Section 5: Advanced Policy Gradients

Author: Pranav Atreya

Material adapted from lectures by Sergey Levine

1 KL Divergence

The Kullback-Leibler divergence is a very widely used probability distribution divergence measure in machine learning and RL. Given two distributions $p(x)$ and $q(x)$ over a space $\mathcal{X}$,

$$D_{\mathrm{KL}}(p\|q) = \mathbb{E}_{x\sim p}\left[\log p(x) - \log q(x)\right]. \tag{1}$$

I.e., KL divergence is measured as a difference in log probabilities taken under expectation over the first distribution. KL divergence is always ≥ 0.

You might observe that the KL divergence is asymmetric. The forward KL, $\mathbb{E}_{x\sim p}\left[\log p(x) - \log q(x)\right]$ is not necessarily equal to the reverse KL, $\mathbb{E}_{x\sim q}\left[\log q(x) - \log p(x)\right]$ (can you calculate under what conditions they are equal?). Depending on the problem, you might prefer to use one over the other. A good way to decide which direction to use is to ask, do you want the distribution $q(x)$ to cover the mass of the distribution $p(x)$, or do you want $q(x)$ to mode-seek $p(x)$? Forward KL encourages the former because $\log q$ goes to $-\infty$ when q assigns low probability mass to places where p does, thus encouraging q to "cover" p. Reverse KL penalizes places where q puts mass where p is tiny (since $\log p$ goes to $-\infty$), which generally leads to mode-seeking, as q is encouraged not to place probability mass where p says it's unlikely. Think about what happens when p is a (fixed) mixture of two well-separated Gaussians, and q is a single Gaussian. What will q look like if you minimize forward vs reverse KL?

KL divergence is a very popular divergence measure because **(1)** it is tractable to compute for many popular distribution parameterizations (calculating the expectation for categorical distributions is very easy, and for many continuous valued distributions, e.g., Gaussians, KL can be calculated in closed form), **(2)** KL divergence is differentiable so long as $\mathrm{supp}(p) \subseteq \mathrm{supp}(q)$ (the support of a distribution p is $\mathrm{supp}(p) = \{x : p(x) > 0\}$), and **(3)** the KL divergence between two distributions close in parameter space can be approximated well with a 2nd order Taylor expansion: $D_{\mathrm{KL}}(p_\theta\|p_{\theta+\delta}) \approx \frac{1}{2}\delta^T F(\theta)\delta$ where F is the Fisher information matrix (see next).

Approximating KL for parametrically close distributions It can be the case that the two distributions we want to measure KL divergence between are both parameterized in the same way (i.e., by a neural network with the same architecture), and the parameters of these distributions happen to be close in parameter space. Denote these distributions p_θ and $p_{\theta+\delta}$. As you will see later in the Natural Policy Gradient, we can set up a scenario where we want to take gradient steps updating $p_{\theta+\delta}$ using samples collected by p_θ, and in order to make this operation safe, we need to constrain $p_{\theta+\delta}$ to be close to p_θ, for which we will use the KL constraint $D_{\mathrm{KL}}(p_\theta\|p_{\theta+\delta})$. While we can solve the problem of optimizing $p_{\theta+\delta}$ while constraining $D_{\mathrm{KL}}(p_\theta\|p_{\theta+\delta})$ by fully evaluating $D_{\mathrm{KL}}(p_\theta\|p_{\theta+\delta})$, we will find we can make the constrained optimization problem

computationally easier by approximating $D_{\mathrm{KL}}\left(p_\theta \| p_{\theta+\delta}\right)$ with a Taylor expansion, which will be an accurate approximation if δ is small. Let's work out the Taylor expansion:

$$
\begin{aligned}
D_{\mathrm{KL}}\left(p_\theta \| p_{\theta+\delta}\right) &= \mathbb{E}_{x\sim p_\theta}\left[\log p_\theta(x) - \log p_{\theta+\delta}(x)\right] && (2)\\
&\approx \mathbb{E}_{x\sim p_\theta}\left[\log p_\theta(x) - \left(\log p_\theta(x) + \delta^T \nabla_\theta \log p_\theta(x) + \frac{1}{2}\delta^T \nabla_\theta^2 \log p_\theta(x)\delta\right)\right] && (3)\\
&= \mathbb{E}_{x\sim p_\theta}\left[-\delta^T \nabla_\theta \log p_\theta(x) - \frac{1}{2}\delta^T \nabla_\theta^2 \log p_\theta(x)\delta\right] && (4)\\
&= -\delta^T \mathbb{E}_{x\sim p_\theta}\left[\nabla_\theta \log p_\theta(x)\right] - \frac{1}{2}\mathbb{E}_{x\sim p_\theta}\left[\delta^T \nabla_\theta^2 \log p_\theta(x)\delta\right] && (5)\\
&= -\delta^T \int p_\theta(x) \nabla_\theta \log p_\theta(x)\, \mathrm{d}x - \frac{1}{2}\mathbb{E}_{x\sim p_\theta}\left[\delta^T \nabla_\theta^2 \log p_\theta(x)\delta\right] && (6)\\
&= -\delta^T \int p_\theta(x) \frac{\nabla_\theta p_\theta(x)}{p_\theta(x)}\, \mathrm{d}x - \frac{1}{2}\mathbb{E}_{x\sim p_\theta}\left[\delta^T \nabla_\theta^2 \log p_\theta(x)\delta\right] && (7)\\
&= -\delta^T \int \nabla_\theta p_\theta(x)\, \mathrm{d}x - \frac{1}{2}\mathbb{E}_{x\sim p_\theta}\left[\delta^T \nabla_\theta^2 \log p_\theta(x)\delta\right] && (8)\\
&= -\delta^T \nabla_\theta \int p_\theta(x)\, \mathrm{d}x - \frac{1}{2}\mathbb{E}_{x\sim p_\theta}\left[\delta^T \nabla_\theta^2 \log p_\theta(x)\delta\right] && (9)\\
&= -\delta^T \nabla_\theta(1) - \frac{1}{2}\mathbb{E}_{x\sim p_\theta}\left[\delta^T \nabla_\theta^2 \log p_\theta(x)\delta\right] && (10)\\
&= -\frac{1}{2}\mathbb{E}_{x\sim p_\theta}\left[\delta^T \nabla_\theta^2 \log p_\theta(x)\delta\right] && (11)\\
&= -\frac{1}{2}\mathbb{E}_{x\sim p_\theta}\left[\delta^T \nabla_\theta \left(\nabla_\theta \log p_\theta(x)\right)\delta\right] && (12)\\
&= -\frac{1}{2}\mathbb{E}_{x\sim p_\theta}\left[\delta^T \nabla_\theta \frac{\nabla_\theta p_\theta(x)}{p_\theta(x)}\delta\right] && (13)\\
&= -\frac{1}{2}\mathbb{E}_{x\sim p_\theta}\left[\delta^T \left(\frac{\nabla_\theta^2 p_\theta(x)}{p_\theta(x)} - \frac{\nabla_\theta p_\theta(x) \nabla_\theta p_\theta(x)^T}{p_\theta(x)^2}\right)\delta\right] && (14)\\
&= -\frac{1}{2}\delta^T \left(\mathbb{E}_{x\sim p_\theta}\left[\frac{\nabla_\theta^2 p_\theta(x)}{p_\theta(x)}\right] - \mathbb{E}_{x\sim p_\theta}\left[\frac{\nabla_\theta p_\theta(x) \nabla_\theta p_\theta(x)^T}{p_\theta(x)^2}\right]\right)\delta && (15)\\
&= -\frac{1}{2}\delta^T \left(\left(\int \nabla_\theta^2 p_\theta(x)\, \mathrm{d}x\right) - \left(\int \frac{\nabla_\theta p_\theta(x) \nabla_\theta p_\theta(x)^T}{p_\theta(x)}\, \mathrm{d}x\right)\right)\delta && (16)\\
&= -\frac{1}{2}\delta^T \left(\left(\nabla_\theta^2 \int p_\theta(x)\, \mathrm{d}x\right) - \left(\int \left(\nabla_\theta p_\theta(x)\right)\left(\frac{\nabla_\theta p_\theta(x)}{p_\theta(x)}\right)^T \mathrm{d}x\right)\right)\delta && (17)\\
&= -\frac{1}{2}\delta^T \left(\nabla_\theta^2(1) - \left(\int \left(p_\theta(x)\nabla_\theta \log p_\theta(x)\right)\left(\nabla_\theta \log p_\theta(x)\right)^T \mathrm{d}x\right)\right)\delta && (18)\\
&= -\frac{1}{2}\delta^T \left(-\mathbb{E}_{x\sim p_\theta} \nabla_\theta \log p_\theta(x) \nabla_\theta \log p_\theta(x)^T\right)\delta && (19)\\
&= \frac{1}{2}\delta^T F(\theta)\delta && (20)
\end{aligned}
$$

where $F(\theta) = \mathbb{E}_{x\sim p_\theta} \nabla_\theta \log p_\theta(x) \nabla_\theta \log p_\theta(x)^T$ is called the Fisher information matrix. So as desired we have written out an approximation of the KL divergence with a 2nd order Taylor expansion,

and we will use this approximation later to do KL-constrained policy optimizations.

2 A First Attempt at Using Off-Policy Data: the Off-Policy Policy Gradient

The remainder of this section note will be structured as follows. We will start from the vanilla policy gradient and the REINFORCE algorithm from lectures 5/6, we will motivate why we would like to make REINFORCE more sample efficient, and then we will try to find ways to increase sample efficiency by using slightly off-policy data.

Recap: the REINFORCE Algorithm Recall our estimation for the policy gradient:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta} \sum_{t=0}^{H-1} \nabla_\theta \log \pi_\theta(a_t|s_t) \hat{A}^\pi_t \tag{21}$$

$$\approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{H-1} \nabla_\theta \log \pi_\theta(a^i_t|s^i_t) \hat{A}^\pi_{i,t} \tag{22}$$

Using this gradient expression to optimize π_θ yielded a very simple RL algorithm called REINFORCE:

Algorithm 1 REINFORCE

1: Sample trajectories $\{\tau^i\}_{i=1}^N$ by running $\pi_\theta(a_t \mid s_t)$.
2: $\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{H-1} \nabla_\theta \log \pi_\theta(a^i_t \mid s^i_t) \hat{A}^\pi_{i,t}$.
3: $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$.

A goal in reinforcement learning is to train policies that optimize the RL objective well, but to also do so in a manner that is sample efficient, because for many problems we care about collecting samples can be expensive. It is easy to see how REINFORCE, at an intuitive level, is wasteful in the samples it collects; our estimate of the gradient in step 2 can only use samples collected from π_θ, because the expectation in eq. 21 is over π_θ. This means that when we take a single gradient step in step 3, for the next iteration we are forced to collect a brand new set of samples (and we often want to take many thousands of gradient steps). This issue is further exacerbated by the fact that our policy gradient estimate, eq. 22, is itself high variance. We studied how intelligently computing $\hat{A}^\pi_{i,t}$, such as by using discounting, baselines, causality, GAE, etc., can improve sample efficiency, but in practice even with these improvements many samples can still be needed to get sufficiently low-variance gradient estimates useful for learning.

It would be great from a sample efficiency perspective if our policy gradient estimate could use samples collected by previous policies. This would unlock access to all data collected by all previous policies, and more data intuitively should yield better gradient estimates. It turns out we can exactly compute a form of the policy gradient that leverages off-policy samples.

2.1 The Off-Policy Policy Gradient

The goal in computing the off-policy policy gradient will be to compute $\nabla_\theta J(\theta) = \nabla_\theta \mathbb{E}_{\tau\sim p_\theta(\tau)}\left[r(\tau)\right]$ using trajectories not sampled from $p_\theta(\tau)$, but from some other distribution $\bar{p}(\tau)$, despite the expectation in the RL objective being under $p_\theta(\tau)$. $\bar{p}(\tau)$ can for example represent the combined distribution of all previous policies, or even previous policies and some human demos. We will denote the "behavior" policy that collected $\bar{p}(\tau)$ as $\pi_\beta(a|s)$.

$$\begin{aligned}
\nabla_\theta J(\theta) &= \nabla_\theta \mathbb{E}_{\tau\sim p_\theta(\tau)}\left[r(\tau)\right] && (23)\\
&= \nabla_\theta \int p_\theta(\tau) r(\tau)\ \mathrm{d}\tau && (24)\\
&= \nabla_\theta \int \frac{\bar{p}(\tau)}{\bar{p}(\tau)} p_\theta(\tau) r(\tau)\ \mathrm{d}\tau && (25)\\
&= \nabla_\theta \int \bar{p}(\tau) \frac{p_\theta(\tau)}{\bar{p}(\tau)} r(\tau)\ \mathrm{d}\tau && (26)\\
&= \nabla_\theta \mathbb{E}_{\tau\sim\bar{p}(\tau)}\left[\frac{p_\theta(\tau)}{\bar{p}(\tau)} r(\tau)\right] && (27)
\end{aligned}$$

where we have just applied a trick called importance sampling (IS), with $\frac{p_\theta(\tau)}{\bar{p}(\tau)}$ being the IS weights. Expanding from eq. 26,

$$\begin{aligned}
\nabla_\theta \int \bar{p}(\tau) \frac{p_\theta(\tau)}{\bar{p}(\tau)} r(\tau)\ \mathrm{d}\tau &= \int \nabla_\theta \left(\bar{p}(\tau) \frac{p_\theta(\tau)}{\bar{p}(\tau)} r(\tau)\right)\ \mathrm{d}\tau && (28)\\
&= \int \bar{p}(\tau) \frac{\nabla_\theta p_\theta(\tau)}{\bar{p}(\tau)} r(\tau)\ \mathrm{d}\tau && (29)\\
&= \mathbb{E}_{\tau\sim\bar{p}(\tau)} \frac{\nabla_\theta p_\theta(\tau)}{\bar{p}(\tau)} r(\tau) && (30)
\end{aligned}$$

because only $p_\theta(\tau)$ depends on θ.

$$\mathbb{E}_{\tau\sim\bar{p}(\tau)} \frac{\nabla_\theta p_\theta(\tau)}{\bar{p}(\tau)} r(\tau) = \mathbb{E}_{\tau\sim\bar{p}(\tau)} \frac{p_\theta(\tau)\nabla_\theta \log p_\theta(\tau)}{\bar{p}(\tau)} r(\tau) \tag{31}$$

applying the same log-derivative trick we used to derive the (vanilla) policy gradient.

$$\begin{aligned}
\mathbb{E}_{\tau\sim\bar{p}(\tau)} \frac{p_\theta(\tau)}{\bar{p}(\tau)} \nabla_\theta \log p_\theta(\tau) r(\tau) &= \mathbb{E}_{\tau\sim\bar{p}(\tau)} \frac{p(s_0)\prod_{t=0}^{H-1} \pi_\theta(a_t|s_t) p(s_{t+1}|s_t,a_t)}{p(s_0)\prod_{t=0}^{H-1} \pi_\beta(a_t|s_t) p(s_{t+1}|s_t,a_t)} \nabla_\theta \log p_\theta(\tau) r(\tau) && (32)\\
&= \mathbb{E}_{\tau\sim\bar{p}(\tau)} \frac{\prod_{t=0}^{H-1} \pi_\theta(a_t|s_t)}{\prod_{t=0}^{H-1} \pi_\beta(a_t|s_t)} \nabla_\theta \log p_\theta(\tau) r(\tau) && (33)
\end{aligned}$$

which, then expanding out $\nabla_\theta \log p_\theta(\tau) r(\tau)$ like we did for the vanilla PG gives us

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\bar{p}(\tau)} \left(\frac{\prod_{t=0}^{H-1} \pi_\theta(a_t|s_t)}{\prod_{t=0}^{H-1} \pi_\beta(a_t|s_t)}\right) \left(\sum_{t=0}^{H-1} \nabla_\theta \log \pi_\theta(a_t|s_t)\right) \left(\sum_{t=0}^{H-1} r(s_t,a_t)\right) \tag{34}$$

Note that this last expression is exactly the same as the on-policy policy gradient, except with an IS term on the left that is a product of probability ratios. Let's multiply out the three terms inside the expectation, applying the principle of causality.

$$= \mathbb{E}_{\tau\sim\bar{p}(\tau)}\left[\sum_{t=0}^{H-1}\nabla_\theta \log \pi_\theta(a_t|s_t)\left(\prod_{t'=0}^{t}\frac{\pi_\theta(a_{t'}|s_{t'})}{\pi_\beta(a_{t'}|s_{t'})}\right)\left(\sum_{t'=t}^{H-1} r(s_{t'},a_{t'})\left(\prod_{t''=t+1}^{t'}\frac{\pi_\theta(a_{t''}|s_{t''})}{\pi_\beta(a_{t''}|s_{t''})}\right)\right)\right] \quad (35)$$

This last step was a little tricky, so try to work out how multiplying out the terms results in this expression and how causality is applied (hint: first apply causality to the return-to-go term, then distribute in the IS term, split the product, then distribute again into the RTG term, then apply causality to the 2nd IS term).

Okay! We have now derived an exact expression for the off-policy policy gradient. Are we done? Unfortunately no. While we can derive from this an unbiased estimator for the off-policy policy gradient (by sampling from $\bar{p}(\tau)$), the variance of this estimator has blown up, now for a new reason: the importance-sampling weights are products of per-step ratios over the horizon H, and products like this can vary dramatically in magnitude across trajectories. Since these weights multiply the $\nabla_\theta \log \pi_\theta$ terms in the gradient expression, $\nabla_\theta \log \pi_\theta$ terms with very small weights are effectively "ignored" relative to the ones with larger weights, in effect decreasing the effective dataset size, blowing up the sample variance.

3 Off-Policy Policy Iteration, Instead of PG

Attempting to optimize the RL objective by computing (and taking gradient steps in the direction of) the off-policy policy gradient led to a gradient estimate, that while could consume off-policy samples, was very high variance. So towards finding our desired more sample efficient algorithm, we are now going to pursue a new direction: finding an off-policy policy iteration algorithm. Recall that a policy iteration algorithm is one that takes steps that guarantee improvement of the RL objective. As we will see, we will be able to write out a practical policy iteration algorithm that will look very similar to policy gradient.

We can think of the goal of policy iteration as finding a new set of parameters θ' such that $J(\theta') - J(\theta) > 0$. Recall the time-discounted form of the RL objective:

$$J(\theta) = \mathbb{E}_{\tau\sim p_\theta(\tau)}\sum_t \gamma^t r(s_t,a_t) \quad (36)$$

Our plan will be as follows. We will choose an appropriate $\boxed{?}$ in the equation $J(\theta') - J(\theta) = \boxed{?}$, where then instead of optimizing $J(\theta') - J(\theta)$ w.r.t. θ' with gradient ascent (which is equivalent to just optimizing $J(\theta')$ w.r.t. θ'), we will optimize $\boxed{?}$. If we were to directly try to optimize $\boxed{?}$ with gradient ascent, then we will exactly recover the policy gradient (because $J(\theta') - J(\theta) = \boxed{?}$), which doesn't get us anywhere because as we just saw it is difficult to optimize the policy gradient with off-policy samples. Instead what we will do is show that if θ' is similar enough to θ, then we can find an expression that lower-bounds $\boxed{?}$ (where the tightness of this bound is a function of *how* similar θ' is to θ), where this lower bound only involves expectations over θ, allowing us to optimize

it with gradient ascent using samples only from θ. Optimizing lower bounds of objectives that are difficult to optimize directly is a very standard practice in machine learning. We will recover a gradient (the gradient of the lower bound of $\boxed{?}$) that will look a lot like the policy gradient, but won't exactly equal it (unless $\theta' = \theta$), but this is okay because following the gradient of (the lower bound of) $J(\theta') - J(\theta)$ still improves the RL objective. This makes this a policy iteration algorithm. Okay let's get started:

$$
\begin{aligned}
J(\theta') - J(\theta) &= \boxed{?} && (37)\\
&= J(\theta') - \mathbb{E}_{s_0 \sim p(s_0)}\left[V^{\pi_\theta}(s_0)\right] && (38)\\
&= J(\theta') - \mathbb{E}_{\tau \sim p_{\theta'}(\tau)}\left[V^{\pi_\theta}(s_0)\right] && (39)\\
&= J(\theta') - \mathbb{E}_{\tau \sim p_{\theta'}(\tau)}\left[\sum_{t=0}^{\infty} \gamma^t V^{\pi_\theta}(s_t) - \sum_{t=1}^{\infty} \gamma^t V^{\pi_\theta}(s_t)\right] && (40)\\
&= J(\theta') + \mathbb{E}_{\tau \sim p_{\theta'}(\tau)}\left[\sum_{t=0}^{\infty} \gamma^t \left(\gamma V^{\pi_\theta}(s_{t+1}) - V^{\pi_\theta}(s_t)\right)\right] && (41)\\
&= \mathbb{E}_{\tau \sim p_{\theta'}(\tau)}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right] + \mathbb{E}_{\tau \sim p_{\theta'}(\tau)}\left[\sum_{t=0}^{\infty} \gamma^t \left(\gamma V^{\pi_\theta}(s_{t+1}) - V^{\pi_\theta}(s_t)\right)\right] && (42)\\
&= \mathbb{E}_{\tau \sim p_{\theta'}(\tau)}\left[\sum_{t=0}^{\infty} \gamma^t \left(r(s_t, a_t) + \gamma V^{\pi_\theta}(s_{t+1}) - V^{\pi_\theta}(s_t)\right)\right] && (43)\\
&= \mathbb{E}_{\tau \sim p_{\theta'}(\tau)}\left[\sum_{t=0}^{\infty} \gamma^t A^{\pi_\theta}(s_t, a_t)\right] && (44)
\end{aligned}
$$

where in eq. 38 we changed the expectation (which is valid because τ sampled from any distribution, including $p_{\theta'}(\tau)$, always contains s_0), in eq. 40 and eq. 41 we applied the standard telescoping sum trick, and in eq. 44 we invoked the standard definition of the advantage function. What this final equation tells us is that $J(\theta') - J(\theta)$ is equal to the advantage of the old policy in expectation on trajectories sampled from the new.

As you might observe, the RHS in eq. 44 involves an expectation over trajectories sampled from the latest policy $\pi_{\theta'}$. As per our plan, we will next find a lower bound whose expectation is over $\tau \sim p_\theta(\tau)$, which can allow us to use off-policy samples to optimize θ' (the "off"-policy in question being π_θ). First, let's change the expectation in eq. 44 to be a composition of expectations over states and actions:

$$
\begin{aligned}
\mathbb{E}_{\tau \sim p_{\theta'}(\tau)}\left[\sum_{t=0}^{\infty} \gamma^t A^{\pi_\theta}(s_t, a_t)\right] &= \sum_{t=0}^{\infty} \mathbb{E}_{\tau \sim p_{\theta'}(\tau)}\left[\gamma^t A^{\pi_\theta}(s_t, a_t)\right] && (45)\\
&= \sum_{t=0}^{\infty} \mathbb{E}_{s_t \sim p_{\theta'}(s_t)}\left[\mathbb{E}_{a_t \sim \pi_{\theta'}(a_t|s_t)}\left[\gamma^t A^{\pi_\theta}(s_t, a_t)\right]\right] && (46)\\
&= \sum_{t=0}^{\infty} \mathbb{E}_{s_t \sim p_{\theta'}(s_t)}\left[\mathbb{E}_{a_t \sim \pi_{\theta}(a_t|s_t)}\left[\frac{\pi_{\theta'}(a_t|s_t)}{\pi_\theta(a_t|s_t)} \gamma^t A^{\pi_\theta}(s_t, a_t)\right]\right] && (47)
\end{aligned}
$$

where in eq. 45 we used the linearity of expectations, in eq. 46 we used the fact that although the expectation was originally over trajectories, the term inside the expectation depended only on s_t

and a_t, and in eq. 47 we used importance sampling. The IS weight in eq. 47 is acceptable because it is a single probability ratio, instead of a variance-exploding product we saw before. The remaining step is to try to change the outer expectation to be over $p_\theta(s_t)$ by constructing a lower bound.

To begin constructing the lower bound, we will first prove the following claim: $p_\theta(s_t)$ is close to $p_{\theta'}(s_t)$ when π_θ is close to π'_θ.

Claim: $p_\theta(s_t)$ is close to $p_{\theta'}(s_t)$ when π_θ is close to π'_θ.

Proof. We will say that π_θ is close to π'_θ if $|\pi_{\theta'}(a_t|s_t) - \pi_\theta(a_t|s_t)| \leq \epsilon$ for all s_t, where $|\cdot|$ is the total variational divergence measure (to illustrate TVD: if the two distributions are categorical, then TVD is the sum of absolute values of differences of the probabilities of each bin). Next, we are going to use the following lemma (which was proved in a prior work): if $|p_\mathcal{X}(x) - p_\mathcal{Y}(x)| = \epsilon, \exists.p(x,y)$ such that $p(x) = p_\mathcal{X}(x)$ and $p(y) = p_\mathcal{Y}(y)$ and $p(x = y) = 1 - \epsilon$. I.e., if the TVD between two distributions $p_\mathcal{X}$ and $p_\mathcal{Y}$ over the same space (whose samples we will denote by x) is equal to ϵ, then there exists a joint distribution over two random variables whose marginal over the first is distributed according to $p_\mathcal{X}$, and whose marginal over the second is distributed according to $p_\mathcal{Y}$, and where $p(x = y) = 1 - \epsilon$. This lemma is useful because it translates a TVD assumption into an equality assumption. Concretely, plugging in $\pi_{\theta'}$ and π_θ for $p_\mathcal{X}$ and $p_\mathcal{Y}$, our initial $|\pi_{\theta'}(a_t|s_t) - \pi_\theta(a_t|s_t)| \leq \epsilon$ assumption allows us to say that $\pi_{\theta'}(a_t|s_t) \neq \pi_\theta(a_t|s_t)$ with probability at most ϵ. If over a sequence of t decisions $\pi_{\theta'}$ does not make a "mistake", and predict a different action than π_θ, then the state distribution $p_{\theta'}(s_t) = p_\theta(s_t)$. More generally, $\pi_{\theta'}(a_t|s_t) \neq \pi_\theta(a_t|s_t)$ with probability at most ϵ means that $p_{\theta'}(s_t) = (1-\epsilon)^t p_\theta(s_t) + (1 - (1-\epsilon)^t) p_{\text{mistake}}(s_t)$. Then, $|p_{\theta'}(s_t) - p_\theta(s_t)| = |(1-\epsilon)^t p_\theta(s_t) + (1-(1-\epsilon)^t) p_{\text{mistake}}(s_t) - p_\theta(s_t)| = |p_\theta(s_t)((1-\epsilon)^t - 1) + (1-(1-\epsilon)^t) p_{\text{mistake}}(s_t)| = |(1-(1-\epsilon)^t) p_{\text{mistake}}(s_t) - (1-(1-\epsilon)^t) p_\theta(s_t)| = (1-(1-\epsilon)^t)|p_{\text{mistake}}(s_t) - p_\theta(s_t)| \leq 2(1-(1-\epsilon)^t)$ because the maximum TVD between any two distributions is two, and we don't know anything about $p_{\text{mistake}}(s_t)$ so we conservatively assume maximum TVD. Now, for $\epsilon \in [0,1], (1-\epsilon)^t \geq 1 - \epsilon t$ (can you prove this?), and so our bound on the state distribution mismatch becomes $|p_{\theta'}(s_t) - p_\theta(s_t)| \leq 2(1-(1-\epsilon)^t) \leq 2\epsilon t$. □

Now that we have a bound on the TVD between the state marginals given an assumption on the TVD between the policies, we can proceed to bound the RHS of eq. 47 and the version of the RHS with the outer expectation over $p_\theta(s_t)$ instead of $p_{\theta'}(s_t)$.

First note the following:

$$\mathbb{E}_{p_{\theta'}(s_t)}[f(s_t)] = \sum_{s_t} p_{\theta'}(s_t) f(s_t) \tag{48}$$

$$= \sum_{s_t} p_\theta(s_t) f(s_t) + \sum_{s_t} (p_{\theta'}(s_t) - p_\theta(s_t)) f(s_t) \tag{49}$$

$$\geq \sum_{s_t} p_\theta(s_t) f(s_t) - \left| \sum_{s_t} (p_{\theta'}(s_t) - p_\theta(s_t)) f(s_t) \right| \tag{50}$$

$$\geq \sum_{s_t} p_\theta(s_t) f(s_t) - \sum_{s_t} |p_{\theta'}(s_t) - p_\theta(s_t)| \, |f(s_t)| \tag{51}$$

$$\geq \sum_{s_t} p_\theta(s_t) f(s_t) - \left(\max_{s_t} |f(s_t)| \right) \sum_{s_t} |p_{\theta'}(s_t) - p_\theta(s_t)| \tag{52}$$

$$= \mathbb{E}_{p_\theta(s_t)}[f(s_t)] - \left(\max_{s_t} |f(s_t)| \right) |p_{\theta'}(s_t) - p_\theta(s_t)| \tag{53}$$

where in eq. 53 we replaced $\sum_{s_t} |p_{\theta'}(s_t) - p_\theta(s_t)|$, the TVD between $p_{\theta'}(s_t)$ and $p_\theta(s_t)$, with the same thing in shorthand notation, $|p_{\theta'}(s_t) - p_\theta(s_t)|$. Plugging in the bound we derived for $|p_{\theta'}(s_t) - p_\theta(s_t)|$,

$$\mathbb{E}_{p_{\theta'}(s_t)}[f(s_t)] \geq \mathbb{E}_{p_\theta(s_t)}[f(s_t)] - 2\epsilon t \left(\max_{s_t} |f(s_t)| \right) \tag{54}$$

Now we can directly use this to get the bound for $J(\theta') - J(\theta)$:

$$J(\theta') - J(\theta) = \sum_t \mathbb{E}_{s_t \sim p_{\theta'}(s_t)} \left[\mathbb{E}_{a_t \sim \pi_\theta(a_t|s_t)} \left[\frac{\pi_{\theta'}(a_t|s_t)}{\pi_\theta(a_t|s_t)} \gamma^t A^{\pi_\theta}(s_t, a_t) \right] \right] \tag{55}$$

$$\geq \sum_t \mathbb{E}_{s_t \sim p_\theta(s_t)} \left[\mathbb{E}_{a_t \sim \pi_\theta(a_t|s_t)} \left[\frac{\pi_{\theta'}(a_t|s_t)}{\pi_\theta(a_t|s_t)} \gamma^t A^{\pi_\theta}(s_t, a_t) \right] \right] - \sum_t 2\epsilon t C \tag{56}$$

where C is the maximum value our "f" can take on. "f" in our case is an expected value of an importance weight (which in expectation will become 1) times an advantage, which has maximum value of the horizon (H) times the maximum possible reward. So

$$J(\theta') - J(\theta) \geq \sum_t \mathbb{E}_{s_t \sim p_\theta(s_t)} \left[\mathbb{E}_{a_t \sim \pi_\theta(a_t|s_t)} \left[\frac{\pi_{\theta'}(a_t|s_t)}{\pi_\theta(a_t|s_t)} \gamma^t A^{\pi_\theta}(s_t, a_t) \right] \right] - \sum_t 2\epsilon t O(H r_{\max}) \tag{57}$$

Great! We've now successfully derived a lower bound of $J(\theta') - J(\theta)$ whose expectation is over θ, allowing us to optimize it w.r.t. θ' using samples only from θ. And as long as the TVD $|\pi_{\theta'}(a_t|s_t) - \pi_\theta(a_t|s_t)| \leq \epsilon$ for small enough ϵ, then we can be sure that the error term $\sum_t 2\epsilon t O(H r_{\max})$ is small, and so when we optimize our lower bound, we are indeed also optimizing the original objective $J(\theta') - J(\theta)$.

With this, our objective becomes:

$$\theta' \leftarrow \arg\max_{\theta'} \sum_t \mathbb{E}_{s_t \sim p_\theta(s_t)} \left[\mathbb{E}_{a_t \sim \pi_\theta(a_t|s_t)} \left[\frac{\pi_{\theta'}(a_t|s_t)}{\pi_\theta(a_t|s_t)} \gamma^t A^{\pi_\theta}(s_t, a_t) \right] \right] \tag{58}$$

subject to the constraint

$$|\pi_{\theta'}(a_t|s_t) - \pi_\theta(a_t|s_t)| \leq \epsilon \tag{59}$$

Now remember from section 1 that we generally like to work with KL divergences due to their many nice properties. Indeed in this case as well, we would like to re-write this TVD constraint as a KL constraint, because it will make subsequent constrained optimization easier. There is a known relationship between KL divergences and TVDs that we can exploit for this, namely that

$$|p_{\mathcal{A}} - p_{\mathcal{B}}| \leq \sqrt{\frac{1}{2} D_{\mathrm{KL}}\left(p_{\mathcal{A}} \| p_{\mathcal{B}}\right)} \tag{60}$$

So if we constrain KL, we will be constraining an upper bound on TVD, meaning we can simply swap our TVD constraint for a KL constraint. This gives us the following constrained optimization problem:

$$\theta' \leftarrow \arg\max_{\theta'} \sum_t \mathbb{E}_{s_t \sim p_\theta(s_t)} \left[\mathbb{E}_{a_t \sim \pi_\theta(a_t|s_t)} \left[\frac{\pi_{\theta'}(a_t|s_t)}{\pi_\theta(a_t|s_t)} \gamma^t A^{\pi_\theta}(s_t, a_t) \right] \right] \tag{61}$$

subject to the constraint

$$D_{\mathrm{KL}}\left(\pi_{\theta'}(a_t|s_t) \| \pi_\theta(a_t|s_t)\right) \leq \epsilon \tag{62}$$

A standard way to solve a constrained optimization problem when the constraint is an equality constraint is to set up an unconstrained optimization problem that includes the original objective minus a factor λ times the equality constraint, and we optimize the original parameters plus the parameter λ jointly in the unconstrained optimization problem. Concretely, we will make our KL divergence inequality constraint into an equality constraint, $D_{\mathrm{KL}}\left(\pi_{\theta'}(a_t|s_t) \| \pi_\theta(a_t|s_t)\right) = \epsilon$, and then we will maximize the following objective w.r.t. the parameters θ' and λ:

$$\mathcal{L}(\theta', \lambda) = \sum_t \mathbb{E}_{s_t \sim p_\theta(s_t)} \left[\mathbb{E}_{a_t \sim \pi_\theta(a_t|s_t)} \left[\frac{\pi_{\theta'}(a_t|s_t)}{\pi_\theta(a_t|s_t)} \gamma^t A^{\pi_\theta}(s_t, a_t) \right] \right] - \lambda \left(D_{\mathrm{KL}}\left(\pi_{\theta'}(a_t|s_t) \| \pi_\theta(a_t|s_t)\right) - \epsilon \right) \tag{63}$$

We will optimize this objective using something called **dual gradient ascent/descent**. In particular, we will alternate the following procedure:

Algorithm 2 Primal–dual update

1: $\theta' \leftarrow \arg\max_{\theta'} \ \mathcal{L}(\theta', \lambda)$
2: $\lambda \leftarrow \lambda + \alpha\big(D_{\mathrm{KL}}\big(\pi_{\theta'}(a_t \mid s_t) \,\|\, \pi_\theta(a_t \mid s_t)\big) - \epsilon\big)$

where α is a learning rate for λ. In essence what this procedure is doing is optimizing θ' to simultaneously maximize the original unconstrained objective and minimize the KL constraint to θ, where the relative magnitude of these two losses is determined by λ. If the KL constraint is being violated significantly, i.e., if $D_{\mathrm{KL}}\big(\pi_{\theta'}(a_t \mid s_t) \,\|\, \pi_\theta(a_t \mid s_t)\big) - \epsilon$ is large, then it is likely that λ is too small, and the KL part of the loss is effectively being ignored with the model just optimizing the original unconstrained objective, and so the dual optimization step (step 2) increases the KL constraint weight λ. Similarly if the objective is being satisfied, λ can be lowered. Dual gradient descent is an elegant way to turn general constrained optimization problems into amenable unconstrained optimization problems.

And we're done! We now have an algorithm that can make use of slightly off-policy data, boosting sample efficiency. The way you would use an algorithm like this in practice is to iterate (1) collecting samples with the latest policy, and (2) doing multiple gradient updates (instead of just 1 like in REINFORCE) on the same batch of data, using dual gradient descent on the KL-constrained off-policy objective (eqs. 61, 62). **Exercise for you:** differentiate eq. 63 w.r.t. θ'. Hint: first find a way to combine the expectations and move it outside the summation, and use the closed form of the KL from section 1. Does the resultant gradient look familiar?

The KL constraint helps with more than just enabling more sample-efficient learning; it helps stabilize policy optimization by enforcing a so-called "trust region". The KL constraint can also be interpreted as a way to "naturalize" the gradient to reflect the direction to best update the policy (see Natural Policy Gradient for more on this). See Proximal Policy Optimization (PPO) for a modern practical slightly-off-policy policy iteration/gradient RL algorithm that forgoes the complexity of dual gradient descent.