ESE 6510 March 5 Exercises

Problem 1: PPO Surrogate Loss Derivation (TA instruction)

The standard policy gradient objective for a trajectory $\tau = (s_0, a_0, \ldots, s_T)$ is:

$$J(\theta) = \mathbb{E}_{\tau \sim p_\theta}[R(\tau)]$$

where the trajectory density factorizes as:

$$p_\theta(\tau) = \rho(s_0) \prod_{t=0}^{T-1} \pi_\theta(a_t \mid s_t)\, P(s_{t+1} \mid s_t, a_t)$$

Starting from the standard policy gradient objective, derive the gradient of the importance-sampled surrogate objective used in PPO.

Step 1. Write the objective as an integral.
Expanding the expectation explicitly:

$$J(\theta) = \int p_\theta(\tau)\, R(\tau)\, d\tau$$

Step 2. Take the gradient.
Differentiating with respect to θ:

$$\nabla_\theta J(\theta) = \int \nabla_\theta\, p_\theta(\tau)\, R(\tau)\, d\tau$$

Step 3. Apply the log-derivative trick.
Using the identity $\nabla_\theta\, p_\theta(\tau) = p_\theta(\tau)\, \nabla_\theta \log p_\theta(\tau)$:

$$\begin{aligned}\nabla_\theta J(\theta) &= \int p_\theta(\tau)\, \nabla_\theta \log p_\theta(\tau)\, R(\tau)\, d\tau \\ &= \mathbb{E}_{\tau \sim p_\theta}\Big[\nabla_\theta \log p_\theta(\tau)\, R(\tau)\Big]\end{aligned}$$

Step 4. Expand and simplify $\nabla_\theta \log p_\theta(\tau)$.
Taking the logarithm of the trajectory density:

$$\log p_\theta(\tau) = \log \rho(s_0) + \sum_{t=0}^{T-1} \Big[\log \pi_\theta(a_t \mid s_t) + \log P(s_{t+1} \mid s_t, a_t)\Big]$$

The initial state distribution $\rho(s_0)$ and the dynamics $P(s_{t+1} \mid s_t, a_t)$ do not depend on θ, so their gradients vanish:

$$\nabla_\theta \log p_\theta(\tau) = \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

Substituting back:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim p_\theta}\left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t \mid s_t)\, R(\tau)\right]$$

This is the standard REINFORCE policy gradient estimator.

Step 5. Replace $R(\tau)$ with the advantage function.
By the "reward-to-go" argument, where future actions cannot affect past rewards, and by introducing a state-dependent baseline $V^{\pi_\theta}(s_t)$, we can safely replace $R(\tau)$ with the advantage $A^{\pi_\theta}(s_t, a_t)$ without changing the expectation of the gradient:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim p_\theta}\left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t \mid s_t)\, A^{\pi_\theta}(s_t, a_t)\right]$$

Step 6. Importance sampling.
We have data collected under an *old* policy $\pi_{\theta_{\text{old}}}$, but need to estimate the gradient for our *new* parameters θ. We apply importance sampling to each timestep's expectation over (s_t, a_t):

$$\mathbb{E}_{a_t \sim \pi_\theta(\cdot|s_t)}\big[f(s_t, a_t)\big] = \mathbb{E}_{a_t \sim \pi_{\theta_{\text{old}}}(\cdot|s_t)}\left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}\, f(s_t, a_t)\right]$$

Applying this to each term:

$$\nabla_\theta J(\theta) \approx \mathbb{E}_{\tau \sim p_{\theta_{\text{old}}}}\left[\sum_{t=0}^{T-1} \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} \nabla_\theta \log \pi_\theta(a_t \mid s_t)\, A^{\pi_{\theta_{\text{old}}}}(s_t, a_t)\right]$$

Step 7. Recognize this as the gradient of a surrogate objective.
Observe the key identity:

$$\nabla_\theta \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} = \frac{\nabla_\theta \pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

This means the expression in Step 6 is exactly the gradient of the following surrogate objective:

$$L^{\text{PG}}(\theta) = \mathbb{E}_{\tau \sim p_{\theta_{\text{old}}}}\left[\sum_{t=0}^{T-1} \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}\, A^{\pi_{\theta_{\text{old}}}}(s_t, a_t)\right]$$

Problem 2: The PPO Clipped Objective (Gradescope)

PPO replaces the unclipped surrogate $r_t(\theta)\,\hat{A}_t$ with:

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t\Big[\min\Big(r_t(\theta)\,\hat{A}_t,\ \text{clip}(r_t(\theta),\,1-\varepsilon,\,1+\varepsilon)\,\hat{A}_t\Big)\Big]$$

Let $\ell_t(\theta) = \min(r_t\hat{A}_t,\ \text{clip}(r_t, 1-\varepsilon, 1+\varepsilon)\hat{A}_t)$. Determine $\nabla_\theta \ell_t(\theta)$ by analyzing all four cases.

Case 1: $\hat{A}_t > 0$ **and** $r_t \le 1+\varepsilon$

Case 2: $\hat{A}_t > 0$ **and** $r_t > 1+\varepsilon$

Case 3: $\hat{A}_t < 0$ **and** $r_t \ge 1-\varepsilon$

Case 4: $\hat{A}_t < 0$ **and** $r_t < 1-\varepsilon$

Case 1: $\hat{A}_t > 0$ **and** $r_t \le 1+\varepsilon$
The gradient flows normally. The update encourages increasing $\pi_\theta(a_t \mid s_t)$ since $\hat{A}_t > 0$.

Case 2: $\hat{A}_t > 0$ **and** $r_t > 1+\varepsilon$
The gradient is zero. The policy is not rewarded for further increasing the probability of an already successful action.

Case 3: $\hat{A}_t < 0$ **and** $r_t \ge 1-\varepsilon$
Since $\hat{A}_t < 0$, this pushes toward decreasing $\pi_\theta(a_t \mid s_t)$. The gradient flows normally.

Case 4: $\hat{A}_t < 0$ **and** $r_t < 1-\varepsilon$
Zero gradient. The policy is not penalized further for an action whose probability has already been sufficiently reduced.

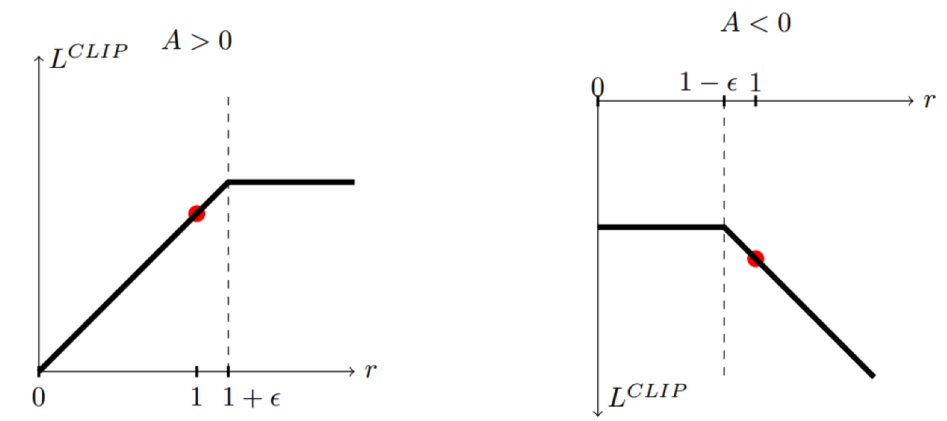

Figure 1: PPO Clipped Surrogate Objective

Problem 3: Fitted Value Iteration and Q-Learning (Gradescope)

Consider an MDP with continuous state space $\mathcal{S} \subseteq \mathbb{R}^n$, finite action space $\mathcal{A} = \{a_1, \ldots, a_m\}$, known transition dynamics $p(s' \mid s, a)$, reward $r(s, a)$, and discount $\gamma \in [0, 1)$.

(a) **Fitted Value Iteration (FVI).** We approximate the optimal value function with a parameterized $V_\phi(s)$. FVI is often presented with *deterministic* dynamics $s' = f(s, a)$. Now suppose the dynamics are known but *stochastic*: given (s, a), the next state is drawn from a known distribution $p(s' \mid s, a)$.

Write down the pseudocode for the Fitted Value Iteration algorithm adapted to this stochastic setting.

(b) **Q-Learning.** Now suppose the dynamics are *unknown*. We have transition data $\{(s_i, a_i, r_i, s_i')\}_{i=1}^N$ collected by some behavior policy, and parameterize $Q_\phi(s, a)$ with a neural network.

(i) Write the Bellman residual loss $\mathcal{L}(\phi)$ and the corresponding targets y_i.

(ii) Consider the Bellman residual loss for a *single transition* (s, a, r, s'). Compute the *full* gradient $\nabla_\phi \mathcal{L}(\phi)$ (differentiating through both occurrences of ϕ). Then explain why in practice we drop the gradient through the target term $\max_{a'} Q_\phi(s', a')$, and write the resulting *semi-gradient* update rule for ϕ.

(iii) Explain why vanilla Q-learning experiences training instability, and describe two stabilization techniques.

Part (a):

The key modification is in the regression targets. In the deterministic case, $V_\phi(f(s_i, a))$ evaluates the value at the single known next state. With stochastic dynamics, we replace this with an expectation over all possible next states weighted by their transition probabilities.

Algorithm: Fitted Value Iteration with Stochastic Dynamics

1. **Collect** states $\{s_i\}_{i=1}^N$ using some policy.
2. **repeat**
3. **Set targets:** for each s_i, compute

$$y_i = \max_{a \in \mathcal{A}} \left(r(s_i, a) + \gamma \sum_{s'} p(s' \mid s_i, a)\, V_\phi(s') \right)$$

4. **Compute gradient:**

$$\hat{g} = \frac{\partial}{\partial \phi} \sum_i \left\| V_\phi(s_i) - y_i \right\|^2$$

5. **Update parameters:** $\phi^{\text{new}} = \phi^{\text{old}} - \alpha \cdot \hat{g}$
6. **until** converged

The change from the deterministic version is that the target replaces $V_\phi(f(s_i, a))$ with $\mathbb{E}_{s' \sim p(\cdot|s_i,a)}[V_\phi(s')] = \sum_{s'} p(s' \mid s_i, a)\, V_\phi(s')$. This is the Bellman equation: $V^*(s) = \max_a \big[r(s,a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)}[V^*(s')]\big]$. We require knowledge of the dynamics $p(s' \mid s, a)$ to compute this expectation.

Part (b):

(i) The Bellman residual loss is:

$$\mathcal{L}(\phi) = \sum_{i=1}^{N} \big(Q_\phi(s_i, a_i) - y_i\big)^2, \qquad y_i = r_i + \gamma \max_{a'} Q_\phi(s_i', a').$$

(ii) The Bellman residual loss for a single transition (s, a, r, s'):

$$L(\phi) = \frac{1}{2}\big(Q_\phi(s, a) - y\big)^2, \qquad y = r + \gamma \max_{a'} Q_\phi(s', a')$$

Taking the full gradient with chain rule:

$$\nabla_\phi L(\phi) = \big(Q_\phi(s, a) - y\big) \cdot \nabla_\phi \big(Q_\phi(s, a) - y\big)$$

$$= \big(Q_\phi(s, a) - y\big) \cdot \Big(\nabla_\phi\, Q_\phi(s, a) - \gamma\, \nabla_\phi \max_{a'} Q_\phi(s', a')\Big)$$

Assuming greedy action $a^* = \arg\max_{a'} Q_\phi(s', a')$:

$$\nabla_\phi L(\phi) = \big(Q_\phi(s, a) - y\big) \cdot \Big(\nabla_\phi\, Q_\phi(s, a) - \gamma\, \nabla_\phi\, Q_\phi(s', a^*)\Big)$$

Notice how the second term can lead to spurious points of zero gradient. In practice we avoid differentiating through the target y. Each gradient step shifts what we are regressing toward, causing instability. Treating y as a constant gives the semi-gradient update:

$$\phi \;\leftarrow\; \phi - \alpha\big(Q_\phi(s, a) - y\big)\, \nabla_\phi\, Q_\phi(s, a)$$