Policy Gradient Methods

Jubayer Ibn Hamid

In this note, I will cover the basics of vanilla policy gradient methods and trust regional policy optimisation.

1 Introduction

In policy gradient methods, we directly parametrise the policy to be $\pi_\theta(a|s) = \mathbb{P}_\theta(a|s)$. Our goal is then to find π_θ that maximises returns.

For policy-based methods, we have no value function but only a learned policy. For actor-critic methods, we have both a learned value function and a learned policy.

Policy gradient methods are useful for generating stochastic policies but they are often hard to evaluate and can be high variance.

2 Vanilla Policy Gradient Methods

In this note, we will cover episodic tasks. Let $V(\theta) = V(s_0, \theta)$, where s_0 is the starting state (which we consider to be fixed for simplicity) and the dependency on θ specifies that the value depends on the policy π_θ.

Now, we want to maximize $V(\theta)$:

$$\begin{aligned} V(\theta) &= V(s_0, \theta) \\ &= \sum_a \pi_\theta(a|s_0) Q(s_0, a; \theta) \\ &= \sum_\tau P_\theta(\tau) R(\tau) \end{aligned}$$

where $Q(s_0, a; \theta)$ is the expected reward attained under the policy π_θ after taking action a from state s_0. $P_\theta(\tau)$ is the probability of trajectory $\tau = (s_0, a_0, r_0, s_1, ..., s_{T-1}, a_{T-1}, r_{T-1}, s_T)$ under policy π_θ and $R(\tau)$ is the total reward from trajectory τ.

Lemma 1. $\nabla_\theta V(\theta) = \sum_\tau R(\tau) P_\theta(\tau) \nabla_\theta \log(P_\theta(\tau))$

Proof.

$$\begin{aligned}
\nabla_\theta V(\theta) &= \nabla_\theta \sum_\tau P_\theta(\tau) R(\tau) \\
&= \sum_\tau R(\tau) \nabla_\theta P_\theta(\tau) \\
&= \sum_\tau R(\tau) \frac{P_\theta(\tau)}{P_\theta(\tau)} \nabla_\theta P_\theta(\tau) \\
&= \sum_\tau R(\tau) \frac{P_\theta(\tau)}{P_\theta(\tau)} \nabla_\theta P_\theta(\tau) \\
&= \sum_\tau R(\tau) P_\theta(\tau) \nabla_\theta \log(P_\theta(\tau))
\end{aligned}$$

□

Now, using Lemma 1, we can get an approximation for our update rule:

$$\nabla_\theta V(\theta) \approx \frac{1}{m} \sum_{i=1}^{m} R(\tau^{(i)}) \nabla_\theta \log(P_\theta(\tau^{(i)})).$$

However, we still don't know how to compute the gradient of the log-probability. For that, we need the following lemme:

Lemma 2. $\nabla_\theta \log(P_\theta(\tau^{(i)})) = \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t)$

Proof.

$$\begin{aligned}
\nabla_\theta \log(P_\theta(\tau^{(i)}) &= \nabla_\theta \log(\mu(s_0) \prod_{t=0}^{T-1} \pi_\theta(a_t|s_t) P(s_{t+1}|s_t, a_t)) \\
&= \nabla_\theta \log(\mu(s_0)) + \sum_{t=0}^{T-1} \log \pi_\theta(a_t|s_t) + \log P(s_{t+1}|s_t, a_t) \\
&= \sum_{t=0}^{T-1} \nabla_\theta \log(\pi_\theta(a_t|s_t))
\end{aligned}$$

□

With Lemma 2, we get the following update rule:

$$\nabla_\theta V(\theta) = \sum_\tau R(\tau) P_\theta(\tau) \left(\sum_{t=0}^{T-1} \nabla_\theta \log(\pi_\theta(a_t|s_t)) \right)$$

which can be approximated as

$$\nabla_\theta V(\theta) = \sum_{i=0}^{m} R(\tau^{(i)}) \left(\sum_{t=0}^{T-1} \nabla_\theta \log(\pi_\theta(a_t^{(i)}|s_t^{(i)})) \right)$$

We generalize this approach through the policy gradient theorem [1]. For an episodic task, let $\eta(s)$ be the number of time-steps spent in state s within a single episode. If $\mu_0(s)$ is the initial state distribution, then $\eta(s) = \mu_0(s) + \sum_{s'} \eta(s') \sum_a \pi_\theta(a|s') p(s|s', a)$. To turn this into a probability distribution, which we call the on-policy distribution i.e the fraction of time-steps spent in a state s, we get $\mu(s) = \frac{\eta(s)}{\sum_{s'} \eta(s')}$

Theorem 3. *Suppose, our task is episodic. For simplicity, we let $\gamma = 1$. Furthermore, suppose, our start state is s_0 for all trajectories. Then, $\nabla_\theta V(\theta) \propto \sum_s \mu(s) \sum_a q_{\pi_\theta}(s, a) \nabla_\theta \pi_\theta(a|s)$*

Proof.

$$\begin{aligned}
\nabla_\theta V(\theta) &= \nabla_\theta \left(\sum_a \pi_\theta(a|s) q_{\pi_\theta}(s, a) \right) \\
&= \sum_a \left(\nabla_\theta \pi_\theta(a|s) \right) q_{\pi_\theta}(s, a) + \pi_\theta(a|s) \nabla_\theta q_{\pi_\theta}(a, s) \\
&= \sum_a \left(\nabla_\theta \pi_\theta(a|s) \right) q_{\pi_\theta}(s, a) + \pi_\theta(a|s) \nabla_\theta \left(\sum_{s', r} p(s', r|s, a) \left(r + V_{\pi_\theta}(s') \right) \right) \\
&= \sum_a \left(\nabla_\theta \pi_\theta(a|s) \right) q_{\pi_\theta}(s, a) + \pi_\theta(a|s) \left(\sum_{s', r} p(s', r|s, a) \nabla_\theta \left(V_{\pi_\theta}(s') \right) \right) \\
&= \sum_a \nabla_\theta \pi_\theta(a|s) q_{\pi_\theta}(s, a) + \\
&\pi_\theta(a|s) \sum_{s'} p(s'|s, a) \left(\sum_{a'} \left(\nabla_\theta \pi_\theta(a'|s') q_{\pi_\theta}(s', a') \right) + \pi_\theta(a'|s') \sum_{s''} P(s''|s', a') \nabla_\theta V_{\pi_\theta}(s'') \right) \\
&= \sum_{x \in S} \sum_{k=0}^{\infty} \mathbb{P}(s_0 \to x, k, \pi_\theta) \sum_a \nabla_\theta \pi_\theta(a|x) q_{\pi_\theta}(x, a)
\end{aligned}$$

where $\mathbb{P}(s \to x, k, \pi_\theta)$ is the probability of reaching state x from x in k steps under policy π_θ.

$$\begin{aligned}
\nabla_\theta V(\theta) &= \sum_s \sum_{k=0}^{\infty} \mathbb{P}(s_0 \to s, k, \pi_\theta) \sum_a \nabla_\theta \pi_\theta(a|s) q_{\pi_\theta}(s,a) \\
&= \sum_s \eta(s) \sum_a \nabla_\theta \pi_\theta(a|s) q_{\pi_\theta}(s,a) \\
&= \sum_{s'} \eta(s') \sum_s \frac{\eta(s)}{\sum_{s'} \eta(s')} \sum_a \nabla_\theta \pi_\theta(a|s) q_{\pi_\theta}(s,a) \\
&= \sum_{s'} \eta(s') \sum_s \mu(s) \sum_a \nabla_\theta \pi_\theta(a|s) q_{\pi_\theta}(s,a) \\
&\propto \sum_s \mu(s) \sum_a \nabla_\theta \pi_\theta(a|s) q_{\pi_\theta}(s,a)
\end{aligned}$$

□

Thefore, our algorithm so far is:

$$\nabla_\theta V(\theta) = \sum_\tau R(\tau) P_\theta(\tau) \left(\sum_{t=0}^{T-1} \nabla_\theta \log(\pi_\theta(a^t|s^t)) \right)$$

where a^t, s^t are in the trajectory τ. We can now approximate this as:

$$\nabla_\theta V(\theta) \approx \frac{1}{m} \sum_{i=1}^{m} R(\tau^{(i)}) \left(\sum_{t=0}^{T-1} \nabla_\theta \log(\pi_\theta(a_t^{(i)}|s_t^{(i)})) \right)$$

This algorithm is, however, very noisy since our trajectory samples are often quite noisy. To do this, we use several modifications.

2.1 REINFORCE (monte carlo policy gradient)

So far, our approach has been to calculate $\nabla_\theta V(\theta) = \nabla_\theta \mathbb{E}_\tau[R(\tau)]$ which we calculated as: $\nabla_\theta V(\theta) = \nabla_\theta \mathbb{E}_\tau[R(\tau)] = \nabla_\theta \sum_\tau P_\theta(\tau) R(\tau) = \mathbb{E}_\tau \left[\sum_{t=0}^{T-1} r_t \left(\sum_{t=0}^{T-1} \nabla_\theta \log(\pi_\theta(a^t|s^t)) \right) \right]$.

Instead, what if we maximized the reward at a single timestep. Through a similar process of derivation, we can write $\nabla_\theta \mathbb{E}[r_{t'}] = \mathbb{E}[r_{t'} \sum_{t=0}^{t'} \nabla_\theta \log(\pi_\theta(a_t|s_t))]$. We can then sum over all t'

to get:

$$\nabla_\theta V(\theta) = \nabla_\theta \mathbb{E}[R] = \mathbb{E}\left[\sum_{t'=0}^{T-1} r_{t'} \sum_{t=0}^{t'} \nabla_\theta \log(\pi_\theta(a_t|s_t))\right] = \mathbb{E}\left[\sum_{t=0}^{T-1} \nabla_\theta \log(\pi_\theta(a_t|s_t)) \sum_{t'=t}^{T-1} r_{t'}\right]$$

.

Now, we use $G_t^{(i)} = \sum_{t'=t}^{T-1} r_{t'}$ to finally write:

$$\nabla_\theta V(\theta) \approx \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{T-1} \nabla_\theta \log(\pi_\theta(a_t|s_t)) G_t^{(i)}$$

We can derive this from the policy gradient theorem as follow:

$$\begin{aligned}
\nabla_\theta V(\theta) &\propto \sum_s \mu(s) \sum_a q_{\pi_\theta(s,a)} \nabla_\theta \pi_\theta(a|s) \\
&= \mathbb{E}_{s_t \sim \pi_\theta}\left[\sum_a q_{\pi_\theta}(s_t, a) \nabla_\theta \pi_\theta(a|s_t)\right] \\
&= \mathbb{E}_{s_t \sim \pi_\theta}\left[\sum_a q_{\pi_\theta}(s_t, a) \pi_\theta(a|s_t) \frac{\nabla_\theta \pi_\theta(a|s_t)}{\pi_\theta(a|s_t)}\right] \\
&= \mathbb{E}_{s_t \sim \pi_\theta}\left[\mathbb{E}_{a_t \sim \pi_\theta}\left[q_{\pi_\theta}(s_t, a_t) \frac{\nabla_\theta \pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)} | s_t\right]\right] \\
&= \mathbb{E}_{\pi_\theta}\left[q_{\pi_\theta}(s_t, a_t) \frac{\nabla_\theta \pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)}\right] \\
&= \mathbb{E}_{\pi_\theta}\left[\mathbb{E}_{\pi_\theta}\left[G_t \frac{\nabla_\theta \pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)} | s_t, a_t\right]\right] \\
&= \mathbb{E}_{\pi_\theta}\left[G_t \frac{\nabla_\theta \pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)}\right]
\end{aligned}$$

where in the second last line we used the fact that $\mathbb{E}_{\pi_\theta}[G_t|s_t, a_t] = q_{\pi_\theta}(s_t, a_t)$.

2.2 REINFORCE with baseline

We introduce a baseline to further reduce variance:

$$\nabla_\theta V(\theta) \approx \mathbb{E}_\tau\left[\sum_{t=0}^{T-1} \nabla_\theta \log(\pi_\theta(a_t|s_t)) \sum_{t'=t}^{T-1} r_{t'} - b(s_t)\right]$$

where we usually select $b(s_t) = \mathbb{E}[r_t + r_{t+1} + ... + r_{T-1}]$. We could also use a learned state-action value function parametrized by ϕ, $v_\phi(s_t) =: b(s_t)$.

2.3 Actor-Critic Methods

Actor-critic methods use a learned value function $v\phi(s_t)$ but not just for baseline. Instead, we use it for an approximation of the rewards too:

$$\begin{aligned}\nabla_\theta V(\theta) &\approx \mathbb{E}_\tau\left[\sum_{t=0}^{T-1} \nabla_\theta \log(\pi_\theta(a_t|s_t))(r(s_t,a_t)+v_\phi(s_{t+1})-b(s_t))\right] \\ &\approx \mathbb{E}_\tau\left[\sum_{t=0}^{T-1} \nabla_\theta \log(\pi_\theta(a_t|s_t))(A^{\pi_\theta}(s_t,a_t))\right]\end{aligned}$$

where $A^{\pi_\theta}(s_t,a_t) := q^{\pi_\theta}(s_t,a_t) - V^{\pi_\theta}(s_t)$ is our advantage function.

2.4 Pseudocode for "Vanilla" Policy Gradient Algorithm

```
Initialize policy parametre $\theta$ and baseline $b$
for iteration=1,2,... do:
    Collect a set of trajectories under current policy
    For each timestep $t$ in each trajectory $T^{(i)}$, compute:
        $G_t^i = \sum_{t'=t}^{T-1} r_{t'}^i$
        $A_t^i = G_t^i - b(s_t)$
    Refit the baseline by minimizing $\sum_{t,i} ||b(s_t) - G_t^i||^2$
    Update policy using policy gradient estimate:
        Gradient is equal to sum of the terms $\nabla_\theta \log(\pi_\theta(a_t|s_t))A_t$
```

3 Performance Difference Lemma

We want to prove a few results that are building blocks for trust regional policy optimisation.

Firstly, note that the probability of sampling any particular trajectory $\tau = (s_0, a_0, r_0, s_1, a_1, ...)$ is $P_\theta(\tau) = \prod_{i=0}^{\infty} \pi_\theta(a_i|s_i)P(s_{i+1}|s_i,a_i)$. The probability of sampling a particular trajectory τ

such that $s_t = s$ is $P_\theta(s_t = s) = \sum_{a_0} \sum_{s_1} ... \sum_{s_{t-1}} \sum_{a_{t-1}} \Big(\prod_{i=1}^{T-1} \pi_\theta(a_i|s_i) P(s_{i+1}|s_i, a_i) \Big)$.

Let the distribution over all states induced by π_θ be:

$$d^{\pi_\theta}(s) := (1-\gamma) \sum_{t=0}^{\infty} \gamma^t P_\theta(s_t = s).$$

Lemma 4. $\mathbb{E}_{\tau\sim P_\theta}\left[\sum_{t=0}^{\infty}\gamma^t f(s_t,a_t)\right] = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi_\theta}}\left[\mathbb{E}_{a\sim\pi_\theta(\cdot|s)}\left[f(s,a)\right]\right]$

Proof.

$$\begin{aligned}
\mathbb{E}_{\tau\sim P_\theta}\left[\sum_{t=0}^{\infty}\gamma^t f(s_t,a_t)\right] &= \sum_{\tau} P_\theta(\tau)\sum_{t=0}^{\infty}\gamma^t f(s_t,a_t)\\
&= \sum_{\tau}\prod_{i=0}^{\infty}\pi_\theta(a_i|s_i)P(s_{i+1}|s_i,a_i)\sum_{t=0}^{\infty}\gamma^t f(s_t,a_t)\\
&= \sum_{a_0}\sum_{s_1}\sum_{a_1}\cdots\prod_{i=0}^{\infty}\pi(a_i|s_i)P(s_{i+1}|s_i,a_i)\sum_{t=0}^{\infty}\gamma^t f(s_t,a_t)\\
&= \sum_{a_0}\pi_\theta(a_0|s_0)f(s_0,a_0) + \gamma\sum_{a_0}\sum_{s_1}\sum_{a_1}\pi_\theta(a_0|s_0)P(s_1|s_0,a_0)\pi_\theta(a_1|s_1)f(s_1,a_1) + \cdots\\
&= \sum_{t=0}^{\infty}\gamma^t\sum_{a_0}\sum_{s_1}\sum_{a_1}\cdots\sum_{s_t}\sum_{a_t}\sum_{s_{t+1}}\prod_{i=0}^{t}\pi_\theta(a_i|s_i)P(s_{i+1}|s_i,a_i)f(s_t,a_t)\\
&= \sum_{t=0}^{\infty}\gamma^t\sum_{s_{t+1}}(\cdots)\,f(s_t,a_t)\\
&= \sum_{t=0}^{\infty}\gamma^t(\cdots)\,f(s_t,a_t)\\
&= \sum_{t=0}^{\infty}\gamma^t\sum_{s_t}\sum_{a_t}\left(\sum_{a_0}\cdots\sum_{s_{t-1}}\sum_{a_{t-1}}\prod_{i=0}^{t-1}\pi_\theta(a_i|s_i)P(s_{i+1}|s_i,a_i)\right)\pi_\theta(a_t|s_t)f(s_t,a_t)\\
&= \sum_{t=0}^{\infty}\gamma^t\sum_{s_t}\sum_{a_t}P_\theta(s_t)\pi_\theta(a_t|s_t)f(s_t,a_t)\\
&= \frac{1}{1-\gamma}(1-\gamma)\sum_{t=0}^{\infty}\gamma^t\sum_{s_t}P_\theta(s_t)\sum_{a_t}\pi_\theta(a_t|s_t)f(s_t,a_t)\\
&= \frac{1}{1-\gamma}(1-\gamma)\sum_{t=0}^{\infty}\gamma^t\sum_{s_t}P_\theta(s_t)\mathbb{E}_{a\sim\pi_\theta}\left[f(s_t,a)\right]\\
&= \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi_\theta}}\left[\mathbb{E}_{a\sim\pi_\theta}\left[f(s_t,a)\right]\right]
\end{aligned}$$

□

Theorem 5. *(Performance Difference Lemma)*

$$V_\pi(s_0) - V_{\pi'}(s_0) = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^\pi}\left[\mathbb{E}_{a\sim\pi(s)}\left[A^{\pi'}(s,a)\right]\right]$$

where the advantage function is $A^\pi(s,a) := q^\pi(s,a) - V^\pi(s)$.

Proof.

$$
\begin{aligned}
&V_\pi(s_0) - V_{\pi'}(s_0)\\
&= \mathbb{E}_{\tau\sim P^\pi}\left[\sum_{t=0}^{\infty}\gamma^t R(s_t,a_t)\right] + \mathbb{E}_{\tau\sim P^\pi}\left[\sum_{t=0}^{\infty}\gamma^{t+1}V_{\pi'}(s_{t+1})\right] - \mathbb{E}_{\tau\sim P^\pi}\left[\sum_{t=0}^{\infty}\gamma^{t+1}V_{\pi'}(s_{t+1})\right] - V_{\pi'}(s_0)\\
&= \mathbb{E}_{\tau\sim P^\pi}\left[\sum_{t=0}^{\infty}\gamma^t (R(s_t,a_t) + \gamma V_{\pi'}(s_{t+1})) - \sum_{t=0}^{\infty}\gamma^{t+1}V_{\pi'}(s_{t+1}) - V_{\pi'}(s_0)\right]
\end{aligned}
$$

Now, we expand the first term:

$$
\begin{aligned}
&\mathbb{E}_{\tau\sim P^\pi}\left[\sum_{t=0}^{\infty}\gamma^t\left(R(s_t,a_t) + \gamma V_{\pi'}(s_{t+1})\right)\right]\\
&= \sum_{t=0}^{\infty}\gamma^t\mathbb{E}_{\tau\sim P^\pi}\left[R(s_t,a_t) + \gamma V^{\pi'}(s_{t+1}|s_t,a_t)\right]\\
&= \sum_{t=0}^{\infty}\gamma^t\mathbb{E}_{s_t\sim P^\pi}\left[\mathbb{E}_{a_t\sim P^\pi}\left[\mathbb{E}_{s_{t+1}\sim P^\pi}\left[R(s,a) + \gamma V^{\pi'}(s_{t+1})|s=s_t, a=a_t\right]|a=a_t\right]|s=s_t\right]\\
&= \sum_{t=0}^{\infty}\gamma^t\mathbb{E}_{s_t,a_t\sim P^\pi}\left[R(s,a) + \gamma\sum_{s_{t+1}}P(s_{t+1}|s_t,a_t)V^{\pi'}(s_{t+1})|s=s_t, a=a_t\right]\\
&= \sum_{t=0}^{\infty}\gamma^t\mathbb{E}_{\tau\sim P^\pi}\left[Q^{\pi'}(s,a)|s=s_t, a=a_t\right]\\
&= \mathbb{E}_{\tau\sim P^\pi}\left[\sum_{t=0}^{\infty}\gamma^t Q^{\pi'}(s_t,a_t)\right]
\end{aligned}
$$

Therefore, we get

$$
\begin{aligned}
V^\pi(s_0) - V^{\pi'}(s_0) &= \mathbb{E}_{\tau\sim P^\pi}\left[\sum_{t=0}^{\infty}\gamma^t\left(Q^{\pi'}(s_t,a_t) - V^{\pi'}(s_t)\right)\right]\\
&= \mathbb{E}_{\tau\sim P^\pi}\left[\sum_{t=0}^{\infty}\gamma^t A^{\pi'}(s_t,a_t)\right]\\
&= \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^\pi}\left[\mathbb{E}_{a\sim\pi}\left[A^{\pi'}(s,a)\right]\right]
\end{aligned}
$$

□

4 Brief Notes on TRPO and PPO

There are two major issues with vanilla policy gradient methods. Firstly, it is difficult to optimize in the sense that it is difficult to find the right step size to use in gradient descent. The input data distribution is non-stationary - you sample trajectories using a learned policy, then you use those samples to update your policy, then you use this updated policy to sample new trajectories. However, if, at any point in time, you use a set of bad samples and therefore, your optimisation step is wrong, this could lead to performance collapse - with the bad samples, you take a "wrong step" to get a poor policy, with which you sample new trajectories which are also poor which you then use to optimise again. The second issue is that the algorithm is sample inefficient - with any particular set of sampled trajectories, we carry out one step of gradient descent and then throw those samples out. For future optimisation steps, we sample *new* trajectories. Although we have made some modifications to the basic vanilla PG algorithm like we found the actor-critic methods, they are still insufficient in completely curbing these issues.

We now derive the building blocks of Trust Region Policy Optimisation (TRPO): We already saw the performance difference lemme:

$$V_{\pi'} - V_\pi = \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi'}}\left[\mathbb{E}_{a\sim\pi'}\left[A^\pi(s,a)\right]\right]$$

Now, suppose, our current policy is π. In our next step, we essentially want to maximize the difference between $V_{\pi'} - V_\pi$. Therefore,

$$\begin{aligned}\arg\max_{\pi'} V_{\pi'} - V_\pi &= \arg\max_{\pi'} \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi'}}\left[\mathbb{E}_{a\sim\pi'}\left[A^\pi(s,a)\right]\right] \\ &= \arg\max_{\pi'} \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi'}}\left[\mathbb{E}_{a\sim\pi}\left[\frac{\pi'(a|s)}{\pi(a|s)}A^\pi(s,a)\right]\right] \\ &\approx \arg\max_{\pi'} \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\pi}}\left[\mathbb{E}_{a\sim\pi}\left[\frac{\pi'(a|s)}{\pi(a|s)}A^\pi(s,a)\right]\right] \\ &=: \arg\max_{\pi'} \mathcal{L}_\pi(\pi')\end{aligned}$$

where, in the second last line, we made the approximation $d^{\pi'} \approx d^\pi$. This approximation only holds true if

$$||V_{\pi'} - V_\pi - \mathcal{L}_\pi(\pi')|| \leq C\sqrt{\mathbb{E}_{s_t\sim\pi}\left[D_{KL}(\pi(\cdot|s_t)||\pi'(\cdot|s_t))\right]}$$

With this, TRPO maximises $\mathcal{L}_\pi(\pi')$ subject to $\mathbb{E}_{s\sim\pi}\left[D_{KL}(\pi(\cdot|s)||\pi'(\cdot|s))\right] \leq \delta$. Note that, in actual implementation, we use a learned approximation for the advantage function.

Also note that we get monotonic improvement since the KL divergence is zero when $\pi' = \pi$ whereas $\mathcal{L}_\pi(\pi) = 0$ too, therefore, the performance of π' is at least as good as π.

PPO slightly modifies this - instead of placing a harsh constraint in the optimization process (which requires conjugate gradient descent otherwise), instead maximize $\mathbb{E}_{s_t\sim d^\pi, a_t\sim\pi}\left[\frac{\pi'(a_t|s_t)}{\pi(a_t|s_t)}A^\pi(s_t, a_t) - \beta \cdot D_{KL}(\pi'(\cdot|s_t)||\pi(\cdot|s_t))\right]$. If the KL-divergence is too high, we adaptively increase β and if it is small, then we decrease β.

5 References

[1] RS Sutton, AG Barto. *Reinforcement learning: An introduction.* MIT Press. 2018.

[2] Emma Brunskill's lectures on Policy Gradient Methods, *CS 234: Reinforcement Learning*, Stanford 2023.