2}, a_{t-2}, \cdots) p^\pi(s_{t-1}, a_{t-1}, s_{t-2}, a_{t-2}, \cdots) \\
&= \sum_{(s_0, a_0, \cdots s_{t-1}, a_{t-1})} p^\pi(s_t = s \mid s_{t-1}, a_{t-1}) p^\pi(s_{t-1}, a_{t-1}, s_{t-2}, a_{t-2}, \cdots) \\
&= \sum_{(s_0, a_0, \cdots s_{t-1}, a_{t-1})} p^\pi(s_t = s \mid s_{t-1}, a_{t-1}) p^\pi(s_{t-1}, a_{t-1} \mid s_{t-2}, a_{t-2}) p^\pi(s_{t-2}, a_{t-2}, \cdots) \\
&= \sum_{(s_0, a_0, \cdots s_{t-1}, a_{t-1})} p^\pi(s_t = s \mid s_{t-1}, a_{t-1}) p(s_0) \prod_{i=0}^{t-2} p^\pi(s_{i+1}, a_{i+1} \mid s_i, a_i) \\
&= \sum_{(s_0, a_0, \cdots s_{t-1}, a_{t-1})} p^\pi(s_t = s \mid s_{t-1}, a_{t-1}) p(s_0) \prod_{i=0}^{t-2} p(s_{i+1} \mid s_i, a_i) \pi(a_{i+1} \mid s_{i+1})
\end{aligned}$$

A great question fom lecture today ("I thought dynamics are Markovian, so why do we use this object?"): yes, we can indeed write/break down P_γ^π in terms of the transitions, policies, etc. However, this is all quite taxing to write out as you can see – it's also easier to think about the discounted state occupancy measure as a way to sample states from the policy-induced trajectory distribution *without* necessarily needing to (mentally) iterate timestep-by-timestep through individual, sampled trajectories.

Side note: In some RL methods (i.e. GCRL), the critic is actually the (goal-conditioned) state occupancy measure! So there are settings where we are directly estimating this object. However, in your homework, you'll be directly iterating through trajectories.

The key idea here is that summing rewards along the discounted trajectory is equivalent to sampling from a discounted occupancy measure. See here for more details. Some major cons of REINFORCE: high variance and on-policy.

2. Q-Learning: We learned about Q-Learning. Q-Learning is an off-policy TD-learning method that, given some set of transitions, applies a (weighted) Bellman Optimality update. Written out, our update is

$$Q(s, a) \leftarrow (1-\alpha) Q(s, a) + \alpha[r(s, a) + \gamma \max_{a'} Q(s', a')].$$

When we do DQN, we use a neural network to approximate the Q-function, and instead update the parameters of Q based on the TD-error:

$$\theta_{i+1} \leftarrow \theta_i - \xi \, \text{grad}_\theta \, \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}[(r + \gamma \max_{a'} Q(s', a'; \theta_i) - Q(s, a; \theta_i))^2].$$

Some major cons of Q-Learning: can't deal w/ high-dimensional and/or continuous action spaces (directly maximizing over high-dimensional, continuous action space is rough), high bias, and unclear how to get good exploration if replay buffer collected from epsilon greedy of estimates Qs...

There are quite a few shortcomings of these methods that boil down to the **bias-variance tradeoff**.

Definition 1 (Bias-Variance Tradeoff)
The bias-variance tradeoff is a general statistical concept to capture generalization error: how well a model will perform on unseen (assume in-distribution) test-data. **Bias error** comes from missing salient features in your underlying data, such that your model, in expectation, will not converge to the optimal model. **Variance error** comes from having models overparameterized relative to the noise in your data, so your fitted model will have high variance (i.e. if you fit a model N times, you may get N very different models that, in expectation of many, many samples, are correct).

In general, we'd like to keep both bias and variance low.

2.1 Bias-Variance tradeoff in REINFORCE and Q-Learning

Q: What does Bias-Variance tradeoff look like for REINFORCE and Q-Learning updates?

A: (REINFORCE) REINFORCE, as we've learned it, is high-variance but bias-free. Consider the $R(\tau)$ term in the gradient: this term may have an incredible amount of variance! However, we also know that, in expectation of an infinite number of trajectories, the gradient will converge to $\text{grad}_\theta \, \mathbb{E}_{\tau\sim\pi(\cdot)}[R(\tau)]$ – thus, the bias here is zero.

What else do we love about REINFORCE? It can handle high-dimensional action spaces and continuous action spaces! This is because we're directly optimizing the policy, not the Q-function.

A: (Q-Learning) Q-Learning is high bias. Why is this? We compute the TD target using the bootstrapped value – the maximization over the Q-function with function-approximation will always introduce (overestimation) bias! For example, assume function approximation errors on $Q(s', a')$ in the Bellman Update are normally distributed (and $(s', a') \neq (s, a)$). By virtue of taking the max epoch, the error will always be positive – thus, this is *bias*.

Q: When would we want to use REINFORCE? When would we want to use Q-Learning?

2.2 Side note: Double Deep Q Networks (DDQN)

The goal of DDQN is to reduce the bias in Q-Learning by decorrelating the action a' selected by the max operator, and the Q-value used to evaluate the target. Maintaining the correlations between the two, by deriving both from the same Q-value, can lead to overestimation bias: even if the underlying error in, say, the action values $(Q(s', a'))$ are uniformly distributed, the target will still be overestimated!

Thus, we use two separate networks: one to select the action, and one to evaluate the target, and randomly choosing which does which at each iteration of the algorithm.

Concretely, the update is as follows:

$$\theta_{i+1} \leftarrow \theta_i - \xi \, \text{grad}_\theta \, \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}[(r + \gamma Q(s', \arg\max_{a'} Q(s', a'; \theta_i); \theta_i') - Q(s, a; \theta_i'))^2].$$

where θ' and θ parameterize the two networks.

Q: Do we actually know if this reduces bias?

A: Original DDQN authors show that double Q-learning can reduce overestimation bias with a specific set of assumptions. It's hard to prove bounds in the general DDQN case, but has been shown to work well empirically.

3 Methods to improve REINFORCE

Let's focus on improving REINFORCE to set up PPO. We want to deal with the high-variance nature of $R(\tau)$... is there anything we can use as a proxy? The answer is... to replace the reward-to-go with a critic-estimate $Q^\pi(s_t, a_t)$! Note that they're equivalent in expectation, but since we're function-approximating Q, we will run into maximization bias.

Rewriting the gradient,

$$\mathrm{grad}_\theta\, \mathbb{E}_{\tau\sim\pi(\cdot)}[R(\tau)] = \mathbb{E}_{s_t\sim P_\gamma^\pi(\cdot),a_t\sim\pi(\cdot|s_t)}[\sum_{t=0}^{T} Q^\pi(s_t, a_t)(\nabla_\theta \log \pi_\theta(a_t \mid s_t))].$$

3.1 Advantage Estimation

Beyond using a critic-estimate Q value in place of sampled $R(\tau)$, a key way to reduce variance in the REINFORCE gradient is to use **advantage estimation**. The idea is to subtract a baseline from the reward to get the advantage. Explicitly,

Definition 2 (Advantage)
The advantage of a state-action pair (s, a) is defined as

$$A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s).$$

Let's take the critic update in REINFORCE and replace $Q^\pi(s_t, a_t)$ with $A^\pi(s_t, a_t)$. For less notational headache, let's just consider an arbitrary STATE-BASED baseline (no action!) $b(s_t)$:

$$\begin{aligned}
\mathrm{grad}_\theta\, \mathbb{E}_{\tau\sim\pi(\cdot)}[R(\tau)] &= \mathbb{E}_{\tau\sim\pi_\theta(\cdot)}[\sum_{t=0}^{T} Q^\pi(s_t, a_t)\nabla_\theta \log \pi_\theta(a_t \mid s_t)] \\
&= \mathbb{E}_{\tau\sim\pi_\theta(\cdot)}[\sum_{t=0}^{T}(Q^\pi(s_t, a_t) - b(s_t))\nabla_\theta \log \pi_\theta(a_t \mid s_t)] \\
&= \mathbb{E}_{\tau\sim\pi_\theta(\cdot)}[\sum_{t=0}^{T} Q^\pi(s_t, a_t)\nabla_\theta \log \pi_\theta(a_t \mid s_t)] - \mathbb{E}_{\tau\sim\pi_\theta(\cdot)}[\sum_{t=0}^{T} b(s_t)\nabla_\theta \log \pi_\theta(a_t \mid s_t)] \\
&= \mathbb{E}_{\tau\sim\pi_\theta(\cdot)}[\sum_{t=0}^{T} Q^\pi(s_t, a_t)\nabla_\theta \log \pi_\theta(a_t \mid s_t)] - \mathbb{E}_{s\sim P_\gamma^{\pi_\theta}(\cdot),a\sim\pi_\theta(\cdot|s)}[b(s)\nabla_\theta \log \pi_\theta(a \mid s)].
\end{aligned}$$

where we can rewrite the expectation in terms of the state-occupancy measure (see Pg. 2 for explicit expression). Now, we show the second term equals 0:

$$\begin{aligned}
\mathbb{E}_{s\sim P_\gamma^{\pi_\theta}(\cdot),a\sim\pi_\theta(\cdot|s)}[b(s)\nabla_\theta \log \pi_\theta(a \mid s)] &= \mathbb{E}_{s\sim P_\gamma^{\pi_\theta}(\cdot)}[b(s)\mathbb{E}_{a\sim\pi_\theta(\cdot|s)}[\nabla_\theta \log \pi_\theta(a \mid s)]] \\
&= 0.
\end{aligned}$$

Thus, we can replace the $Q^\pi(s_t, a_t)$ term in the REINFORCE gradient with the advantage $A^\pi(s_t, a_t)$!

How do we know this reduces the variance of the gradient beyond inspection? We can do an explicit calculation.

3.2 Variance reduction proof

NOTE: This is not super important. But just in case folks are curious. We can show that variance COMPUTED WITH RESPECT TO DISTRIBUTION $\pi_\theta(a_t \mid s_t)$ of random varialbe g reduces for a GIVEN t and $s_t \sim P_t^{\pi_\theta}(\cdot)$ term (as used in the last derivation):

$$g = (\nabla_\theta \log \pi_\theta(a \mid s))\, Q(s, a).$$

Let's consider arbitrary baseline $b(s)$. The estimator becomes:

$$g_b = (\nabla_\theta \log \pi_\theta(a \mid s)) \Big(Q(s, a) - b(s)\Big).$$

For notational clarity (a.k.a. we're taking expectation w.r.t. action a with assumed s), define

$$d(a) \triangleq \nabla_\theta \log \pi_\theta(a \mid s) \quad \text{and} \quad Q \triangleq Q(s, a).$$

and use $\mathbb{E}_a$ as shorthand for $\mathbb{E}_{a \sim \pi_\theta(\cdot \mid s)}$ Then the baseline-subtracted estimator is

$$Z = d(a)\,(Q - b).$$

3.2.1 Set gradient to 0

We wish to choose b to minimize the variance of Z:

$$f_{\mathrm{Var}}(b) = \mathbb{E}_a\Big[\|d(a)\|^2\,(Q - b)^2\Big] - \mathbb{E}\Big[d(a)\,(Q - b)\Big]^2$$

The optimal b minimizes $f_{\mathrm{Var}}(b)$. Let's differentiate $f_{\mathrm{Var}}(b)$ with respect to b. Note that the second term loses dependence on b from the score-function trick! Thus,

$$\begin{aligned}\frac{d}{db} f(b) &= \frac{d}{db}\,(\mathbb{E}\Big[\|d(a)\|^2\,(Q - b)^2\Big] \\ &= \mathbb{E}\Big[\|d(a)\|^2\,\frac{d}{db}\,(Q - b)^2\Big] \\ &= -2\,\mathbb{E}\Big[\|d(a)\|^2(Q - b)\Big].\end{aligned}$$

Setting the derivative equal to zero for minimization:

$$\mathbb{E}\Big[\|d(a)\|^2(Q - b)\Big] = 0.$$

Expanding, we have:

$$\mathbb{E}\Big[\|d(a)\|^2 Q\Big] - b\,\mathbb{E}\Big[\|d(a)\|^2\Big] = 0.$$

Solving for b gives:

$$b = \frac{\mathbb{E}\Big[\|d(a)\|^2 Q\Big]}{\mathbb{E}\Big[\|d(a)\|^2\Big]}.$$

Recalling that $Q = Q(s, a)$ and $d(a) = \nabla_\theta \log \pi_\theta(a \mid s)$, we can write our final expression:

$$b^*(s) = \frac{\mathbb{E}_{a \sim \pi_\theta(\cdot \mid s)}\Big[\|\nabla_\theta \log \pi_\theta(a \mid s)\|^2\, Q(s, a)\Big]}{\mathbb{E}_{a \sim \pi_\theta(\cdot \mid s)}\Big[\|\nabla_\theta \log \pi_\theta(a \mid s)\|^2\Big]}.$$

3.2.2 Interpretation

So... our optimal baseline doesn't look like the advantage function... there's this extra scaling by the norm of the gradient of the log-probability. Why don't we just use this for our baseline?

Let's see the additional variance we incur by using the advantage function as a baseline. The variance of the optimal baseline-modified estimator is:

$$\mathbb{E}_a\left[\|d(a)\|^2 (Q - b^*)^2\right].$$

and the variance of the advantage-modified estimator is:

$$\mathbb{E}_a\left[\|d(a)\|^2 (Q - b)^2\right].$$

After some calculation that I omit here, we find that the difference $f_{\text{Var}}(b) - f_{\text{Var}}(b^*)$ ends up being:

$$\frac{(\text{Cov}_a\left[\|d(a)\|^2 Q\right])^2}{\mathbb{E}[\|d(a)\|^2])}.$$

Can we bound this? Maybe... but it's not super clear because we don't know the relationship between θ and π_θ: the functional relationship could be totally arbitrary. However, there's likely some information geometric way to bound this term when using trust regions, which is beyond the scope of these lecture notes.

3.3 Summarizing Advantage Estimation

We have turned our original update

$$\theta_{i+1} \leftarrow \theta_i + \xi \mathbb{E}_{\tau\sim\pi_\theta(\cdot)}[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t) \sum_{t'=t}^{T} \gamma^{t'-t} r(s_{t'}, a_{t'})].$$

into a variance-reduced update

$$\theta_{i+1} \leftarrow \theta_i + \xi \mathbb{E}_{\tau\sim\pi_\theta(\cdot)}[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t) A^\pi(s_{t'}, a_{t'})].$$

But wait... there has to be some catch based on the bias-variance tradeoff! The answer is that even though this update is great if we know A^π perfectly, in AC methods, we will always have some amount of bias in our critic, which now propagates to our policy gradient estimate!!!

(Skip during lecture.) We can use Generalized Advantage Estimation (GAE) that uses hyperparameter λ that balances bias and variance:

Definition 3 (Generalized Advantage Estimation (GAE))
The Generalized Advantage Estimation (GAE) estimates advantage, where hyperparameter λ tunes the bias-variance tradeoff. The GAE is defined as

$$A^\pi(s_t, a_t) = \sum_{t'=t}^{T} (\gamma\lambda)^{t'-t} \delta^\pi_{t'},$$

where $\delta^\pi_{t'} = r(s_{t'}, a_{t'}) + \gamma V^\pi(s_{t'+1}) - V^\pi(s_{t'})$ is the TD-error. In essence, it is balancing TD estimates of advantage and monte carlo estimates of advantage. Full explanation of this object is here.

3.4 Another method to reduce variance: importance sampling

Another method to reduce the generalization error from variance in policy gradient methods is to use importance weighting – this is a method to weight data collected from previous policies to update our current policy, and thus use a replay buffer! Thus, effectively, we are converting our on-policy REINFORCE (and variants w/ advantage estimation, other tricks above...) to an "off-policy" method.

Definition 4 (Importance Sampling)
Importance sampling estimates the expectation of a function under one distribution w/ samples from *another distribution*. Given $p(x)$ and $q(x)$, the expectation of a function $f(x)$ under $p(x)$ can be estimated using samples from $q(x)$ as

$$\mathbb{E}_{x\sim p(x)}[f(x)] = \mathbb{E}_{x\sim q(x)}\left[\frac{p(x)}{q(x)}f(x)\right]$$

which in the limit of infinite samples, will converge to the true expectation.

In our policy gradient setting, our distributions $p(x) \to \pi_\theta(x)$ and $q(x) \to \pi_{\theta_{old}}(x)$, where $\pi_{\theta_{old}}$ is the policy that generated the data in our replay buffer. Thus, rewriting our REINFORCE update with importance sampling and using the discounted state occupancy measure, we have:

$$\text{grad}_\theta\, \mathbb{E}_{\tau\sim\pi(\cdot)}[R(\tau)] = \mathbb{E}_{s\sim P_\gamma^{\pi_{\text{old}}}(\cdot)}\mathbb{E}_{a\sim\pi_{\text{old}}(\cdot|s)}\left[\frac{\pi_\theta(a\mid s)}{\pi_{\theta_{old}}(a\mid s)}A^{\pi_{\theta_{old}}}(s,a)\nabla_\theta \log\pi_\theta(a\mid s)\right]$$

so we can now use a replay buffer to update our policy! You may say: hey, that's kind of fishy! We made all these bias-free arguments to explain the advantage function... but, there, we assumed that we assume behavior and target policies are the same (on-policy). Note that this importance sampling **is not an unbiased estimator of the policy gradient**! Namely, we don't know how $\pi_{\theta_{\text{old}}}$ relates to π_θ and, in fact, may actively want these policies to be different (off-policy method!) But how big of an update do we want to make? Tune in for trust regions...

4 Review of Actor-Critic Methods

Recall from last lecture: Actor-Critic methods combine policy gradient and value-based methods. Namely, they consist of two components: the **actor** (policy) and the **critic** (value function). In the vanilla (on-policy) algorithm,

Algorithm 1 Most Vanilla Actor-Critic (on-policy)

```
Initialize $\theta$ and $w$ arbitrarily, which parameterize $\pi$ and $Q$
for each episode do
    Initialize $s$
    for each step of episode do
        Collect on-policy transitions: $a$ from $\pi_\theta(\cdot \mid s)$, observe $r, s'$, sample $a'$ from $\pi_\theta(\cdot \mid s')$
        Update critic (TD-error): $w \leftarrow w - \xi\, \text{grad}_w (r + \gamma Q(s', a'; w) - Q(s, a; w))^2$
        Update actor: $\theta \leftarrow \theta + \xi Q(s, a; w)(\text{grad}_\theta \log \pi_\theta(a \mid s))$
        Go to next state: $s \leftarrow s'$
    end for
end for
```

Based on this Vanilla on-policy algorithm, your first reaction may be... let's make this off-policy so we can use a replay buffer and exploration in our action-selection policy! Also, let's use advantage functions to reduce variance without affecting bias!

Q: How can we make this off-policy? A: We can use importance sampling!

Q: How can we use advantage functions to reduce variance? A: We can replace the $Q(s, a)$ term in the policy gradient with the advantage function $A^{\pi}(s, a)$!

Q: What other improvements can we make to this algorithm? A: We can use GAE. [other suggestions]. We also can add on constraints to our policy update in **probability space**. PPO does this in two ways: clipping the policy update and adding a KL-divergence penalty to the policy update. General intuition: when we tune the weights willy nilly, our guaranteed bounds on policy deviation live in Euclidian space (i.e. Taylor expansion)... but we perhaps want our policy deviation to live in probability space...

A way to measure probability deviation is KL-divergence.

Definition 5 (KL-Divergence recap)
The KL-divergence between two distributions p and q is defined as

$$D_{KL}(p \mid q) = \mathbb{E}_{x\sim p(x)}[\log \frac{p(x)}{q(x)}].$$

It captures the expected number of extra bits needed to encode data from p using a code optimized for q. Aka, it is an asymmetric "distance" between two distributions (warning: does not satisfy triangle inequality).

More on this next time...

5 Summary

Today, we talked about bias-variance tradeoff. We explored various methods to reduce variance in policy gradient methods, including using advantage functions, importance sampling, and GAE.

We also discussed the Actor-Critic method, and how we can improve using n Next time, we will discuss Proximal Policy Optimization (PPO) with these information-geometry guarantees.