Robot Learning	**Assignment 3**
CS 4756 Fall 2024	**Due 11:59pm Thursday, October 24**

(1) Policy Gradients (10 points)

Recap: Recall that the goal of RL is to learn some θ^* that maximizes the objective function:

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta(\tau)}\big[r(\tau)\big] \tag{1}$$

where each τ is a rollout of length T_τ and $r(\tau) = \sum_{t=0}^{T_\tau - 1} r(s_t, a_t)$ is the reward for that rollout. $\pi_\theta(\tau)$ is the probability of the rollout under policy π_θ, i.e. $\pi_\theta(\tau) = \Pr[s_0]\pi_\theta(a_0|s_0)\prod_{t=1}^{T_\tau - 1}\Pr[s_t|s_{t-1}, a_{t-1}]\pi_\theta(a_t|s_t)$.

The policy gradient approach requires that we take the gradient of this objective as follows:

$$\nabla_\theta J(\theta) = \nabla_\theta \int \pi_\theta(\tau) r(\tau)\, d\tau = \int \pi_\theta(\tau) \nabla_\theta \log \pi_\theta(\tau) r(\tau)\, d\tau \tag{2}$$

$$= \mathbb{E}_{\tau \sim \pi_\theta(\tau)}\big[\nabla_\theta \log \pi_\theta(\tau) r(\tau)\big] \tag{3}$$

The gradient can further be refined by noting that future actions do not affect past rewards (the causality assumption), resulting in the following "reward-to-go" formulation:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta(\tau)}\left[\sum_{t=0}^{T_\tau - 1}\left(\nabla_\theta \log \pi_\theta(a_t|s_t) \cdot \sum_{t'=t}^{T_\tau - 1} r(s_{t'}, a_{t'})\right)\right] \tag{4}$$

In this question, we consider a toy MDP and get familiar with computing policy gradients.

(a) Show the following step in 2 holds true:

$$\nabla_\theta \int \pi_\theta(\tau) r(\tau)\, d\tau = \int \pi_\theta(\tau) \nabla_\theta \log \pi_\theta(\tau) r(\tau)\, d\tau$$

and explain why this step is valid.

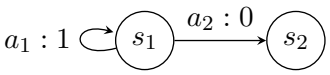
(b) Starting from Equation 3, use the causality assumption (that future actions do not affect past rewards) to derive the "reward-to-go" formulation given in Equation 4. Show the intermediate steps in your derivation.

(c) Introduce a baseline $b(s_t)$ to reduce the variance of the policy gradient estimator, leading to the advantage function $A(s_t, a_t) = Q(s_t, a_t) - b(s_t)$. Show that subtracting a baseline does not introduce bias in the gradient estimation, i.e., prove that:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta(\tau)}\left[\sum_{t=0}^{T_\tau - 1} \nabla_\theta \log \pi_\theta(a_t|s_t) \cdot A(s_t, a_t)\right] \tag{5}$$

Explain why the expectation of the baseline term vanishes.

Now, consider the following infinite-horizon MDP.



The initial state is always s_1, and the episode terminates when s_2 is reached. The agent receives reward 1 for taking action a_1 and reward 0 for taking action a_2. In this case, we can define the policy with a single parameter θ:

$$\pi_\theta(a_1|s_1) = \theta, \quad \pi_\theta(a_2|s_1) = 1 - \theta$$

(d) Use policy gradients to compute the gradient of the expected return of π_θ with respect to the parameter θ (Eq. 3). Do not use discounting. The sum should telescope to a closed form solution.
You may find this fact useful:

$$\sum_{k=1}^{\infty} k\alpha^{k-1} = \frac{d}{d\alpha} \sum_{k=1}^{\infty} \alpha^k$$

(e) Compute the expected return of the policy π_θ directly (Eq. 1). Compute the gradient of this expression and verify that it matches your result in **(d)**.

(f) Reward-to-go can be helpful and improve the statistical qualities of our policy gradient. Apply reward-to-go as an advantage estimator. Write the new policy gradient (Eq. 4), and verify that it is unbiased.

(2) Bounding Error in Approximate Policy Iteration (10 points)

In this problem, we look at how errors in approximating the value function affect the performance of a policy during policy iteration. We aim to understand how the error ϵ in the value function propagates and impacts the overall performance of the policy.

Let's assume we are in an infinite horizon MDP with discount factor γ. We have a reference policy π whose true value function is $V^{\pi}(s)$. We collect rollouts with π, and fit a neural network to approximate this value function, where $\hat{V}(s) \approx V^{\pi}(s)$, given that the value function approximation error is bounded by ϵ.

Let's assume we did a really good job and can guarantee that the error from the fit is at most ϵ. More formally, let $||V^{\pi} - \hat{V}||_{\infty} \leq \epsilon$.[1]

We now choose a greedy policy to improve upon policy $\hat{\pi}$:

$$\hat{\pi}(s) = \underset{a}{\text{argmax}} \left[R(s,a) + \gamma \sum_{s'} P(s'|s,a)\hat{V}(s') \right]$$

Note that this is exactly the policy improvement step, except the value function is substituted with our approximate value function. We want to know how the greedy policy $\hat{\pi}(s)$ performs with respect to $\pi(s)$.

In other words, $\hat{\pi}$ can end up doing much worse than π. This shows that even though the error in approximating the value function is at most ϵ, the performance error scales up by a factor of $\frac{1}{1-\gamma}$.

(a) Let $V^{\hat{\pi}}(s)$ be the value of the greedy policy $\hat{\pi}(s)$. Prove the following:

$$V^{\pi}(s) - V^{\hat{\pi}}(s) \leq \frac{2\gamma\epsilon}{1-\gamma} \text{ , for all } s$$

In other words, $\hat{\pi}$ can end up doing much worse than π. Additionally, even though the error from fitting the value was ϵ, the performance error scales up by a factor of $\frac{1}{1-\gamma}$.
Hint: One way to approach the question would be:

1. Start by using the Bellman equation for any policy:

   $$V^{\pi}(s) = R(s,\pi(s)) + \gamma \sum_{s'} P(s'|s,\pi(s))V^{\pi}(s')$$

   Use this substitution to expand $V^{\pi}(s) - V^{\hat{\pi}}(s)$.

2. Next, note that you need to establish a relationship between $\pi(s)$ and $\hat{\pi}(s)$. Exploit the following observation: $\hat{\pi}(s) = \text{argmax}_a f(s,a)$ must imply $f(s,\hat{\pi}(s)) \geq f(s,\pi(s))$ for any policy π.

3. Use these facts to obtain the following intermediate result. For any s,

   $$V^{\pi}(s) - V^{\hat{\pi}}(s) \leq 2\gamma\epsilon + \gamma \sum_{s' \in \mathcal{S}} \Pr[s'|s,\hat{\pi}(s)](V^{\pi}(s') - V^{\hat{\pi}}(s'))$$

   From the above, you should be able to prove the final result for all s.

[1] Note: $||x||_{\infty}$ is the L-infinity norm

(b) Explain why the performance error bound between $\pi(s)$ and $\hat{\pi}(s)$ is not simply ϵ. Does the error remain constant, scale with time, or compound over time? Why is the scaling factor $\frac{1}{1-\gamma}$ significant in this bound?

Hint: Think about how the value function accumulates rewards over an infinite horizon and how errors in each step affect the future value estimates.

(c) What would happen to the performance bound between if the time horizon T were finite instead of infinite? How would the bound change, and what implications does this have for practical applications of approximate policy iteration?

(3) Reward Shaping with an Approximate Value Function (5 points)

Previously, we saw how acting greedily with an approximate value function can result in a *worse* policy. Instead, what if we use the approximate value function to *shape the reward*?

Let's define a *reward bonus* using the approximate value function from Q2.

$$F(s, s') = \gamma \hat{V}(s') - \hat{V}(s)$$

This extra reward is gained whenever we transition from state s to state s'. Informally, we are giving a small intermediate reward for moving toward states of higher value. Adding these intermediate rewards helps in speeding up a policy's convergence in environments with sparse rewards.
At each step i of an episode, the shaped reward R_i is then defined as

$$R_i = r_i + F(s_i, s_{i+1})$$

where r_i is the base reward $r(s_i, a_i)$ received for step i. We continue to use a *discount factor* of γ in computing total reward. Recall that for an infinite-horizon setting, the total cumulative reward is typically expressed as:

$$R = \sum_{i=0}^{\infty} \gamma^i r_i$$

In this problem, we will explore how this changes when shaping the reward with the approximate value function.

(a) Consider a given episode of potentially infinite-length, of visited states $s_0, s_1, ...$
Write out the total reward received in the shaped environment, expressed in terms of the total reward that would have been accrued in the unshaped environment. What is noticeable about this relationship?

(b) The policy $\hat{\pi}$ is found by optimizing the shaped rewards, while the policy π^* is found by optimizing the unshaped rewards. Although the policies are derived using different reward structures, we ultimately want to compare their performance using the same value function.

Explain why the performance of the optimal policy $\hat{\pi}$ computed with the shaped rewards is the same as the performance of the optimal policy π^* computed with the unshaped rewards. Specifically, explain why:

$$||V^{\pi^*} - V^{\hat{\pi}}||_\infty = 0$$

Hint: Use your interpretation from part (a) to reason about why the performance is unaffected by reward shaping. You can either use the math or provide an explanation based on the takeaway from part (a).

(4) (Mandatory for 5756): Off Policy Gradient Estimation (10 points)

In this problem, you will work towards deriving the off-policy gradient of a policy using importance weighting techniques. Consider a finite-horizon MDP $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, P, R, H, \mu_0\}$, with a policy π_θ that we want to optimize, and a different policy π' that generates the trajectory data.

The trajectory $\tau = \{s_0, a_0, s_1, a_1, \ldots, s_{H-1}, a_H\}$ is sampled using the policy π', and the objective is to compute the gradient of π_θ using this off-policy data. The reward function is defined as the sum of rewards over the trajectory, $R(\tau) = \sum_{t=0}^{H-1} r(s_t, a_t)$, and the goal is to maximize the expected cumulative reward. Recall that the gradient of the objective function can be expressed as:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\nabla_\theta \log \pi_\theta(\tau) R(\tau) \right]$$

where $\pi_\theta(\tau) = \mu_0(s_0) \prod_{t=0}^{H-1} \pi_\theta(a_t|s_t) P(s_{t+1}|s_t, a_t)$.

Since the data is collected under a different policy π', we need to use importance weighting to correct for the distribution mismatch. This will allow us to estimate the policy gradient for π_θ using data collected from π'.

(a) Show that the policy gradient can be written using importance weights as:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi'} \left[\frac{\pi_\theta(\tau)}{\pi'(\tau)} \nabla_\theta \log \pi_\theta(\tau) R(\tau) \right]$$

Hint: Multiply and divide by $\pi'(\tau)$, and use the fact that the expectation over π_θ can be transformed into an expectation over π' using importance sampling.

(b) Derive an expression for the ratio $\frac{\pi_\theta(\tau)}{\pi'(\tau)}$ in terms of the individual action probabilities $\pi_\theta(a_t|s_t)$ and $\pi'(a_t|s_t)$ for $t = 0, \ldots, H-1$.

Hint: Use the fact that the probability of a trajectory is the product of action probabilities and transition probabilities under the respective policies.

(c) Now, derive an expression for $\nabla_\theta \log \pi_\theta(\tau)$ in terms of $\nabla_\theta \log \pi_\theta(a_t|s_t)$ for $t = 0, \ldots, H-1$.

(d) What are the key benefits of using importance weighting to estimate the gradient of a target policy π_θ using data collected under a different policy π'? Why is it useful in practical settings?