Policy Gradient Mathematics

Prof Richard Yi Da Xu
richardxu.com
Hong Kong Baptist University

June 24, 2026

1 Introduction

This document covers the following topics:

1. Policy Gradient Theorem
2. Mathematics on Trusted Region Optimization in RL
3. Natural Gradients on TRPO
4. Proximal Policy Optimization (PPO)
5. Conjugate Gradient Algorithm

This lecture is referenced heavily from:

- https://lilianweng.github.io/lil-log/2018/04/08/policy-gradient-algorithms.html. I borrowed it heavily, please check her goodies on RL and GAN
- https://medium.com/@jonathan_hui/rl-trust-region-policy-optimization-trpo-explained-a6ee04eeeee9 Jonathan Hui's blog. Again, lots of goodies.
- http://www.cs.cmu.edu/~pradeepr/convexopt/Lecture_Slides/conjugate_direction_methods.pdf

2 Preliminaries for the Policy Gradient Theorem

Let's start with the solution first:

$$\begin{aligned} J(\theta) &= \sum_{s\in\mathcal{S}} \rho^{\pi}(s) V^{\pi}(s) \\ &= \sum_{s\in\mathcal{S}} \rho^{\pi}(s) \sum_{a\in\mathcal{A}} \pi_{\theta}(a|s) Q^{\pi}(s,a) \end{aligned} \tag{1}$$

$$\begin{aligned}\nabla_\theta J(\theta) &\propto \sum_{s\in\mathcal{S}} \rho^\pi(s) \sum_{a\in\mathcal{A}} Q^\pi(s,a) \nabla_\theta \pi_\theta(a|s) \\ &= \mathbb{E}_\pi\left[Q^\pi(s,a) \nabla_\theta \log \pi_\theta(a|s)\right]\end{aligned} \tag{2}$$

let's look at the $\mathbb{E}_\pi$ part, it is not: $\mathbb{E}_{a\sim\pi_\theta(\cdot|s)}$, but instead is $\mathbb{E}_{a\sim\pi_\theta(\cdot|s),s\sim\rho^\pi}$, so one may also write it as:

$$\nabla_\theta J(\theta) \propto \mathbb{E}_{a\sim\pi_\theta(\cdot|s),s\sim\rho^\pi}\left[Q^\pi(s,a) \nabla_\theta \log \pi_\theta(a|s)\right] \tag{3}$$

Each sample (s,a) collected from the agent's interaction with the environment contributes one term $Q^\pi(s,a)\,\nabla_\theta \log \pi_\theta(a|s)$ to the Monte-Carlo estimate of this expectation. A high Q^π value pushes θ strongly towards that action, weighted by how often the agent visits the corresponding state ($\rho^\pi(s)$) — this is why state-visitation frequency matters and not just the per-step reward.

Why does $Q^\pi(s,a)$ act as a multiplier that pushes towards action a? The gradient update for θ is

$$\theta \;\leftarrow\; \theta + \alpha \cdot \underbrace{Q^\pi(s,a)}_{\text{how good?}} \cdot \underbrace{\nabla_\theta \log \pi_\theta(a|s)}_{\text{in which direction?}}$$

In other words, Q^π and $\nabla_\theta \log \pi_\theta$ are matched: the direction to push θ is determined entirely by the score function, and the strength of that push is determined entirely by Q^π. This multiplicative matching is the reason the policy gradient theorem is so elegant — good actions are reinforced proportionally to their long-run value, without ever needing a model of the environment.

But wait — how does a single sample (s,a) tell you $Q^\pi(s,a)$? Recall that $Q^\pi(s,a)$ is defined as the expected cumulative discounted reward when starting from state s, taking action a, and thereafter following policy π:

$$Q^\pi(s,a) = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \;\middle|\; S_t = s,\, A_t = a\right].$$

This involves the future — you need to know all rewards $R_{t+1}, R_{t+2}, R_{t+3}, \ldots$ that follow. A single (s,a) pair alone cannot tell you this.
In practice, $Q^\pi(s,a)$ is estimated using one of two approaches:

1. Monte-Carlo return. After taking action a in state s, the agent plays out the rest of the episode and records the discounted sum of all future rewards:

$$\hat{G}_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots$$

This $\hat{G}_t$ is an unbiased sample estimate of $Q^\pi(s_t, a_t)$. So the "single sample" is really $(s_t, a_t, \hat{G}_t)$ — you need the full trajectory tail to form it.

2. Critic / function approximator. A separate neural network (the "critic") $\hat{Q}_w(s, a)$ is trained on past experience to predict the Q-value given only the current (s, a). This is the idea behind Actor-Critic methods (covered later).

Either way, the key insight is: the gradient formula uses $Q^\pi(s, a)$ in expectation, and a single trajectory sample — with its attached future return — provides an unbiased (though noisy) estimate of that expectation.

Numerical Example. Suppose the agent lives in a tiny world with two states $\mathcal{S} = \{s_1, s_2\}$ and two actions $\mathcal{A} = \{\text{left}, \text{right}\}$. The stationary state distribution is

$$\rho^\pi(s_1) = 0.6, \qquad \rho^\pi(s_2) = 0.4.$$

The parameterised policy (e.g. a softmax with a single parameter θ) gives

$$\pi_\theta(\text{left}|s_1) = 0.7, \quad \pi_\theta(\text{right}|s_1) = 0.3,$$
$$\pi_\theta(\text{left}|s_2) = 0.4, \quad \pi_\theta(\text{right}|s_2) = 0.6.$$

And suppose the Q-values are

$$Q^\pi(s_1, \text{left}) = 5, \quad Q^\pi(s_1, \text{right}) = 2,$$
$$Q^\pi(s_2, \text{left}) = 3, \quad Q^\pi(s_2, \text{right}) = 8.$$

For a softmax policy $\pi_\theta(a|s) = \frac{e^{\theta_a}}{\sum_{a'} e^{\theta_{a'}}}$, the score function is $\nabla_\theta \log \pi_\theta(a|s) = \mathbf{1}_{[\text{chosen action}]} - \pi_\theta(\cdot|s)$, which is a vector of dimension $|\mathcal{A}|$. For simplicity, treat $\nabla_\theta \log \pi_\theta(a|s)$ as a scalar placeholder and write the expectation explicitly as:

$$\mathbb{E}_{a\sim\pi_\theta(\cdot|s),\, s\sim\rho^\pi}\Big[Q^\pi(s,a)\,\nabla_\theta \log \pi_\theta(a|s)\Big]$$
$$= \sum_{s\in\mathcal{S}} \rho^\pi(s) \sum_{a\in\mathcal{A}} \pi_\theta(a|s)\, Q^\pi(s,a)\, \nabla_\theta \log \pi_\theta(a|s).$$

Expanding the double sum state-by-state:

$$\begin{aligned} s_1 \text{ term:} \quad & \rho^\pi(s_1)\Big[\pi_\theta(\text{L}|s_1)\, Q^\pi(s_1, \text{L})\, \nabla_\theta \log \pi_\theta(\text{L}|s_1) \\ & \quad + \pi_\theta(\text{R}|s_1)\, Q^\pi(s_1, \text{R})\, \nabla_\theta \log \pi_\theta(\text{R}|s_1)\Big] \\ & = 0.6\Big[0.7 \times 5 \times \nabla_\theta \log(0.7) + 0.3 \times 2 \times \nabla_\theta \log(0.3)\Big], \\ s_2 \text{ term:} \quad & \rho^\pi(s_2)\Big[\pi_\theta(\text{L}|s_2)\, Q^\pi(s_2, \text{L})\, \nabla_\theta \log \pi_\theta(\text{L}|s_2) \\ & \quad + \pi_\theta(\text{R}|s_2)\, Q^\pi(s_2, \text{R})\, \nabla_\theta \log \pi_\theta(\text{R}|s_2)\Big] \\ & = 0.4\Big[0.4 \times 3 \times \nabla_\theta \log(0.4) + 0.6 \times 8 \times \nabla_\theta \log(0.6)\Big]. \end{aligned}$$

2.1 Why is this Theorem so Significant?

Before this theorem, calculating the gradient of a policy was considered extremely difficult because the environment's dynamics are usually unknown: we do not know the probability of moving from state s to state s' after taking action a. In a game, for example, you usually do not know exactly how the opponent will respond.

The significance of this specific proportionality is that:

- Model-Free Learning: You do not need a model of the environment (physics, rules, transition probabilities) to calculate this. You only need the samples of states s, actions a, and rewards r collected by the agent.
- Stability: It allows us to optimize stochastic policies (where the agent outputs probabilities rather than a hard decision), which is essential for complex environments.
- it allows us to compute the $\nabla_\theta J(\theta)$ using Monte-Carlo integration.

3 Policy Gradient Theorem

Before we dive into the Policy Gradient Theorem, we need to understand the policy π, the V-function, and the Q-function.

3.1 The Policy π

A policy π is the agent's strategy: it tells the agent what to do (which action to take) in any given state. Formally, a policy is a mapping from states to a probability distribution over actions:

$$\pi(a \mid s) \;=\; \Pr\big[\, A_t = a \,\big|\, S_t = s \,\big], \qquad \sum_{a \in \mathcal{A}} \pi(a \mid s) = 1.$$

We distinguish two common forms:

- Deterministic policy: $a = \pi(s)$, a single action is chosen for each state.
- Stochastic policy: $\pi(a \mid s)$ assigns a probability to every action; the agent samples an action from this distribution.

In policy gradient methods, we work with a parameterised stochastic policy $\pi_\theta(a \mid s)$, where θ is a vector of learnable parameters (e.g. the weights of a neural network). Our goal is to find θ that maximises the expected return:

$$\theta^* \;=\; \arg\max_\theta \; J(\theta), \qquad J(\theta) \;=\; \mathbb{E}_{\pi_\theta}\left[\sum_{t \geq 0} \gamma^t R_{t+1} \right].$$

Intuitively, $\pi_\theta(a \mid s)$ is what the agent does, while $V^\pi(s)$ and $Q^\pi(s, a)$ (introduced next) measure how good the resulting behaviour is.

3.2 The V-function (State-Value Function) and the Q-function (Action-Value Function)

In Reinforcement Learning, the V-function (State-Value function) and the Q-function (Action-Value function) are fundamental concepts used to evaluate how good a particular state or action is for an agent. They quantify the expected return (cumulative discounted reward).

3.2.1 The V-function (State-Value Function)

The State-Value function, denoted as $V^\pi(s)$, measures "how good" it is for an agent to be in a specific state s while following a specific policy π. It represents the expected return starting from state s and following policy π thereafter. Formally:

$$V^\pi(s) = \mathbb{E}_\pi \left[G_t \mid S_t = s \right] \tag{4}$$

where the return G_t is the discounted sum of all future rewards from time step t:

$$G_t \;=\; \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \;=\; R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots \tag{5}$$

Substituting (5) into (4) gives the expanded definition:

$$\begin{aligned}
V^\pi(s) &= \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \;\middle|\; S_t = s \right] \qquad (6)\\
&= \mathbb{E}_\pi \left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots \;\middle|\; S_t = s \right] \\
&= \mathbb{E}_\pi \left[R_{t+1} + \gamma \big(R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} + \cdots \big) \;\middle|\; S_t = s \right] \\
&= \mathbb{E}_\pi \left[R_{t+1} + \gamma G_{t+1} \;\middle|\; S_t = s \right] \\
&= \sum_a \Pr(A_t = a \mid S_t = s)\, \mathbb{E}_\pi \left[R_{t+1} + \gamma G_{t+1} \;\middle|\; S_t = s, A_t = a \right] \\
&= \sum_a \pi(a|s)\, \mathbb{E}_\pi \left[R_{t+1} + \gamma G_{t+1} \;\middle|\; S_t = s, A_t = a \right] \\
&= \sum_a \pi(a|s) \sum_{s',r} \Pr(S_{t+1} = s', R_{t+1} = r \mid S_t = s, A_t = a) \\
&\qquad\qquad \times \mathbb{E}_\pi \left[R_{t+1} + \gamma G_{t+1} \;\middle|\; S_t = s, A_t = a, S_{t+1} = s', R_{t+1} = r \right] \\
&= \sum_a \pi(a|s) \sum_{s',r} p(s', r \mid s, a) \left(r + \gamma \mathbb{E}_\pi \left[G_{t+1} \mid S_{t+1} = s' \right] \right) \\
&= \sum_a \pi(a|s) \sum_{s',r} p(s', r \mid s, a) \big(r + \gamma \underbrace{V^\pi(s')}_{\text{value of next state } s'} \big). \qquad (7)
\end{aligned}$$

The steps above use two standard ideas. First, the return is split into the immediate reward R_{t+1} plus the discounted future return G_{t+1}. Second, the expectation is expanded by conditioning on the one-step events: the policy first chooses $A_t = a$ with probability $\pi(a|s)$, then the environment produces the next state and reward $(S_{t+1}, R_{t+1}) = (s', r)$ with probability $p(s', r \mid s, a)$. After reaching s', the expected remaining return is exactly $V^\pi(s')$ by the definition of the value function. This one-step "bootstrapping" — V^π appears on both sides — is exactly what dynamic programming (and later, TD learning) exploits.

Note that there are three random variables in the above equation, and their associated conditions (or constraints):

- a, the action to take in state s,
- s', the next state, after taking action a,
- r, the reward received after taking action a and landing in state s'.

3.2.2 The Q-function (Action-Value Function)

The Action-Value function, denoted as $Q_\pi(s, a)$, measures "how good" it is for an agent to take a specific action a while in a specific state s, and then follow policy π thereafter.

While $V(s)$ tells you the value of the state, $Q(s, a)$ tells you the value of a specific choice within that state.

Formally, the action-value function $Q_\pi(s, a)$ is defined as the expected return starting from state s, taking action a, and then following policy π:

$$Q^\pi(s, a) = \mathbb{E}_\pi \left[G_t \mid S_t = s, A_t = a \right] \tag{8}$$

Expanded with the return:

$$Q^\pi(s, a) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \mid S_t = s, A_t = a \right] \tag{9}$$

Then, the Bellman Equation for Q^π is:

$$Q^\pi(s, a) = \sum_{s', r} p(s', r \mid s, a) \big(r + \gamma \underbrace{\sum_{a'} \pi(a'|s') Q^\pi(s', a')}_{V^\pi(s')} \big) \tag{10}$$

In some places of this notes, w.l.o.g, we removed γ to make the equations more readable.

3.2.3 Relationship Between V and Q

The V-function and Q-function are tightly coupled. The value of a state is simply the expected value of the actions you can take from that state, weighted by the probability of taking them under policy π.

1. V in terms of Q:

$$V^{\pi}(s) = \sum_{a} \pi(a|s) Q^{\pi}(s,a) \tag{11}$$

   (The value of state s is the average Q-value of all possible actions in that state).

2. Q in terms of V:

$$Q^{\pi}(s,a) = \sum_{s',r} p(s', r \mid s, a) \left[r + \gamma V^{\pi}(s') \right] \tag{12}$$

   (The value of taking action a is the immediate reward plus the discounted value of the next state).

3.2.4 Optimal Value Functions

In RL, the goal is often to find the optimal policy π^*. The optimal value functions are defined as the maximum return achievable by any policy.

- Optimal State-Value Function (V^*):

$$V^*(s) = \max_{\pi} V_{\pi}(s)$$

- Optimal Action-Value Function (Q^*):

$$Q^*(s,a) = \max_{\pi} Q_{\pi}(s,a)$$

The relationship between the optimal functions is:

$$V^*(s) = \max_{a} Q^*(s,a)$$

This equation states that the value of an optimal state is equal to the value of the best possible action taken from that state.

3.3 Start to derive the Policy Gradient Theorem

We use a representation:

$$J(\theta) \equiv V^{\pi}(s_0) \tag{13}$$

which says that the expected reward from the start state s_0 is the value function $V^{\pi}(s_0)$. Without explicit knowledge of a specific s_0, the objective can also be written by averaging over the state space $s \in \mathcal{S}$ and then over the action space $a \in \mathcal{A}$:

$$\begin{aligned} J(\theta) &= \sum_{s \in \mathcal{S}} \rho^{\pi}(s) V^{\pi}(s) \\ &= \sum_{s \in \mathcal{S}} \rho^{\pi}(s) \sum_{a \in \mathcal{A}} \pi_{\theta}(a|s) Q^{\pi}(s,a) \end{aligned} \tag{14}$$

Basically, it says that the expected reward is the expected value of $V^{\pi}(s)$ under the stationary state distribution induced by the policy π_{θ}.

where $\rho^{\pi}(s)$ is the stationary distribution of the Markov chain induced by π_{θ}, i.e.,

$$\rho^{\pi}(s) = \lim_{t \to \infty} P(s_t = s | s_0, \pi_{\theta}) \tag{15}$$

This means that regardless of the start state s_0, if one travels along the Markov chain forever, the probability of being in state s eventually becomes stable; this limiting probability is the stationary probability $\rho^{\pi}(s)$.

Computing gradient $\nabla_{\theta} J(\theta)$ is difficult because it depends on both:

1. action selection directly determined by π_{θ}, and
2. stationary state $\rho^{\pi}(s)$ following action selection behavior indirectly determined by π_{θ}

Therefore, it is difficult to estimate policy update effect on state distribution for generally unknown environment. However, Policy gradient theorem states:

$$\begin{aligned} \nabla_{\theta} J(\theta) &= \nabla_{\theta} \Big(\sum_{s \in \mathcal{S}} \rho^{\pi}(s) \sum_{a \in \mathcal{A}} Q^{\pi}(s,a) \pi_{\theta}(a|s) \Big) \\ &\propto \sum_{s \in \mathcal{S}} \rho^{\pi}(s) \sum_{a \in \mathcal{A}} Q^{\pi}(s,a) \nabla_{\theta} \pi_{\theta}(a|s) \end{aligned} \tag{16}$$

Significance : the above objective function does not involve derivative of state distribution $\rho^{\pi}(.)$

3.4 Proof of Policy Gradient Theorem

We want a policy to maximize $J(\theta) \equiv V^\pi(s_0)$, and the first step is always to write:

$$V^\pi(s_0) = \sum_{a\in\mathcal{A}} \pi_\theta(a|s) Q^\pi(s_0, a) \tag{17}$$

You see that the above summation does not contain the term $\sum_{s\in\mathcal{S}} \rho^\pi(s)$, however, during the derivation of the policy gradient theorem, we will "add" it in the final equation, i.e., Eq. (16).

we start by writing it in a more general form, of letting $s_0 \to s$:

$$\begin{aligned}
\nabla_\theta V^\pi(s) &= \nabla_\theta \Big(\sum_{a\in\mathcal{A}} \underbrace{\pi_\theta(a|s)}_{u} \underbrace{Q^\pi(s,a)}_{v} \Big) \\
&= \underbrace{\sum_{a\in\mathcal{A}} \Big(\nabla_\theta \pi_\theta(a|s) Q^\pi(s,a) }_{=\phi(s) \text{ which contain } \nabla_\theta \text{we leave it alone for now}} + \underbrace{\sum_{a\in\mathcal{A}} \pi_\theta(a|s) \nabla_\theta Q^\pi(s,a) }_{\text{see how we make } \nabla_\theta \text{ disappear in this term}} \Big)
\end{aligned} \tag{18}$$

Up to here, you can foresee that we want to make the $\nabla_\theta Q^\pi(s,a)$ term disappear.

To do so, we can now write the following recursive equation for $Q^\pi(s,a)$, where we only extend the reward to the next state s' (so that it is to be handled by $V^\pi(s')$) and the immediate reward r.

$$Q^\pi(s,a) = \sum_{s'} \sum_{r} P(s', r|s, a) \big(\underbrace{r}_{\text{immediate reward}} + \underbrace{V^\pi(s')}_{\text{future reward}} \big) \tag{19}$$

$$\begin{aligned}
&= \phi(s) + \sum_{a\in\mathcal{A}} \Big(\pi_\theta(a|s) \nabla_\theta \sum_{s'} \sum_{r} P(s', r|s, a) \big(r + V^\pi(s') \big) \Big) \\
&= \phi(s) + \sum_{a\in\mathcal{A}} \Big(\pi_\theta(a|s) \sum_{s'} \sum_{r} P(s', r|s, a) \nabla_\theta V^\pi(s') \Big) \quad \text{move } \nabla_\theta \text{ inside the sum over } s' \\
&= \phi(s) + \sum_{a\in\mathcal{A}} \Big(\pi_\theta(a|s) \sum_{s'} P(s'|s, a) \nabla_\theta V^\pi(s') \Big) \quad \text{retain marginal by integrate out } r
\end{aligned} \tag{20}$$

Let $P^\pi(s \to s', t)$ to be the probability of transition from state $s \to s'$ in t steps, and can be written as, and by integrating out the action $a \in \mathcal{A}$, and let $t = 1$, we get:

$$P^\pi(s \to s', 1) = \sum_{a\in\mathcal{A}} \pi_\theta(a|s) P(s'|s, a) \tag{21}$$

meaning that "regardless" of the action a taken, the probability of transitioning from state s to state s' in 1 step is $\rho^\pi(s \to s', 1)$.

$$
\begin{aligned}
\nabla_\theta V^\pi(s) =& \phi(s) + \sum_a \pi_\theta(a|s) \sum_{s'} P(s'|s,a) \nabla_\theta V^\pi(s') \\
=& \phi(s) + \sum_{s'} \sum_a \pi_\theta(a|s) P(s'|s,a) \nabla_\theta V^\pi(s') && \text{switch the two summation places} \\
=& \phi(s) + \sum_{s'} P^\pi(s \to s', 1) \nabla_\theta V^\pi(s') && \text{substitute (21)} \\
=& \phi(s) + \sum_{s'} P^\pi(s \to s', 1) \Big[\phi(s') + \sum_{s''} P^\pi(s' \to s'', 1) \nabla_\theta V^\pi(s'') \Big] \\
& && \text{by induction, expand this recursion: } s \to s' \text{ and } s' \to s'' \\
=& \phi(s) + \sum_{s'} P^\pi(s \to s', 1) \phi(s') + \sum_{s'} \sum_{s''} P^\pi(s \to s', 1) P^\pi(s' \to s'', 1) \nabla_\theta V^\pi(s'') \\
=& \phi(s) + \sum_{s'} P^\pi(s \to s', 1) \phi(s') + \underbrace{\sum_{s''} P^\pi(s \to s'', 2)}_{\text{Repeatedly expand}} \nabla_\theta V^\pi(s'')
\end{aligned}
\tag{22}
$$

After repeatedly expand, we get:

$$
\begin{aligned}
=& \underbrace{P^\pi(s \to s, 0)}_{=1} \phi(s) + \sum_{s^{(1)} \in \mathcal{S}} P^\pi(s \to s^{(1)}, 1) \phi(s^{(1)}) + \sum_{s^{(2)} \in \mathcal{S}} P^\pi(s \to s^{(2)}, 2) \phi(s^{(2)}) + \dots \\
=& \sum_{t=0}^{\infty} \sum_{s^{(t)} \in \mathcal{S}} P^\pi(s \to s^{(t)}, t) \phi(s^{(t)})
\end{aligned}
\tag{23}
$$

Let's change some variables, and let $s \to s_0$, $s^{(t)} \to s$. It's important to note that the choice of s_0 is arbitrary, as it won't appear in the final equation in Eq. (24) below:

$$
\begin{aligned}
J(\theta) =& \nabla_\theta V^\pi(s_0) \\
=& \sum_{t=0}^{\infty} \sum_{s \in \mathcal{S}} P^\pi(s_0 \to s, t) \phi(s) \\
=& \sum_{s \in \mathcal{S}} \underbrace{\sum_{t=0}^{\infty} P^\pi(s_0 \to s, t)}_{\eta(s)} \phi(s) \\
\propto& \sum_s \rho^\pi(s) \phi(s) \quad \text{where } \rho^\pi(s) \equiv \frac{\eta(s)}{\sum_s \eta(s)} \text{ is a normalized version of } \eta(s) \\
=& \sum_s \rho^\pi(s) \sum_a \nabla_\theta \pi_\theta(a|s) Q^\pi(s,a)
\end{aligned}
\tag{24}
$$

Remark 1 $\rho^\pi(s)$ can also been seen as the weight of derivative concerning particular s

Remark 2 in words, it says using policy π, which is the probability end up in a particular state s. Also, giving enough t, one may transition from s_0 to any any state s.

Remark 3 As it roll out to fullest, you see the role of $\nabla_\theta V^\pi(s^\infty)$ becomes negligible.

finally, we express the policy gradient formula in terms of $\mathbb{E}_\pi[.]$:

$$
\begin{aligned}
\nabla_\theta J(\theta) \equiv \nabla_\theta V^\pi(s_0) &\propto \sum_{s \in \mathcal{S}} \rho^\pi(s) \sum_{a \in \mathcal{A}} Q^\pi(s,a) \nabla_\theta \pi_\theta(a|s) \\
&= \sum_{s \in \mathcal{S}} \rho^\pi(s) \sum_{a \in \mathcal{A}} \pi_\theta(a|s) Q^\pi(s,a) \underbrace{\frac{\nabla_\theta \pi_\theta(a|s)}{\pi_\theta(a|s)}} \\
&= \underbrace{\sum_{s \in \mathcal{S}} \rho^\pi(s) \sum_{a \in \mathcal{A}} \pi_\theta(a|s)}_{\mathbb{E}_\pi} \left[Q^\pi(s,a) \underbrace{\nabla_\theta \log \pi_\theta(a|s)} \right] \\
&= \mathbb{E}_\pi \left[Q^\pi(s,a) \nabla_\theta \ln \pi_\theta(a|s) \right]
\end{aligned}
\tag{25}
$$

let's reemphasize that we have the final equation:

$$
\nabla_\theta J(\theta) \propto \mathbb{E}_\pi \left[Q^\pi(s,a) \nabla_\theta \ln \pi_\theta(a|s) \right] \tag{26}
$$

3.5 Writing $V^\pi(s)$ and $Q^\pi(s,a)$

- $V^\pi(s)$:

$$
\begin{aligned}
V^{\pi}(s_t) &= \mathbb{E}_{a_t, s_{t+1}, a_{t+1}, \ldots}\left[\sum_{k=0}^{\infty} \gamma^k r(s_{t+k}) \middle| s_t\right] \\
&= \mathbb{E}_{a_t, s_{t+1}, a_{t+1}, \ldots}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} \middle| s_t\right] \quad \text{let } r_{t+k} \equiv r(s_{t+k}) \\
\text{by induction, } V^{\pi}(s_{t+1}) &= \underline{\mathbb{E}_{a_{t+1}, s_{t+2}, a_{t+2}, \ldots}\left[\sum_{k=0}^{\infty} \gamma^k r_{(t+1)+k} \middle| s_{t+1}\right]}
\end{aligned} \tag{27}
$$

note that the last term still summing from $k = 0$ to ∞, but $r_{t+k} \to r_{(t+1)+k}$.

- $Q^{\pi}(s, a)$:

$$
\begin{aligned}
Q^{\pi}(s_t, a_t) &= \mathbb{E}_{s_{t+1}, a_{t+1}, \ldots}\left[\sum_{k=0}^{\infty} \gamma^k r(s_{t+k}) \middle| s_t, a_t\right] \\
&= \mathbb{E}_{s_{t+1}, a_{t+1}, \ldots}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} \middle| s_t, a_t\right] \quad \text{let } r_{t+k} \equiv r(s_{t+k}) \\
&= \mathbb{E}_{s_{t+1}, a_{t+1}, \ldots}\left[r_t + \sum_{k=1}^{\infty} \gamma^k r_{t+k} \middle| s_t, a_t\right] \\
&= \mathbb{E}_{s_{t+1}, a_{t+1}, \ldots}\left[r_t + \gamma \sum_{k=0}^{\infty} \gamma^k r_{(t+1)+k} \middle| s_t, a_t\right] \quad \text{note that } r_{t+k} \to r_{(t+1)+k} \\
&= \mathbb{E}_{s_{t+1}}\Bigg[\underbrace{r_t}_{\text{only } s_{t+1} \text{ affect it}} + \gamma \underline{\mathbb{E}_{a_{t+1}, s_{t+2}, a_{t+2}, \ldots}\left[\sum_{k=0}^{\infty} \gamma^k r_{(t+1)+k} \middle| s_{t+1}\right]} \Bigg| s_t, a_t\Bigg] \\
&= \mathbb{E}_{s_{t+1}}\left[r_t + \gamma V^{\pi}(s_{t+1})\right]
\end{aligned} \tag{28}
$$

3.6 Variance reduction using baseline

Subtract a baseline function $B(s)$ from policy gradient, note $B(s)$ only depends on state s, not depends on action a, we have:

$$
\mathbb{E}_{\pi}\big[\underbrace{Q^{\pi}(s, a)}_{\text{replace with} B(s)} \nabla_{\theta} \ln \pi_{\theta}(a|s)\big] \tag{29}
$$

Therefore we have:

$$\begin{aligned}\mathbb{E}_\pi\big[B(s)\nabla_\theta \ln \pi_\theta(a|s)\big] &= \sum_{s\in\mathcal{S}} \rho^\pi(s) \sum_{a\in\mathcal{A}} \nabla_\theta \pi_\theta(a|s) B(s) \\ &= \sum_{s\in\mathcal{S}} \rho^\pi(s) B(s) \nabla_\theta \sum_{a\in\mathcal{A}} \pi_\theta(a|s) \\ &= 0\end{aligned} \tag{30}$$

A good baseline is $B(s) = V^\pi(s)$:

$$\begin{aligned}\text{without baseline} \quad & \nabla_\theta J(\theta) = \mathbb{E}_\pi\big[Q^\pi(s,a)\nabla_\theta \ln \pi_\theta(a|s)\big] \\ \text{with baseline} \quad & \nabla_\theta J(\theta) = \mathbb{E}_\pi\big[\nabla_\theta \ln \pi_\theta(a|s)(Q^\pi(s,a) - V^\pi(s))\big] \\ & \quad\;\; = \mathbb{E}_\pi\big[\nabla_\theta \ln \pi_\theta(a|s) A^\pi(s,a)\big]\end{aligned} \tag{31}$$

3.7 Writing Advantage function

If we choose Advantage function to be:

$$A^\pi(s,a) = Q^\pi_{\theta_Q}(s,a) - V^\pi_{\theta_V}(s) \tag{32}$$

i.e., if we to construct two neural networks for $Q^\pi_{\theta_Q}$ and $V^\pi_{\theta_V}$, is very inefficient:
Now we can substitute $Q^\pi_{\theta_Q}(s_t, a_t) \equiv \mathbb{E}_{s_{t+1}}\big[r_t + \gamma V^\pi_{\theta_V}(s_{t+1})\big]$:

$$\begin{aligned}A^\pi(s_t,a_t) &= Q^\pi_{\theta_Q}(s_t,a_t) - V^\pi_{\theta_V}(s_t) \\ &= \mathbb{E}_{s_{t+1}}\Big[r_t + \gamma V^\pi_{\theta_V}(s_{t+1})\Big] - V^\pi_{\theta_V}(s_t) \\ &= \mathbb{E}_{s_{t+1}}\Big[r_t + \gamma V^\pi_{\theta_V}(s_{t+1}) - V^\pi_{\theta_V}(s_t)\Big] \quad \text{put } V^\pi_{\theta_V}(s_t) \text{ inside integral won't affect it} \\ A^\pi(s,a) &= \mathbb{E}_{s'\sim P(s'|s,a)}\big[r(s) + \gamma V^\pi_{\theta_V}(s') - V^\pi_{\theta_V}(s)\big] \quad \text{drop } t \text{ and write } s' \equiv s_{t+1}\end{aligned} \tag{33}$$

Given above, one may approximate $A^\pi(s,a)$, by $s' \sim P(s'|s,a)$ and use the neural network V_{θ_V} to compute:

$$A^\pi(s,a) = r(s) + \gamma V^\pi_{\theta_V}(s') - V^\pi_{\theta_V}(s) \tag{34}$$

using only one network V_{θ_V}

3.8 Off policy

In reinforcement learning, we often want to optimize a policy π by updating it with respect to a different policy β. This is known as off-policy learning. Looking at Policy Gradient Theorem Eq.(1), where we have:

$$J(\theta) = \sum_{s\in\mathcal{S}} \rho^{\pi}(s) \sum_{a\in\mathcal{A}} Q^{\pi}(s,a)\pi_\theta(a|s) \tag{35}$$

Change behavioral distribution from π to $\pi_{\theta_{\text{old}}}$ but target policy is still $\pi_\theta(a|s)$, and therefore making:

$$\begin{aligned} J(\theta) &= \sum_{s\in\mathcal{S}} \rho^{\pi_{\theta_{\text{old}}}} \sum_{a\in\mathcal{A}} Q^{\pi}(s,a)\pi_\theta(a|s) \\ &= \mathbb{E}_{s\sim\rho^{\pi_{\theta_{\text{old}}}}}\big[\sum_{a\in\mathcal{A}} Q^{\pi}(s,a)\pi_\theta(a|s)\big] \end{aligned} \tag{36}$$

For simpler notation, we let $\pi_{\theta_{\text{old}}} \to \beta$, and compute the gradient of $J(\theta)$:

$$\begin{aligned} \nabla_\theta J(\theta) &= \nabla_\theta \mathbb{E}_{s\sim\rho^\beta}\Big[\sum_{a\in\mathcal{A}} \underbrace{Q^{\pi}(s,a)}_{u}\underbrace{\pi_\theta(a|s)}_{v}\Big] \\ &= \mathbb{E}_{s\sim\rho^\beta}\Big[\sum_{a\in\mathcal{A}} \big(Q^{\pi}(s,a)\nabla_\theta\pi_\theta(a|s) + \pi_\theta(a|s)\nabla_\theta Q^{\pi}(s,a)\big)\Big] \\ &\overset{(i)}{\approx} \mathbb{E}_{s\sim\rho^\beta}\Big[\sum_{a\in\mathcal{A}} Q^{\pi}(s,a)\nabla_\theta\pi_\theta(a|s)\Big] \qquad \text{apply Policy gradient theorem} \\ &= \mathbb{E}_{s\sim\rho^\beta}\Big[\sum_{a\in\mathcal{A}} \beta(a|s)\frac{\pi_\theta(a|s)}{\beta(a|s)} Q^{\pi}(s,a)\frac{\nabla_\theta\pi_\theta(a|s)}{\pi_\theta(a|s)}\Big] \\ &= \mathbb{E}_{s\sim\rho^\beta,a\sim\beta}\Big[\underbrace{\frac{\pi_\theta(a|s)}{\beta(a|s)}}_{\text{importance weights}} Q^{\pi}(s,a)\nabla_\theta \ln\pi_\theta(a|s)\Big] \end{aligned} \tag{37}$$

in an iterative algorithm, we use $\beta = \pi_{k\theta_k}(a|s)$, you have on-policy, so we use β genericaly

4 TRPO objective

Once again, for simplicity, we let $\beta \equiv \theta_{\text{old}}$. We aim to demonstrate why the lower bound of the performance difference between a new policy π and an old policy β consists of the advantage function minus the KL divergence.

4.1 Why we need an old policy β or π_{old}

Before going further, why do we need an old policy β or π_{old}?

- β is the model's state from the very last optimization step (or the beginning of the current rollout batch).
- After the current policy (π_θ) is updated for a few epochs, β is updated to match π_θ
- Why we need it? In reinforcement learning, generating new responses (rollouts) is computationally expensive. By keeping track of the "old" policy that generated the data, we can update our current policy (π_θ) multiple times using the same generated data. The ratio simply corrects for the fact that the data was generated by an older version of the model. The clipping mechanism ensures the new policy doesn't change too drastically from the old one in a single step.

4.2 The Performance Difference Lemma

The starting point is the Performance Difference Lemma (Kakade & Langford, 2002), which expresses the exact difference in expected returns as the expected sum of advantages:

$$J(\pi) - J(\beta) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty} \gamma^t A^\beta(s_t, a_t)\right] \tag{38}$$

4.3 Proof

Step 1: Expand $J(\pi)$ and $J(\beta)$

The value of a policy is the expected discounted return. For π, we write this directly as an expectation over trajectories generated by π. For β, it is useful to write the same policy value as the expected β-value of the initial state:

$$\begin{aligned} J(\pi) &= \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right] \\ J(\beta) &= \mathbb{E}_{s_0\sim\rho_0}\left[V^\beta(s_0)\right]. \end{aligned} \tag{39}$$

This does not conflict with the usual expression $J(\beta) = \mathbb{E}_{\tau\sim\beta}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$. The reason is that $V^\beta(s_0)$ already contains the future expectation under policy β: $V^\beta(s_0) =$

$\mathbb{E}_{\tau\sim\beta}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)\mid s_0\right]$. After this future return has been compressed into $V^\beta(s_0)$, the only remaining randomness is the initial state $s_0\sim\rho_0$.

Therefore, we may take the expectation of $V^\beta(s_0)$ under either trajectory distribution, as long as the initial-state distribution is the same:

$$\begin{aligned}\mathbb{E}_{\tau\sim\pi}\left[V^\beta(s_0)\right] &= \mathbb{E}_{s_0\sim\rho_0}\left[V^\beta(s_0)\right]\\ &= \mathbb{E}_{\tau\sim\beta}\left[V^\beta(s_0)\right]\\ &= J(\beta).\end{aligned} \tag{40}$$

We switch to $\tau\sim\pi$ because the next step compares the new policy π against the reference value function V^β along trajectories generated by π:

Step 2: Telescoping Sum of V^β

We can express $V^\beta(s_0)$ as a telescoping sum over time t:

$$V^\beta(s_0)=\sum_{t=0}^{\infty}\left(\gamma^t V^\beta(s_t)-\gamma^{t+1}V^\beta(s_{t+1})\right) \tag{41}$$

we know this is true, because:

$$\begin{aligned}V^\beta(s_0) &= \gamma^0V^\beta(s_0)-\gamma^1V^\beta(s_1)+\gamma^1V^\beta(s_1)-\gamma^2V^\beta(s_2)+\gamma^2V^\beta(s_2)-\gamma^3V^\beta(s_3)+\cdots\\ &= V^\beta(s_0)-\text{a very small term}\end{aligned} \tag{42}$$

This holds because the intermediate terms cancel out.

Step 3: Substitute and Regroup

$$\begin{aligned}J(\pi)-J(\beta) &= \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)-\sum_{t=0}^{\infty}\left(\gamma^tV^\beta(s_t)-\gamma^{t+1}V^\beta(s_{t+1})\right)\right]\\ &= \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t\left(r(s_t,a_t)+\gamma V^\beta(s_{t+1})-V^\beta(s_t)\right)\right]\end{aligned} \tag{43}$$

Step 4: Introduce Q-Function and Advantage

Recall the relationship between Q, V, and the next state:

$$\mathbb{E}_{s_{t+1}}[r(s_t,a_t)+\gamma V^\beta(s_{t+1})]=Q^\beta(s_t,a_t) \tag{44}$$

Substituting this expectation into the sum:

$$J(\pi) - J(\beta) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t\left(Q^\beta(s_t,a_t) - V^\beta(s_t)\right)\right]$$

Finally, using the definition $A^\beta(s,a) = Q^\beta(s,a) - V^\beta(s)$, we arrive at the result:

$$J(\pi) - J(\beta) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t A^\beta(s_t,a_t)\right] \quad \text{on-policy} \tag{45}$$

This can be rewritten using the discounted state visitation distribution $\rho_\pi(s)$ of the new policy:

$$J(\pi) - J(\beta) = \sum_s \rho^\pi(s)\sum_a \pi(a|s)A^\beta(s,a) \tag{46}$$

this is because:

$$\begin{aligned}
\mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t A^\beta(s_t,a_t)\right] &= \sum_{t=0}^{\infty}\gamma^t\mathbb{E}_{\tau\sim\pi}\left[A^\beta(s_t,a_t)\right] \\
&= \sum_{t=0}^{\infty}\gamma^t\sum_s P(s_t=s|\pi)\sum_a \pi(a|s)A^\beta(s,a) \\
&= \sum_s\left(\sum_{t=0}^{\infty}\gamma^t P(s_t=s|\pi)\right)\sum_a \pi(a|s)A^\beta(s,a) \\
&= \sum_s \rho^\pi(s)\sum_a \pi(a|s)A^\beta(s,a)
\end{aligned} \tag{47}$$

moving $J(\beta)$ to the right side, we get:

$$J(\pi) = J(\beta) + \mathbb{E}_{s\sim\rho^\pi}\left[\sum_a \pi(a|s)A^\beta(s,a)\right] \tag{48}$$

4.4 The Surrogate Objective

A major difficulty in optimization is that $\rho^\pi(s)$ is unknown because it depends on the policy π we are trying to find.

To solve this, we define a local approximation called the surrogate objective, denoted $L_\beta(\pi)$. We replace $\rho^\pi(s)$ with the old policy's distribution:

$$L_\beta(\pi) = J(\beta) + \sum_s \rho^\beta(s)\sum_a \pi(a|s)A^\beta(s,a) \tag{49}$$

$$J(\pi) - J(\beta) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t A^\beta(s_t, a_t)\right]$$

to approximate the true objective:

$$J(\pi) = J(\beta) + \sum_s \rho^\pi(s)\sum_a \pi(a|s)A^\beta(s,a) \tag{50}$$

4.4.1 Bounding the

$$L_\beta(\pi) = J(\beta) + \sum_s \rho^\beta(s)\sum_a \pi(a|s)A^\beta(s,a)$$

Any approximation will introduce an error. The error introduced by replacing ρ^π, (in $J(\pi)$) with ρ^β, (in $L_\beta(\pi)$) is bounded by the divergence between the two policies. But exactly how much is the error? Schulman et al. (TRPO paper) proved that:

$$\begin{aligned} &|J(\pi) - L_\beta(\pi)| \le C \cdot \max_s D_{KL}(\beta(\cdot|s)\,||\,\pi(\cdot|s)) \\ \implies &\underbrace{-(J(\pi) - L_\beta(\pi)) \le C \cdot \max_s D_{KL}(\beta(\cdot|s)\,||\,\pi(\cdot|s))}_{\text{taking this part}} \le J(\pi) - L_\beta(\pi) \end{aligned} \tag{51}$$

where $C = \frac{4\epsilon\gamma}{(1-\gamma)^2}$ is a constant related to the discount factor γ and the maximum advantage scalar ϵ. Also, for shorthand noation, we write:

$$D_{KL}^{\max}(\beta\,||\,\pi) \equiv \max_s D_{KL}(\beta(\cdot|s)\,||\,\pi(\cdot|s)) \tag{52}$$

therefore, we have:

$$\begin{aligned} -(J(\pi) - L_\beta(\pi)) &\le C \cdot D_{KL}^{\max}(\beta\,||\,\pi) \\ \implies J(\pi) &\ge L_\beta(\pi) - C \cdot D_{KL}^{\max}(\beta, \pi) \end{aligned} \tag{53}$$

Substituting the definition of $L_\beta(\pi)$ back into the equation, we get the final lower bound for the improvement:

$$\begin{aligned} J(\pi) - J(\beta) &\ge \underbrace{\left(\sum_s \rho^\beta(s)\sum_a \pi(a|s)A^\beta(s,a)\right)}_{\text{Surrogate Advantage } \mathcal{L}_\theta(\pi)} - \underbrace{C \cdot D_{KL}^{\max}(\beta\,||\,\pi)}_{\text{KL Penalty}} \\ &= \sum_s \rho^\beta \sum_a \beta(a|s)\frac{\pi(a|s)}{\beta(a|s)}A^\beta(s,a) - C \cdot D_{KL}^{\max}(\beta\,||\,\pi) \quad \text{change to off-policy} \\ &= \mathbb{E}_{\tau\sim\beta}\left[\sum_{t=0}^{\infty}\frac{\pi(a_t|s_t)}{\beta(a_t|s_t)}A^\beta(s_t,a_t)\right] - C \cdot D_{KL}^{\max}(\beta\,||\,\pi) \end{aligned} \tag{54}$$

or adjusted for the discount factor:

$$J(\pi) - J(\beta) \geq \mathbb{E}_{\tau\sim\beta}\left[\sum_{t=0}^{\infty}\gamma^t \frac{\pi(a_t|s_t)}{\beta(a_t|s_t)} A^{\beta}(s_t, a_t)\right] - C \cdot D_{KL}^{\max}(\beta \,||\, \pi) \tag{55}$$

In words, to guarantee a monotonic improvement in policy performance ($J(\pi) > J(\beta)$), one must maximize the expected advantage under the old state distribution while penalizing the KL divergence between the new and old policies.

4.5 Why equality occurs ($\pi = \beta$)?

$$J(\beta) - J(\beta) = \underbrace{\mathbb{E}_{\tau\sim\beta}\left[\sum_{t=0}^{\infty}\gamma^t \frac{\beta(a_t|s_t)}{\beta(a_t|s_t)} A^{\beta}(s_t, a_t)\right]}_{=\text{what?}} - \underbrace{C\,\mathbb{E}_{s\sim d_k^{\beta}}[\mathrm{KL}(\beta\|\beta)[s]]}_{=0,\text{well, that's KL}}$$

- looking at $\mathbb{E}_{\tau\sim\beta}\left[\sum_{t=0}^{\infty}\gamma^t A^{\beta}(s_t, a_t)\right]$:

$$\begin{aligned}
&\mathbb{E}_{\tau\sim\beta}\left[\sum_{t=0}^{\infty}\gamma^t A^{\beta}(s_t, a_t)\right] = \sum_{t=0}^{\infty}\gamma^t \mathbb{E}_{s_t\sim d_t^{\beta}}\left[\sum_{a_t\in\mathcal{A}} \beta(a_t|s_t) A^{\beta}(s_t, a_t)\right] \\
&= \sum_{t=0}^{\infty}\gamma^t \mathbb{E}_{s_t\sim d_t^{\beta}}\left[\sum_{a_t\in\mathcal{A}} \beta(a_t|s_t)\big(Q^{\beta}(s_t, a_t) - V^{\beta}(s_t)\big)\right] \\
&= \sum_{t=0}^{\infty}\gamma^t \mathbb{E}_{s_t\sim d_t^{\beta}}\left[\underbrace{\sum_{a_t\in\mathcal{A}} \beta(a_t|s_t) Q^{\beta}(s_t, a_t)}_{V^{\beta}(s_t)} - V^{\beta}(s_t) \underbrace{\sum_{a_t\in\mathcal{A}} \beta(a_t|s_t)}_{1}\right] \\
&= \sum_{t=0}^{\infty}\gamma^t \mathbb{E}_{s_t\sim d_t^{\beta}}\left[V^{\beta}(s_t) - V^{\beta}(s_t)\right] = 0
\end{aligned}$$

  The crucial point is the red term $\beta(a_t|s_t)$: under $\tau \sim \beta$, actions are sampled according to β, so the action expectation is a weighted sum, not an unweighted sum over actions.

- As a side note: if instead we look at $\mathbb{E}_{\tau\sim\beta}\left[\sum_{t=0}^{\infty}\gamma^t f(a_t) A^{\beta}(s_t, a_t)\right]$:

$$\mathbb{E}_{\tau\sim\beta}\left[\sum_{t=0}^{\infty} f(a_t)\gamma^t A^\beta(s_t,a_t)\right] = \sum_{t=0}^{\infty}\gamma^t \mathbb{E}_{s_t\sim d_t^\beta}\left[\sum_{a_t\in\mathcal{A}}\beta(a_t|s_t)f(a_t)\big(Q^\beta(s_t,a_t)-V^\beta(s_t)\big)\right]$$

$$= \sum_{t=0}^{\infty}\gamma^t \mathbb{E}_{s_t\sim d_t^\beta}\left[\underbrace{\sum_{a_t\in\mathcal{A}}\beta(a_t|s_t)f(a_t)Q^\beta(s_t,a_t)}_{\neq V^\beta(s_t)\sum_{a_t}\beta(a_t|s_t)f(a_t)} - V^\beta(s_t)\sum_{a_t}\beta(a_t|s_t)f(a_t)\right]$$

5 Details of TRPO objective

5.1 Change max KL to average KL

$D_{KL}^{\max}(\beta \,||\, \pi)$ can be difficult to compute, so we approximate it with the average KL divergence:

$$\widehat{\mathrm{KL}}(\beta \,||\, \pi) = \mathbb{E}_{s\sim\rho^{\beta}}[\mathrm{KL}(\beta(\cdot|s))\|\pi(\cdot|s)] \tag{56}$$

Two different constraints for $\mathrm{KL}(\pi\|\beta)$

- directly penalize the KL divergence:

$$\max_{\pi}\Big[\underbrace{\mathbb{E}_{\tau\sim\beta}\big[\sum_{t=0}^{\infty}\gamma^t\frac{\pi(a_t|s_t)}{\beta(a_t|s_t)}A^{\beta}(s_t,a_t)\big]}_{\mathcal{L}_{\theta_{\text{old}}}(\theta)} - C\,\mathbb{E}_{s\sim\rho^{\beta}}[\widehat{\mathrm{KL}}(\beta(\cdot|s))\|\pi(\cdot|s)]\Big] \tag{57}$$

  where θ_{old} is the parameter of the old policy β, and θ is the parameter of the new policy π.

- constrain the KL divergence (TRPO)

$$\begin{aligned}&\max_{\pi}\Big[\underbrace{\mathbb{E}_{\tau\sim\beta}\Big[\sum_{t=0}^{\infty}\gamma^t\frac{\pi(a_t|s_t)}{\beta(a_t|s_t)}A^{\beta}(s_t,a_t)\Big]}_{\mathcal{L}_{\theta_{\text{old}}}(\theta)}\Big]\\ &\text{s.t.}\quad \mathbb{E}_{s\sim\rho^{\beta}}[\mathrm{KL}(\beta(\cdot|s))\|\pi(\cdot|s)] \le \delta\end{aligned} \tag{58}$$

  We see that the TRPO objective function is formulated as a constrained optimization problem. We maximize a surrogate objective subject to a constraint on the size of the policy update (the trust region). Some literature may express it as:

$$\begin{aligned}&\underset{\theta}{\text{maximize}}\quad \mathbb{E}_{s\sim\rho_{\theta_{\text{old}}},a\sim\pi_{\theta_{\text{old}}}}\left[\frac{\pi_\theta(a|s)}{\pi_{\theta_{\text{old}}}(a|s)}A^{\pi_{\theta_{\text{old}}}}(s,a)\right]\\ &\text{subject to}\quad \underbrace{\mathbb{E}_{s\sim\rho_{\theta_{\text{old}}}}\left[\mathrm{KL}(\pi_{\theta_{\text{old}}}(\cdot|s)\|\pi_\theta(\cdot|s))\right]}_{\widehat{\mathrm{KL}}(\theta_{\text{old}},\theta)} \le \delta\end{aligned} \tag{59}$$

  which is exactly the same as the TRPO objective function stated in (58).

– Surrogate Advantage Objective:

$$\mathcal{L}_\theta(\pi) = \mathbb{E}\left[\frac{\pi_\theta(a|s)}{\pi_{\theta_{\text{old}}}(a|s)} A^{\pi_{\theta_{\text{old}}}}(s,a)\right] \tag{60}$$

The term $\frac{\pi_\theta(a|s)}{\pi_{\theta_{\text{old}}}(a|s)}$ is the importance sampling ratio, allowing us to estimate the performance of the new policy π_θ using trajectories collected by the old policy $\pi_{\theta_{\text{old}}}$. $A^{\pi_{\theta_{\text{old}}}}(s,a)$ is the advantage function.

– Trust Region Constraint:

$$\widehat{\text{KL}}(\theta_{\text{old}}, \theta) \leq \delta$$

This constraint ensures the new policy does not deviate too far from the old one, preventing performance collapse. δ is a hyperparameter (e.g., 0.01) defining the maximum allowed KL divergence.

- $\text{KL}(\beta(\cdot|s)\|\pi(\cdot|s))$ part need concept of natural gradient

6 Natural Gradient manifold: $\text{KL}(\beta(\cdot|s)\|\pi(\cdot|s)) = \delta$

Taylor Expansion and Local Approximation

The first order Taylor expansion is a valid approximation only within a small neighborhood. Therefore, we cannot minimize it globally. We must look for a minimizer subject to a constraint on the size of $\mathbf{h}$ (a 'Trust Region').

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}_0 + \mathbf{h}) &\approx \mathcal{L}(\boldsymbol{\theta}_0) + \nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}_0)^\top \mathbf{h} \\ \implies \arg\min_{\mathbf{h}:\|\mathbf{h}\|\leq\epsilon} \mathcal{L}(\boldsymbol{\theta}_0 + \mathbf{h}) &\approx \arg\min_{\mathbf{h}:\|\mathbf{h}\|\leq\epsilon} \nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}_0)^\top \mathbf{h} \end{aligned} \tag{61}$$

Steepest Gradient Descent (Euclidean Norm)

We minimize the linear approximation over an equiv-euclidean-distance hyper-sphere (constraint $|\mathbf{h}|_2 = 1$).

$$\begin{aligned} \mathbf{h}^* &= \arg\min_{\mathbf{h}:\|\mathbf{h}\|=1} \mathcal{L}(\boldsymbol{\theta}_0 + \mathbf{h}) \\ &\approx \arg\min_{\mathbf{h}:\|\mathbf{h}\|=1} \nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}_0)^\top \mathbf{h} \\ &= -\frac{\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}_0)}{\|\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}_0)\|} \end{aligned} \tag{62}$$

Since policy is a distribution, we can minimize at an equiv-KL-distance manifold:

$$\begin{aligned}\mathbf{h}^* &= \arg\min_{\mathbf{h}} \left\{ \mathcal{L}(\boldsymbol{\theta}+\mathbf{h}) : \mathbf{h} \in \left(\mathrm{KL}[p_\theta \| p_{\theta+\mathbf{h}}] = \delta\right) \right\} \\ &\approx \arg\min_{\mathbf{h}} \left\{ \nabla_\theta \mathcal{L}(\boldsymbol{\theta})^\top \mathbf{h} : \mathbf{h} \in \left(\mathrm{KL}[p_{\boldsymbol{\theta}} \| p_{\boldsymbol{\theta}+\mathbf{h}}] = \delta\right) \right\}\end{aligned} \tag{63}$$

We can solve the above problem using Lagrange Multiplier:

$$\mathbf{h}^* = \arg\min_{\mathbf{h}} \left(\nabla_\theta \mathcal{L}(\theta)^\top \mathbf{h} + \lambda(\mathrm{KL}[p_\theta \| p_{\theta+\mathbf{h}}] - \delta) \right) \tag{64}$$

6.1 The solution

"If" we can prove second degree Taylor approximation of a KL divergence is:

$$\mathrm{KL}[p_{\boldsymbol{\theta}} \| p_{\boldsymbol{\theta}+\mathbf{h}}] \equiv \mathrm{KL}[p(x|\boldsymbol{\theta}) \| p(x|\boldsymbol{\theta}+\mathbf{h})] \approx \frac{1}{2}\mathbf{h}^\top \mathrm{F} \mathbf{h} \tag{65}$$

alternatively, we can also express it by letting $\mathbf{h} = \theta - \theta_0$:

$$\mathrm{KL}[p_{\boldsymbol{\theta}_0} \| p_{\boldsymbol{\theta}}] \equiv \mathrm{KL}[p(x|\boldsymbol{\theta}_0) \| p(x|\boldsymbol{\theta})] \approx \frac{1}{2}(\boldsymbol{\theta}-\boldsymbol{\theta}_0)^\top \mathrm{F}(\boldsymbol{\theta}_0)(\boldsymbol{\theta}-\boldsymbol{\theta}_0) \tag{66}$$

where F is the Fisher Information Matrix. (we will explain what is Fisher Information Matrix later)

Then, Eq.(64) becomes:

$$\begin{aligned}&\mathbf{h}^* \approx \arg\min_{\mathbf{h}} \left(\nabla_\theta \mathcal{L}(\boldsymbol{\theta})^\top \mathbf{h} + \lambda(\frac{1}{2}\mathbf{h}^\top \mathrm{F}\mathbf{h} - c) \right) \\ \Longrightarrow\ &\frac{\partial}{\partial \mathbf{h}} \left(\nabla_\theta \mathcal{L}(\boldsymbol{\theta})^\top \mathbf{h} + \frac{1}{2}\lambda\, \mathbf{h}^\top \mathrm{F}\mathbf{h} - \lambda c \right) = 0 \\ &\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) + \lambda\, \mathrm{F}\mathbf{h} = 0\end{aligned} \tag{67}$$

Finally, the solution is:

$$\mathbf{h}^* = -\frac{1}{\lambda}\mathrm{F}^{-1} \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) \tag{68}$$

6.2 Why second order Taylor Expansion $\mathrm{KL}[p_\theta \| p_{\theta+\mathbf{h}}] \approx \frac{1}{2}\mathbf{h}^\top \mathrm{F}\mathbf{h}$

Let's review the second order Taylor expansion:

$$f(\mathbf{x}_0 + \mathbf{h}) \approx f(\mathbf{x}) + \nabla_{\mathbf{x}} f(\mathbf{x})^\top \mathbf{h} + \frac{1}{2}\mathbf{h}^\top \nabla^2_{\mathbf{x}} f(\mathbf{x}) \mathbf{h} \quad | \ \mathbf{x} = \mathbf{x}_0 \tag{69}$$

We write it as $\mathrm{KL}[p_{\boldsymbol{\theta}_0} \,\|\, p_{\boldsymbol{\theta}+\mathbf{h}}] \quad | \, \boldsymbol{\theta} = \boldsymbol{\theta}_0$, to indicate that $p_{\boldsymbol{\theta}_0}$ is a fixed distribution.

$$
\begin{aligned}
\mathrm{KL}[p_{\boldsymbol{\theta}_0} \,\|\, p_{\boldsymbol{\theta}+\mathbf{h}}]|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0} &\approx \mathrm{KL}[p_{\boldsymbol{\theta}_0} \,\|\, p_{\boldsymbol{\theta}+\mathbf{h}}] + \left(\nabla_{\boldsymbol{\theta}} \mathrm{KL}[p_{\boldsymbol{\theta}} \,\|\, p_{\boldsymbol{\theta}}])^\top \mathbf{h} + \frac{1}{2} \mathbf{h}^\top \left(\nabla^2_{\boldsymbol{\theta}} \mathrm{KL}[p_{\boldsymbol{\theta}} \,\|\, p_{\boldsymbol{\theta}}] \right) \mathbf{h} \quad \big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0} \right. \\
&= \mathrm{KL}[p_{\boldsymbol{\theta}_0} \,\|\, p_{\boldsymbol{\theta}_0}] + \left(\underbrace{\left(\nabla_{\boldsymbol{\theta}} \mathrm{KL}[p_{\boldsymbol{\theta}_0} \,\|\, p_{\boldsymbol{\theta}}] \right)^\top}_{①} \mathbf{h} + \frac{1}{2} \mathbf{h}^\top \underbrace{\left(\nabla^2_{\boldsymbol{\theta}} \mathrm{KL}[p_{\boldsymbol{\theta}_0} \,\|\, p_{\boldsymbol{\theta}}] \right)}_{②} \mathbf{h} \quad \big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0} \right) \\
&= 0 + 0 + \frac{1}{2} \mathbf{h}^\top \mathbf{F} \mathbf{h} \\
&= \frac{1}{2} \mathbf{h}^\top \mathbf{F} \mathbf{h}
\end{aligned}
\tag{70}
$$

for the first derivative of KL divergence w.r.t. $\boldsymbol{\theta}_0$:

$$
\begin{aligned}
\nabla_{\boldsymbol{\theta}} \mathrm{KL}[p(\mathbf{x}|\boldsymbol{\theta}_0) \,\|\, p(\mathbf{x}|\boldsymbol{\theta})] &= \nabla_{\boldsymbol{\theta}} \int p(\mathbf{x}|\boldsymbol{\theta}_0) \log p(\mathbf{x}|\boldsymbol{\theta}_0) \, \mathrm{d}\mathbf{x} - \nabla_{\boldsymbol{\theta}} \int p(\mathbf{x}|\boldsymbol{\theta}_0) \log p(\mathbf{x}|\boldsymbol{\theta}) \, \mathrm{d}\mathbf{x} \\
&= - \int p(\mathbf{x}|\boldsymbol{\theta}_0) \nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}|\boldsymbol{\theta}) \, \mathrm{d}\mathbf{x} \\
&= - \int p(\mathbf{x}|\boldsymbol{\theta}_0) \frac{\nabla_{\boldsymbol{\theta}} p(\mathbf{x}|\boldsymbol{\theta})}{p(\mathbf{x}|\boldsymbol{\theta})} \, \mathrm{d}\mathbf{x}
\end{aligned}
\tag{71}
$$

$$
\begin{aligned}
\implies \nabla_{\boldsymbol{\theta}} \mathrm{KL}[p(\mathbf{x}|\boldsymbol{\theta}_0) \,\|\, p(\mathbf{x}|\boldsymbol{\theta})] \quad \big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0} &= - \int p(\mathbf{x}|\boldsymbol{\theta}_0) \frac{\nabla_{\boldsymbol{\theta}} p(\mathbf{x}|\boldsymbol{\theta}_0)}{p(\mathbf{x}|\boldsymbol{\theta}_0)} \, \mathrm{d}\mathbf{x} \\
&= -\nabla_{\boldsymbol{\theta}} \int p(\mathbf{x}|\boldsymbol{\theta}_0) \mathrm{d}\mathbf{x} \\
&= -\nabla_{\boldsymbol{\theta}} [1] \\
&= 0
\end{aligned}
\tag{72}
$$

for the second derivative of KL divergence w.r.t. $\boldsymbol{\theta}_0$:

$$
\nabla_{\boldsymbol{\theta}} \mathrm{KL}[p(x|\boldsymbol{\theta}_0) \,\|\, p(x|\boldsymbol{\theta})] = - \int p(x|\boldsymbol{\theta}_0) \nabla_{\boldsymbol{\theta}} \log p(x|\boldsymbol{\theta}) \, \mathrm{d}x \quad \text{From Eq.(71)} \tag{73}
$$

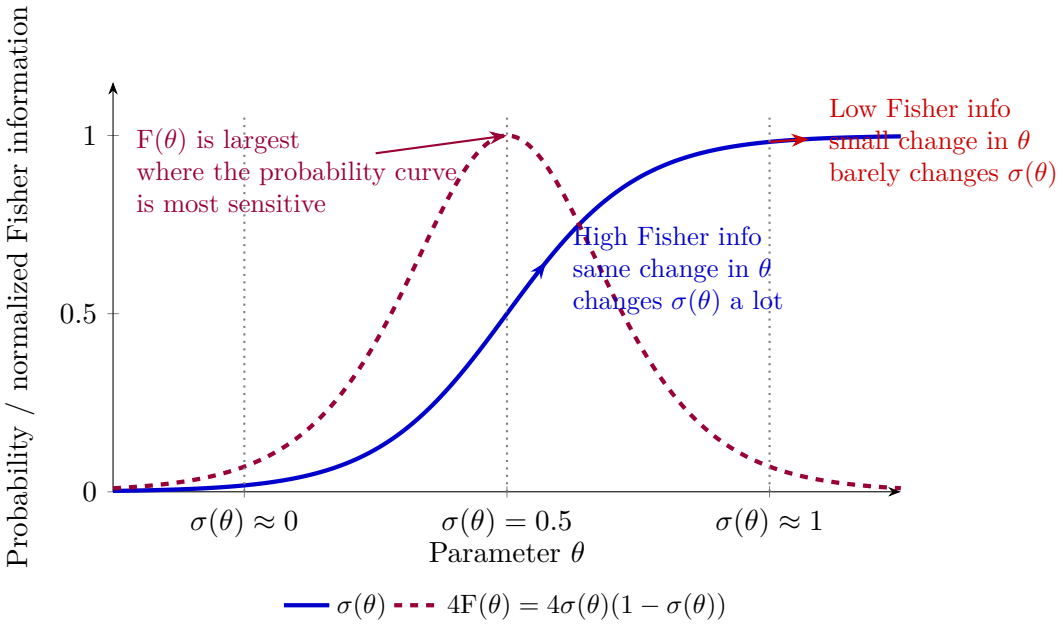
This implies that:

$$
\begin{aligned}
&\nabla^2_{\boldsymbol{\theta}} \mathrm{KL}[p(\mathbf{x}|\boldsymbol{\theta}_0) \,\|\, p(\mathbf{x}|\boldsymbol{\theta})] \\
&= \nabla_{\boldsymbol{\theta}} \Big[- \int p(\mathbf{x}|\boldsymbol{\theta}_0) \nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}|\boldsymbol{\theta}) \, \mathrm{d}\mathbf{x} \Big] \\
&= - \int p(\mathbf{x}|\boldsymbol{\theta}_0) \, \nabla_{\boldsymbol{\theta}} \big[\nabla_{\boldsymbol{\theta}} \big[\log p(\mathbf{x}|\boldsymbol{\theta}) \big] \big] \, \mathrm{d}\mathbf{x} \\
&= - \int p(\mathbf{x}|\boldsymbol{\theta}_0) \, \nabla_{\boldsymbol{\theta}} \left[\frac{\nabla_{\boldsymbol{\theta}} [p(\mathbf{x}|\boldsymbol{\theta})]}{p(\mathbf{x}|\boldsymbol{\theta})} \right] \mathrm{d}\mathbf{x} \\
&= - \int p(\mathbf{x}|\boldsymbol{\theta}_0) \, \nabla_{\boldsymbol{\theta}} \Big[\underbrace{\nabla_{\boldsymbol{\theta}} [p(\mathbf{x}|\boldsymbol{\theta})]}_{u} \underbrace{p(\mathbf{x}|\boldsymbol{\theta})^{-1}}_{v} \Big] \, \mathrm{d}\mathbf{x} \\
&= - \int p(\mathbf{x}|\boldsymbol{\theta}_0) \, \Big[\underbrace{\nabla_{\boldsymbol{\theta}} [p(\mathbf{x}|\boldsymbol{\theta})] (-p(\mathbf{x}|\boldsymbol{\theta})^{-2} \nabla_{\boldsymbol{\theta}} [p(\mathbf{x}|\boldsymbol{\theta})]^\top)}_{uv'} + \underbrace{\nabla^2_{\boldsymbol{\theta}} [p(\mathbf{x}|\boldsymbol{\theta})] p(\mathbf{x}|\boldsymbol{\theta})^{-1}}_{u'v} \Big] \, \mathrm{d}\mathbf{x} \quad \text{scalar form} \\
&= - \int p(\mathbf{x}|\boldsymbol{\theta}_0) \left[\frac{\nabla^2_{\boldsymbol{\theta}} [p(\mathbf{x}|\boldsymbol{\theta})]}{p(\mathbf{x}|\boldsymbol{\theta})} \right] \mathrm{d}\mathbf{x} + \int p(\mathbf{x}|\boldsymbol{\theta}_0) \left[\left(\frac{\nabla p(\mathbf{x}|\boldsymbol{\theta})}{p(\mathbf{x}|\boldsymbol{\theta})} \right) \left(\frac{\nabla p(\mathbf{x}|\boldsymbol{\theta})}{p(\mathbf{x}|\boldsymbol{\theta})} \right)^\top \right] \mathrm{d}\mathbf{x} \quad \text{vector-matrix form}
\end{aligned}
\tag{74}
$$

taking the limit $\boldsymbol{\theta} \to \boldsymbol{\theta}_0$:

$$
\begin{aligned}
&\nabla^2_{\boldsymbol{\theta} \to \boldsymbol{\theta}_0} \mathrm{KL}[p(\mathbf{x}|\boldsymbol{\theta}_0) \,\|\, p(\mathbf{x}|\boldsymbol{\theta})] \\
&= - \int p(\mathbf{x}|\boldsymbol{\theta}_0) \left[\frac{\nabla^2_{\boldsymbol{\theta}} [p(\mathbf{x}|\boldsymbol{\theta}_0)]}{p(\mathbf{x}|\boldsymbol{\theta}_0)} \right] \mathrm{d}\mathbf{x} + \int p(\mathbf{x}|\boldsymbol{\theta}_0) \left[\left(\frac{\nabla p(\mathbf{x}|\boldsymbol{\theta}_0)}{p(\mathbf{x}|\boldsymbol{\theta}_0)} \right) \left(\frac{\nabla p(\mathbf{x}|\boldsymbol{\theta}_0)}{p(\mathbf{x}|\boldsymbol{\theta}_0)} \right)^\top \right] \mathrm{d}\mathbf{x} \\
&= - \int \nabla^2_{\boldsymbol{\theta}} [p(\mathbf{x}|\boldsymbol{\theta}_0)] \mathrm{d}\mathbf{x} + \mathbb{E}_{p(\mathbf{x}|\boldsymbol{\theta}_0)} \left[\left(\frac{\nabla p(\mathbf{x}|\boldsymbol{\theta}_0)}{p(\mathbf{x}|\boldsymbol{\theta}_0)} \right) \left(\frac{\nabla p(\mathbf{x}|\boldsymbol{\theta}_0)}{p(\mathbf{x}|\boldsymbol{\theta}_0)} \right)^\top \right] \\
&= -\nabla^2_{\boldsymbol{\theta}} \int p(\mathbf{x}|\boldsymbol{\theta}_0) \mathrm{d}\mathbf{x} + \mathbb{E}_{p(\mathbf{x}|\boldsymbol{\theta}_0)} \left[\nabla \log p(\mathbf{x}|\boldsymbol{\theta}_0) \, \nabla \log p(\mathbf{x}|\boldsymbol{\theta}_0)^\top \right] \\
&= 0 + \mathrm{F}(\boldsymbol{\theta}_0) \\
&= \mathrm{F}(\boldsymbol{\theta}_0)
\end{aligned}
\tag{75}
$$

6.3 Something about Fisher Information



6.3.1 Intuition

The most intuitive way to read Fisher information is: how much does the probability distribution change when we move θ a little? For a Bernoulli distribution with logit parameter θ, we have

$$\sigma(\theta) = \frac{1}{1+e^{-\theta}}, \qquad \mathrm{F}(\theta) = \sigma(\theta)(1-\sigma(\theta)).$$

When θ is near 0, the sigmoid curve is steep. A small change in θ causes a large change in $\sigma(\theta)$, so Fisher information is high. When θ is very positive or very negative, the sigmoid saturates near 1 or 0. A small change in θ barely changes the distribution, so Fisher information is low.

We evaluate θ_{MLE} at the peak for the Maximum Likelihood Estimate for a practical reason: That is our best guess. At θ_{MLE}: The curvature tells us how "stable" our best guess is.

6.3.2 Relationship with expected Hessian of the log-likelihood

Mathematically, let's have a look at the second derivative of $\log p(\mathbf{x}|\boldsymbol{\theta})$, of a particular parameter θ_i and θ_j in the Hessian matrix:

$$
\begin{aligned}
\nabla^2_{\theta_i,\theta_j}[\log p_{\boldsymbol{\theta}}(\mathbf{x})] &= \nabla_{\theta_i}(\nabla_{\theta_j}\log(p_{\boldsymbol{\theta}}(\mathbf{x}))) \\
&= \nabla_{\theta_i}\left(\frac{\nabla_{\theta_j}p_{\boldsymbol{\theta}}(\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{x})}\right) \\
&= \nabla_{\theta_i}\Bigg(\underbrace{\nabla_{\theta_j}p_{\boldsymbol{\theta}}(\mathbf{x})}_{u}\underbrace{p_{\boldsymbol{\theta}}(\mathbf{x})^{-1}}_{v}\Bigg) \\
&= \underbrace{\frac{\nabla^2_{\theta_i,\theta_j}p_{\boldsymbol{\theta}}(\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{x})}}_{u'v}\underbrace{-\frac{\nabla_{\theta_i}p_{\boldsymbol{\theta}}(\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{x})}\frac{\nabla_{\theta_j}p_{\boldsymbol{\theta}}(\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{x})}}_{uv'}
\end{aligned}
\tag{76}
$$

This implies that:

$$
\begin{aligned}
\mathbb{E}_{p(\mathbf{x}|\boldsymbol{\theta})}\left[\nabla^2_{\theta_i,\theta_j}[\log p_\theta(x)]\right] &= \mathbb{E}_{p(\mathbf{x}|\boldsymbol{\theta})}\left[\frac{\nabla^2_{\theta_i,\theta_j}p_{\boldsymbol{\theta}}(\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{x})}\right] - \mathbb{E}_{p(\mathbf{x}|\boldsymbol{\theta})}\left[\frac{\nabla_{\theta_i}p_{\boldsymbol{\theta}}(\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{x})}\frac{\nabla_{\theta_j}p_{\boldsymbol{\theta}}(\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{x})}\right] \\
&= 0 - \mathbb{E}_{p(\mathbf{x}|\boldsymbol{\theta})}\left[\nabla_{\theta_i}[\log(p_{\boldsymbol{\theta}}(\mathbf{x}))]\nabla_{\theta_j}[\log(p_{\boldsymbol{\theta}}(\mathbf{x}))]\right] \\
&= 0 - \mathrm{F}_{i,j}
\end{aligned}
\tag{77}
$$

by construction, we have:

$$
\mathrm{F}(\boldsymbol{\theta}) = \mathbb{E}_{p(\mathbf{x}|\boldsymbol{\theta})}\left[\nabla_{\boldsymbol{\theta}}[\log(p_{\boldsymbol{\theta}}(\mathbf{x}))]\nabla_{\boldsymbol{\theta}}[\log(p_{\boldsymbol{\theta}}(\mathbf{x}))]^\top\right] \tag{78}
$$

and as a consequence, we have:

$$
\mathrm{F}(\boldsymbol{\theta}) = -\mathbb{E}_{p(\mathbf{x}|\boldsymbol{\theta})}\left[\nabla^2_{\boldsymbol{\theta}}[\log p_{\boldsymbol{\theta}}(\mathbf{x})]\right] \tag{79}
$$

Of course, we pick the right situation to substitute $\mathrm{F}(\boldsymbol{\theta})$.
BTW, now we just proved that, by Eq.(78),

$$
\mathrm{F}(\boldsymbol{\theta}_0) = \left(\nabla^2_{\boldsymbol{\theta}}\mathrm{KL}[p_{\boldsymbol{\theta}_0}\,\|\,p_{\boldsymbol{\theta}}]\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0}\right) \tag{80}
$$

By the definition of the Fisher Information matrix, we know that it must be positive semi-definite. Therefore, the expected Hessian of the log-likelihood must be negative semi-definite.

6.3.3 How does it relate to TRPO objective?

now, we need to replace any arbitrary $p(\cdot|\boldsymbol{\theta})$ with the policy $\pi_{\boldsymbol{\theta}}(a|s)$:

$$\mathrm{F}(\boldsymbol{\theta}) = \mathbb{E}_{a\sim\pi(a|\pi)}\Big[\nabla \log \pi_{\boldsymbol{\theta}}(a|s)\, \nabla \log \pi_{\boldsymbol{\theta}}(a|s)^{\top}\Big] \tag{81}$$

However, in order to cater for $\widehat{\mathrm{KL}}(\beta \,||\, \pi) = \mathbb{E}_{s\sim\rho^{\pi}}[\mathrm{KL}(\beta(\cdot|s))\|\pi(\cdot|s)]$, we need to compute:

$$\bar{\mathbf{F}}(\boldsymbol{\theta}) = \mathbb{E}_{s\sim\rho^{\pi}}[\mathrm{F}(\boldsymbol{\theta})] \tag{82}$$

so you can state them jointly as:

$$\bar{\mathbf{F}}(\boldsymbol{\theta}) = \mathbb{E}_{s\sim\rho^{\pi}, a\sim\pi}\Big[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s)\, \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s)^{\top}\Big] \tag{83}$$

6.4 The algorithm

repeat the steps until convergence:

1. feed-forward
2. compute $\nabla_{\theta}\mathcal{L}_{\theta}(\theta_n)$
3. Compute the fisher information matrix:

$$\hat{\mathbf{F}}(\theta_n) = \mathbb{E}_{s\sim\rho^{\pi}_{\theta_{\text{old}}}, a\sim\pi_{\theta_{\text{old}}}}\Big[\nabla_{\theta} \log \pi_{\theta_{\text{old}}}(a|s)\, \nabla_{\theta} \log \pi_{\theta_{\text{old}}}(a|s)^{\top}\Big] \tag{84}$$

4. perform a natural policy gradient step:

$$\begin{aligned}\theta_{n+1} &= \theta_n - \alpha \mathrm{F}^{-1}\nabla_{\theta_n}\mathcal{L}(\theta_n) \\ &= \theta_n - \alpha\, \mathrm{F}^{-1}\nabla_{\theta}\Big(\mathbb{E}\big[\frac{\pi_{\theta}(a|s)}{\pi_{\theta_{\text{old}}}(a|s)} A^{\pi_{\theta_{\text{old}}}}(s,a)\big]\Big)\end{aligned} \tag{85}$$

5. sparatically, update $\pi_{\theta_{\text{old}}}$ with π_{θ_n}

6.5 Compatible Function Approximation

Proposition 1 we let:

$$J(\pi_\theta) \equiv J(\theta) = \sum_{s\in\mathcal{S}} \rho^\pi(s) \sum_{a\in\mathcal{A}} \pi_\theta(a|s) Q^\pi(s,a) \tag{86}$$

and $\tilde{w} = \mathrm{F}^{-1}\nabla_\theta J(\theta)$ to be a single natural policy gradient step, then $\tilde{w}$ also minimize sqaured error:

$$\tilde{w} = \arg\min_w \left(\sum_s d^\pi(s) \sum_a \pi_\theta(a|s) \big(w^\top \nabla_\theta \log \pi(a|s,\theta) - Q^\pi(s,a)\big)^2 \right) \tag{87}$$

interpretation: Good actions, i.e., those with large $Q^\pi(s,a)$ value should have feature vectors $\nabla_\theta \log \pi(a|s,\theta)$ that have a large inner product with the natural gradient $\tilde{w}$.

- We start the reverse: let $\tilde{w}$ minimize sqaured error:

$$\tilde{w} = \arg\min_w \left(\sum_s d^\pi(s) \sum_a \pi_\theta(a|s) \big(w^\top \nabla_\theta \log \pi(a|s,\theta) - Q^\pi(s,a)\big)^2 \right)$$

- then,

$$\begin{aligned}
&\nabla_w \epsilon(\tilde{w}) = 0 \\
\nabla_w \epsilon(w) = \nabla_w &\left(\sum_s \rho^\pi(s) \sum_a \pi_\theta(a|s) \big(\nabla_\theta \log \pi_\theta(a|s)^\top w - Q^\pi(s,a)\big)^2 \right) \\
&\implies \sum_s \rho^\pi(s) \sum_a \pi_\theta(a|s) \nabla_\theta \log \pi_\theta(a|s) \big(\nabla_\theta \log \pi_\theta(a|s)^\top \tilde{w} - Q^\pi(s,a)\big) = 0 \\
&\implies \underbrace{\sum_s \rho^\pi(s) \sum_a \pi_\theta(a|s) \nabla_\theta \log \pi_\theta(a|s) \nabla_\theta \log \pi_\theta(a|s)^\top \tilde{w}}_{F(\theta)} \\
&\qquad\qquad = \sum_s \rho^\pi(s) \sum_a \underline{\pi_\theta(a|s) \nabla_\theta \log \pi_\theta(a|s)} Q^\pi(s,a) \\
&\qquad\qquad = \underbrace{\sum_s \rho^\pi(s) \sum_a \underline{\nabla_\theta \pi_\theta(a|s)} Q^\pi(s,a)}_{\nabla_\theta J(\theta)} \\
&\implies \mathbf{F}\tilde{w} = \nabla_\theta J(\theta) \\
&\implies \tilde{w} = \mathbf{F}^{-1} \nabla_\theta J(\theta)
\end{aligned}$$

7 Solve TRPO $\mathrm{KL}(\pi\|\beta) \leq \delta$

Taylor expansion of the objective equation of the first order:

$$\begin{aligned}
\mathcal{L}_{\theta_k}(\theta) &\approx \underbrace{\mathcal{L}_{\theta_k}(\theta_k)}_{0} + g_{\theta_k}^{\top}(\theta - \theta_k) \\
&= g_{\theta_k}^{\top}(\theta - \theta_k) \qquad \text{where } g_{\theta_k} = \nabla_\theta \mathcal{L}_{\theta_k}(\theta)\,|\theta_k \\
\bar{\mathrm{KL}}(\theta\|\theta_k) &\approx \underbrace{\bar{\mathrm{KL}}(\theta_k\|\theta_k)}_{0} + \underbrace{\nabla_\theta \bar{\mathrm{KL}}(\theta_k\|\theta_k)}_{0} + \frac{1}{2}(\theta - \theta_k)^{\top}\mathrm{F}(\theta_k)(\theta - \theta_k) \\
&= \frac{1}{2}(\theta - \theta_k)^{\top}\mathrm{F}(\theta_k)(\theta - \theta_k) \qquad \text{where } \mathrm{F}(\theta_k) = \nabla_\theta{}^2 \bar{\mathrm{KL}}(\theta\|\theta_k)\,|\theta_k
\end{aligned} \tag{88}$$

7.1 Objective function

The objective function of:

$$\begin{aligned}
&\max_{\pi}\Bigg[\underbrace{\mathbb{E}_{\tau\sim\beta}\Bigg[\sum_{t=0}^{\infty}\gamma^t \frac{\pi(a_t|s_t)}{\beta(a_t|s_t)} A^{\beta}(s_t, a_t)\Bigg]}_{\mathcal{L}_{\theta_k}(\theta)}\Bigg] \\
&\text{s.t.} \quad \mathrm{KL}(\pi\|\beta) \leq \delta
\end{aligned} \tag{89}$$

can be re-formulated as:

$$\begin{aligned}
&\theta_{k+1} = \arg\max_{\theta}\left[g_{\theta_k}^{\top}(\theta - \theta_k)\right] \\
&\text{s.t.} = \frac{1}{2}(\theta - \theta_k)^{\top}\mathrm{F}(\theta_k)(\theta - \theta_k) \leq \delta
\end{aligned} \tag{90}$$

The optimal solution is:

$$\theta_{k+1} = \theta_k + \frac{1}{\sqrt{g_{\theta_k}^{\top}\mathrm{F}(\theta_k)^{-1}g_{\theta_k}}}\mathrm{F}(\theta_k)^{-1}g_{\theta_k} \tag{91}$$

7.2 KKT condition

To make the problem simpler, we let $x \equiv (\theta - \theta_k)$, and we express $g_{\theta_k} \equiv g$:

$$f = \max\left[g^{\top}x \mid \frac{1}{2}x^{\top}\mathrm{F}x \leq \delta, \quad x, c \in \mathbb{R}^n, \quad \mathrm{F} \in \mathbb{R}^{n\times n}\right] \tag{92}$$

The Lagrangian is:

$$\begin{aligned} \mathcal{L}(x,\lambda) &= -g^\top x + \lambda\frac{1}{2}(x^\top \mathrm{F}x - 2\delta) \\ \implies \nabla_x \mathcal{L}(x,\lambda) &= -g + \lambda \mathrm{F}x \end{aligned} \tag{93}$$

The KKT conditions are:

$$-g + \lambda \mathrm{F}x = 0, \quad \lambda \geq 0, \quad \lambda(x^\top \mathrm{F}x - 2\delta) = 0, \quad x^\top \mathrm{F}x \leq 2\delta \tag{94}$$

7.3 Find x^*

- condition $\lambda(x^\top \mathrm{F}x - 2\delta) = 0$ states two cases: if $x^\top \mathrm{F}x < 2\delta \implies \lambda = 0$, and from condition $-g + \lambda \mathrm{F}x = 0 \implies g = 0$, which can not be the max

  Hence we take another case: $\lambda > 0$, $x^\top \mathrm{F}x = 2\delta$

- find expression of λ without having x

$$\begin{aligned} -g + \lambda \mathrm{F}x = 0 \quad &\implies x = \frac{1}{\lambda}\mathrm{F}^{-1}g \\ x^\top \mathrm{F}x &= \left(\frac{1}{\lambda}\mathrm{F}^{-1}g\right)^\top \mathrm{F}\left(\frac{1}{\lambda}\mathrm{F}^{-1}g\right) \\ &= \frac{1}{\lambda^2} g^\top \underbrace{\mathrm{F}^{-1}}_{\text{symmetric}} \mathrm{F}\mathrm{F}^{-1}g = \frac{1}{\lambda^2} g^\top \mathrm{F}^{-1}g = 2\delta \\ &\implies \lambda^2 = \frac{g^\top \mathrm{F}^{-1}g}{2\delta} \\ &\implies \lambda = \sqrt{\frac{g^T \mathrm{F}^{-1}g}{2\delta}} \qquad \text{since} \lambda \geq 0 \end{aligned}$$

- substitute λ in the expression of x:

$$x^* = \frac{1}{\lambda}\mathrm{F}^{-1}g = \sqrt{\frac{2\delta}{g^T \mathrm{F}^{-1}g}}\mathrm{F}^{-1}g$$

7.4 Gradient ascent via finding the maximum first

- solving it using:

$$x \equiv (\theta - \theta_k) \implies x^* \equiv (\theta_{k+1} - \theta_k)$$
$$\implies \theta_{k+1} = \theta_k + \sqrt{\frac{2\delta}{\hat{g}_k \hat{\mathrm{F}}_k^{-1} \hat{g}_k}} \hat{\mathrm{F}}_k^{-1} \hat{g}_k$$

- Computing $\hat{\mathrm{F}}_k^{-1}$ explicitly is too expensive. We would need $\hat{\mathrm{F}}_k^{-1}$ and $\hat{g}_k$ separately, even though in practice we only need the product $\hat{\mathrm{F}}_k^{-1} \hat{g}_k$.

7.4.1 Conjugate gradient ascent helps

First, we know that the following is true:

$$\mathbf{Q}\mathbf{x}^* = \mathbf{b} \quad \implies \quad \mathbf{x}^* = \mathbf{Q}^{-1}\mathbf{b} \tag{95}$$

But the above still involves solving for $\mathbf{Q}^{-1}$. However, if $\mathbf{Q}$ is symmetric and positive semi-definite, which is the case for F, then finding the minimizer $\mathbf{x}^*$ is the same as minimizing the following quadratic form:

$$\arg\min_{\mathbf{x}} \left(\frac{1}{2}\mathbf{x}^\top \mathbf{Q}\mathbf{x} - \mathbf{b}^\top \mathbf{x}\right) \tag{96}$$

The conjugate gradient algorithm is very good at finding the minimizer of the above quadratic form. Please refer to my notes on Conjugate Gradient Descent https://github.com/roboticcam/machine-learning-notes/blob/master/files/conjugate.pdf!

Using the symbols in this section, we set

$$\mathbf{Q} = \hat{\mathrm{F}}_k, \qquad \mathbf{b} = \hat{g}_k, \qquad \mathbf{x} = \theta - \theta_k.$$

Then solving $\mathbf{Q}\mathbf{x} = \mathbf{b}$ is exactly solving

$$\hat{\mathrm{F}}_k \mathbf{x} = \hat{g}_k, \qquad \text{so} \qquad \mathbf{x} = \hat{\mathrm{F}}_k^{-1} \hat{g}_k.$$

7.4.2 Fisher-Vector Product Trick

The Conjugate Gradient (CG) algorithm can solve $Ax = b$ iteratively without ever needing to construct the matrix A explicitly. It only requires a function that computes the matrix-vector product $y = Av$ for any arbitrary vector v.

In TRPO, we need a function that computes $\mathbf{y} = \mathrm{F}\mathbf{v}$. Therefore, we can employ some kind of "Fisher-Vector Product" Trick to compute $\mathbf{y} = \mathrm{F}\mathbf{v}$ efficiently:

$$\begin{aligned} \mathrm{F}\mathbf{v} &= \underbrace{\big(\nabla_\theta(\nabla_\theta \mathrm{KL}(\theta_{\text{old}} \| \theta))^\top\big)}_{\text{matrix}} \mathbf{v} \\ &= \nabla_\theta\big(\underbrace{\nabla_\theta \mathrm{KL}(\theta_{\text{old}} \| \theta)^\top \mathbf{v}}_{\text{scalar}}\big) \end{aligned} \tag{97}$$

8 Proximal Policy Optimization (PPO)

- The standard TRPO objective that PPO is based on is the following constrained optimization problem:

$$\begin{aligned} \max_{\pi_\theta} \quad & \mathbb{E}_{\tau\sim\beta}\left[\sum_{t=0}^{\infty} \gamma^t \frac{\pi_\theta(a_t|s_t)}{\beta(a_t|s_t)} A^\beta(s_t, a_t)\right] \\ \text{subject to} \quad & \mathbb{E}_{s\sim d^\beta}\left[\mathrm{KL}(\beta(\cdot|s) \,\|\, \pi_\theta(\cdot|s))\right] \le \delta. \end{aligned}$$

  The idea is: improve the new policy π_θ using data sampled from the old policy β, but keep π_θ close to β so that the importance ratio $\pi_\theta(a_t|s_t)/\beta(a_t|s_t)$ does not become too extreme.

- PPO is expressed as, using $r_t(\theta) = \frac{\pi_\theta(a|s)}{\beta(a_t|s_t)}$:

$$\max_{\pi} \left[\mathbb{E}_{\tau\sim\beta}\left[\sum_{t=0}^{\infty} \gamma^t \min\left(\underbrace{r_t(\theta) A^\beta(s_t, a_t)}, \underbrace{\mathrm{clip}\big(r_t(\theta), 1-\epsilon, 1+\epsilon\big) A^\beta(s_t, a_t)}\right)\right]\right]$$

- if $r_t(\theta)$ falls outside $(1-\epsilon)$ and $(1+\epsilon)$, $A^\beta(s_t, a_t)$ will be clipped
- sign of $A^\beta(s_t, a_t)$ plays a part:
  1. if $A^\beta(s_t, a_t) > 0$, PPO clips at $r_t(\theta) = 1+\epsilon$
  2. if $A^\beta(s_t, a_t) < 0$, PPO clips at $r_t(\theta) = 1-\epsilon$

8.1 For Language Models

For Language Models, the objective function is expressed as, where Q is the question and O is the answer/Output, therefore, the question q and all the generated words $o_{<t}$ form the state $s = (q, o_{1:t})$.

$$\mathcal{J}_{PPO}(\theta) = \mathbb{E}_{q\sim P(Q),o\sim\pi_{\theta_{old}}(O|q)} \left[\frac{1}{|o|} \sum_{t=1}^{|o|} \min \left[\frac{\pi_\theta(o_t|q,o_{<t})}{\pi_{\theta_{old}}(o_t|q,o_{<t})} A_t, \text{clip} \left(\frac{\pi_\theta(o_t|q,o_{<t})}{\pi_{\theta_{old}}(o_t|q,o_{<t})}, 1-\epsilon, 1+\epsilon \right) A_t \right] \right] \tag{98}$$

9 GRPO (DeepSeek-R1)

For each question q, GRPO samples a group of outputs $\{o_1, o_2, \cdots, o_G\}$ from the old policy $\pi_{\theta_{old}}(O|q)$, and then optimizes the policy model π_θ by maximizing the following objective:

$$\mathcal{J}_{GRPO}(\theta) = \mathbb{E}_{q\sim P(Q),\{o_i\}_{i=1}^{G}\sim\pi_{\theta_{old}}(O|q)}$$
$$\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\left\{\min\left[\frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{old}}(o_{i,t}|q,o_{i,<t})}A_{i,t}, \text{clip}\left(\frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{old}}(o_{i,t}|q,o_{i,<t})}, 1-\epsilon, 1+\epsilon\right)A_{i,t}\right] - \beta\mathbb{D}_{KL}[\pi_\theta||\pi_{ref}]\right\} \tag{99}$$

where

$$A_i = \frac{r_i - \text{mean}(\{r_1, r_2, \cdots, r_G\})}{\text{std}(\{r_1, r_2, \cdots, r_G\})} \tag{100}$$

The main idea of GRPO is to replace the value-function critic used in PPO with a group-relative baseline. For a fixed question q, the model generates several candidate answers $\{o_1, \ldots, o_G\}$. Each answer receives a scalar reward r_i, and the advantage A_i measures whether answer o_i is better or worse than the other answers generated for the same question. Thus, GRPO does not ask whether an answer is good in an absolute sense; it asks whether it is good relative to the current group.

This is useful for large language models because training a separate critic/value model is expensive and unstable. In PPO, the advantage is usually estimated by a learned value function. In GRPO, the group mean reward plays the role of the baseline:

$$r_i - \text{mean}(\{r_1, \ldots, r_G\}).$$

The division by the group standard deviation normalizes the scale of advantages, making updates more stable across different questions whose reward ranges may be very different.

In the objective above, A_i is usually a sequence-level advantage for the whole output o_i. When it appears inside the token-level sum as $A_{i,t}$, it is commonly understood as broadcasting the same group advantage to every token in that output, i.e. $A_{i,t} = A_i$. Tokens from outputs with positive relative reward are pushed up in probability, while tokens from outputs with negative relative reward are pushed down. The clipping term plays the same role as in PPO: it prevents the new policy π_θ from moving too far away from the old policy $\pi_{\theta_{old}}$ in a single update.

10 Direct Preference Optimization (DPO)

Direct Preference Optimization (DPO) provides a stable, analytical alternative to Reinforcement Learning from Human Feedback (RLHF). Unlike standard RLHF, which requires training a separate reward model and optimizing a policy via PPO, DPO re-parameterizes the reward function in terms of the optimal policy, allowing for a direct optimization objective.

10.1 The Standard RLHF Objective

We begin with the standard RLHF objective: maximize the expected reward while maintaining a KL-divergence constraint relative to a reference model π_{ref} (typically the supervised fine-tuned model). This constraint prevents mode collapse and ensures the policy does not drift too far from the distribution on which the reward model was trained.

The objective is:

$$\max_{\pi} \mathbb{E}_{s\sim\mathcal{D},a\sim\pi(\cdot|s)} \left[r(s,a) - \beta \log \frac{\pi(a|s)}{\pi_{\text{ref}}(a|s)} \right] \tag{101}$$

where β is the KL penalty coefficient (inverse temperature).

10.2 The Analytical Solution

For a fixed reward function $r(s,a)$, the optimization problem in Eq. (101) has a known closed-form solution. The optimal policy π^* takes the form of a Gibbs distribution:

$$\pi^*(a|s) = \frac{1}{Z(s)} \pi_{\text{ref}}(a|s) \exp\left(\frac{1}{\beta} r(s,a)\right) \tag{102}$$

where $Z(s)$ is the partition function:

$$Z(s) = \sum_a \pi_{\text{ref}}(a|s) \exp\left(\frac{1}{\beta} r(s,a)\right) \tag{103}$$

10.3 Re-parameterizing the Reward

In standard RLHF, we estimate $r(s,a)$ via a neural network and use RL to approximate π^*. DPO inverts this relationship. We solve Eq. (102) for the reward function $r(s,a)$ in terms of the optimal policy π^* and the reference policy π_{ref}:

$$\pi^*(a|s) \cdot Z(s) = \pi_{\text{ref}}(a|s) \exp\left(\frac{1}{\beta} r(s,a)\right) \tag{104}$$

$$\frac{\pi^*(a|s)}{\pi_{\text{ref}}(a|s)} Z(s) = \exp\left(\frac{1}{\beta} r(s,a)\right) \tag{105}$$

$$\log \frac{\pi^*(a|s)}{\pi_{\text{ref}}(a|s)} + \log Z(s) = \frac{1}{\beta} r(s,a) \tag{106}$$

Thus, the reward can be expressed analytically as:

$$r(s,a) = \beta \log \frac{\pi^*(a|s)}{\pi_{\text{ref}}(a|s)} + \beta \log Z(s) \tag{107}$$

10.4 The Bradley-Terry Model Substitution

We assume human preferences follow the Bradley-Terry model. Given a state s and two actions a_w (winner) and a_l (loser), the probability that a_w is preferred is given by:

$$P(a_w \succ a_l \mid s) = \sigma(r(s,a_w) - r(s,a_l)) \tag{108}$$

where σ is the logistic sigmoid function.

We now substitute the expression for $r(s,a)$ from Eq. (107) into the preference model. Crucially, the partition function term $\beta \log Z(s)$ cancels out because it depends only on the state s, not on the action a.

$$r(s,a_w) - r(s,a_l) = \left(\beta \log \frac{\pi^*(a_w|s)}{\pi_{\text{ref}}(a_w|s)} + \beta \log Z(s)\right) - \left(\beta \log \frac{\pi^*(a_l|s)}{\pi_{\text{ref}}(a_l|s)} + \beta \log Z(s)\right) \tag{109}$$

$$= \beta \log \frac{\pi^*(a_w|s)}{\pi_{\text{ref}}(a_w|s)} - \beta \log \frac{\pi^*(a_l|s)}{\pi_{\text{ref}}(a_l|s)} \tag{110}$$

10.5 The Final DPO Loss

Finally, we replace the theoretical optimal policy π^* with our parameterized policy network π_θ. We minimize the negative log-likelihood of the preference data $\mathcal{D} = \{s^{(i)}, a_w^{(i)}, a_l^{(i)}\}_{i=1}^N$:

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(s,a_w,a_l)\sim\mathcal{D}}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(a_w|s)}{\pi_{\text{ref}}(a_w|s)} - \beta \log \frac{\pi_\theta(a_l|s)}{\pi_{\text{ref}}(a_l|s)}\right)\right] \tag{111}$$

This loss function allows us to optimize the policy directly on preference data without training an explicit reward model or using reinforcement learning.