

---

# DEEP REINFORCEMENT LEARNING

## HOMEWORK 3 | ACTOR-CRITIC METHODS

---

**Professor Mohammad Hossein Rohban**

Sharif University of Technology  
Spring 2026



# Contents

|          |                                                                      |           |
|----------|----------------------------------------------------------------------|-----------|
| <b>1</b> | <b>Deterministic Actor-Critic: From DPG to TD3</b>                   | <b>1</b>  |
| 1.1      | Background: Stochastic Policy Gradients . . . . .                    | 1         |
| 1.2      | Action-Value Gradients . . . . .                                     | 2         |
| 1.3      | Motivation for Deterministic Policy Gradients . . . . .              | 2         |
| 1.4      | The Deterministic Policy Gradient Theorem . . . . .                  | 2         |
| 1.5      | Benefits of Deterministic Policy Gradients . . . . .                 | 3         |
| 1.6      | Deterministic Actor-Critic Algorithms . . . . .                      | 4         |
| 1.7      | Compatible Function Approximation . . . . .                          | 5         |
| 1.8      | From DPG to DDPG: Practical Challenges . . . . .                     | 5         |
| 1.9      | Deep Deterministic Policy Gradient (DDPG) . . . . .                  | 6         |
| 1.10     | Function Approximation Error in Deterministic Actor-Critic . . . . . | 7         |
| 1.11     | TD3: Addressing Function Approximation Error . . . . .               | 8         |
| 1.12     | Clipped Double Q-Learning . . . . .                                  | 8         |
| 1.13     | Delayed Policy Updates . . . . .                                     | 9         |
| 1.14     | Target Policy Smoothing . . . . .                                    | 9         |
| <b>2</b> | <b>Soft Actor-Critic</b>                                             | <b>11</b> |
| 2.1      | Maximum Entropy Framework . . . . .                                  | 11        |
| 2.2      | Soft Policy Evaluation . . . . .                                     | 12        |
| 2.3      | Soft Policy Improvement . . . . .                                    | 13        |
| 2.4      | Soft Policy Iteration . . . . .                                      | 13        |
| 2.5      | Soft Actor-Critic Losses . . . . .                                   | 13        |
| 2.6      | Reparameterized Gradient . . . . .                                   | 14        |
| 2.7      | Temperature Tuning . . . . .                                         | 15        |
| <b>3</b> | <b>Asynchronous Advantage Actor-Critic (A3C)</b>                     | <b>16</b> |
| 3.1      | Variance Reduction in Policy Gradients . . . . .                     | 16        |
| 3.2      | Multi-Step Advantage Estimation . . . . .                            | 17        |
| 3.3      | Synchronous Advantage Actor-Critic (A2C) . . . . .                   | 17        |
| 3.4      | Entropy Regularization . . . . .                                     | 18        |
| 3.5      | Asynchronous Advantage Actor-Critic (A3C) . . . . .                  | 19        |

## Important Notes

- **Grading Policy:** This homework is considered complete once you reach a total of **100 points**. Up to *30 additional bonus points* may be earned; however, any score above 100 will still be recorded as 100. Bonus points do *not* transfer or overflow to other assignments.
- **Homework Deadline:** The assignment is due on the posted deadline. Please make sure to submit your work on time.
- **Delay Usage Policy:** You may use *at most 3 days* of your allowed late-day budget for this homework. Late days will be automatically applied if you submit after the deadline.
- **Solution Release:** Official solutions will be released exactly **3 days after the deadline**. No submissions will be accepted once the solutions are published.
- **Cheating and Collaboration Policy:** You must write all solutions in your own words and produce all code independently. Discussing general ideas is allowed, but sharing written solutions, code, or using someone else's work (including from online sources, previous semesters, or AI tools without proper authorization) is strictly prohibited. Violations will result in academic misconduct procedures.

## Advice

*Please avoid asking LLMs for the answer. Sure, doing so might give you an immediate reward of 5 points, but in the long run, it won't help you become the next Rockstar of DRL. Remember the key lesson of this course: chasing immediate rewards is rarely the best strategy. Instead, struggle with the problems, read the papers, and truly understand the algorithms. The satisfaction of solving a challenging problem yourself is far greater than any grade. Good luck! [1]*

# Grading Breakdown

| Task / Criteria                                                              | Points           |
|------------------------------------------------------------------------------|------------------|
| <b>Deterministic Actor-Critic: From DPG to TD3</b>                           | <b>58 pts</b>    |
| Action-Value Gradient Derivation                                             | 1 pt             |
| Proof of the DPG Theorem                                                     | 8 pts            |
| Deterministic On-Policy Actor-Critic                                         | 2 pts            |
| Deterministic Off-Policy Actor-Critic                                        | 3 pts            |
| <i>Implementation:</i> Deterministic Policy Gradient (DPG)                   | 5 pts            |
| Proof of Compatible Function Approximation                                   | 3 pts            |
| DDPG Component Analysis                                                      | 3 pts            |
| <i>Implementation:</i> Deep Deterministic Policy Gradient (DDPG)             | 5 pts            |
| Critic Gradient Error and Actor Bias                                         | 2 pts            |
| Overestimation and Policy Performance                                        | 2 pts            |
| Sources of Overestimation Bias                                               | 2 pts            |
| Minimum of Two Q-Estimates                                                   | 3 pts            |
| Underestimation vs. Overestimation                                           | 2 pts            |
| Delayed Policy Updates                                                       | 2 pts            |
| Target Policy Smoothing Intuition                                            | 2 pts            |
| Target Smoothing and Regularization                                          | 3 pts            |
| <i>Implementation:</i> Twin Delayed Deep Deterministic Policy Gradient (TD3) | 10 pts           |
| <b>Soft Actor-Critic</b>                                                     | <b>39 pts</b>    |
| Maximum Entropy Framework                                                    | 3 pts            |
| Soft Value Function and the Variational Formula                              | 5 pts            |
| Analysis on $\alpha$                                                         | 2 pts            |
| Soft Policy Evaluation Lemma                                                 | 3 pts            |
| Soft Bellman Residual Bound                                                  | 4 pts            |
| Soft Policy Improvement Lemma                                                | 3 pts            |
| Soft Policy Iteration Theorem                                                | 3 pts            |
| SAC Actor Objective as KL Minimization                                       | 3 pts            |
| REINFORCE Variance                                                           | 2 pts            |
| Reparameterized Gradient                                                     | 5 pts            |
| Squashed Gaussian Policy and Change of Variables                             | 3 pts            |
| Automatic Temperature Tuning                                                 | 3 pts            |
| <b>Asynchronous Advantage Actor-Critic (A3C)</b>                             | <b>28 pts</b>    |
| Baselines, Bias, and the Optimal Baseline                                    | 7 pts            |
| Multi-Step Advantage as a Sum of TD Errors                                   | 3 pts            |
| Critic-Induced Bias in Multi-Step Actor-Critic                               | 5 pts            |
| A2C Update Equations and Semi-Gradient Critic Learning                       | 3 pts            |
| Entropy as KL Regularization                                                 | 2 pts            |
| Stale Gradients in Asynchronous Actor-Critic                                 | 3 pts            |
| Decorrelation Through Parallel Workers                                       | 5 pts            |
| <b>Bonus</b>                                                                 | <b>+5 pts</b>    |
| Clarity and Quality of Report                                                | +5 pts           |
| <b>Total</b>                                                                 | <b>125+5 pts</b> |

# 1 Deterministic Actor-Critic: From DPG to TD3

This problem studies deterministic policy gradients, DDPG, and TD3, emphasizing derivations, off-policy actor-critic learning, function approximation error, overestimation bias, and algorithmic implications in continuous action spaces [2, 3, 4].

## 1.1 Background: Stochastic Policy Gradients

Recall the **Stochastic Policy Gradient (SPG) Theorem**:

### Stochastic Policy Gradient Theorem

For a stochastic policy  $\pi_\theta(a|s)$ , the gradient of the performance objective

$$J(\theta) = \mathbb{E}_{s \sim \rho^{\pi_\theta}} [v^{\pi_\theta}(s)]$$

is

$$\nabla_\theta J(\theta) = \mathbb{E}_{s \sim \rho^{\pi_\theta}, a \sim \pi_\theta} [\nabla_\theta \log \pi_\theta(a|s) Q^{\pi_\theta}(s, a)].$$

In continuous action spaces  $\mathcal{A} \subseteq \mathbb{R}^n$ , computing this gradient requires integrating over the action space:

$$\nabla_\theta J(\theta) = \int_{\mathcal{S}} \rho^{\pi_\theta}(s) \int_{\mathcal{A}} \nabla_\theta \pi_\theta(a|s) Q^{\pi_\theta}(s, a) da ds.$$

This integration can be computationally expensive or intractable in high-dimensional action spaces.

### Stochastic Actor-Critic Methods

To estimate the policy gradient, we use actor-critic methods that maintain:

- **Actor**: policy  $\pi_\theta(a|s)$ ;
- **Critic**: action-value approximation  $Q^w(s, a) \approx Q^{\pi_\theta}(s, a)$ .

### On-Policy Stochastic Actor-Critic:

Here  $a_t \sim \pi_\theta(\cdot|s_t)$  and  $a_{t+1} \sim \pi_\theta(\cdot|s_{t+1})$ .

$$\begin{aligned} \delta_t &= r_t + \gamma Q^w(s_{t+1}, a_{t+1}) - Q^w(s_t, a_t), \\ w_{t+1} &= w_t + \alpha_w \delta_t \nabla_w Q^w(s_t, a_t), \\ \theta_{t+1} &= \theta_t + \alpha_\theta \nabla_\theta \log \pi_\theta(a_t|s_t) Q^w(s_t, a_t). \end{aligned}$$

### Off-Policy Stochastic Actor-Critic:

A behavior policy  $\beta(a|s)$  generates transitions, and importance sampling corrections are used [7]. Define

$$\rho_t = \frac{\pi_\theta(a_t|s_t)}{\beta(a_t|s_t)}.$$

Here  $a_t \sim \beta(\cdot|s_t)$  and  $a_{t+1} \sim \beta(\cdot|s_{t+1})$  are sampled from the behavior policy. The critic update must also be corrected with the importance weight  $\rho_t$  to properly estimate  $Q^\pi$ :

$$\begin{aligned} \delta_t &= r_t + \gamma Q^w(s_{t+1}, a_{t+1}) - Q^w(s_t, a_t), \\ w_{t+1} &= w_t + \alpha_w \rho_t \delta_t \nabla_w Q^w(s_t, a_t), \\ \theta_{t+1} &= \theta_t + \alpha_\theta \rho_t \nabla_\theta \log \pi_\theta(a_t|s_t) Q^w(s_t, a_t). \end{aligned}$$

## 1.2 Action-Value Gradients

Consider a **deterministic policy**  $\mu_\theta : \mathcal{S} \rightarrow \mathcal{A}$ , which directly maps states to actions without stochasticity. For a given policy  $\mu^k$ , define the action-value function  $Q^{\mu^k}(s, a)$ .

A key insight is that we can compute the gradient of  $Q^{\mu^k}(s, \mu_\theta(s))$  with respect to  $\theta$  using the chain rule, avoiding the need to integrate over actions.

### Question 1: Action-Value Gradient Derivation

[1 Points]

Using the chain rule, derive the gradient of the action-value function evaluated at the deterministic policy:

$$\nabla_\theta Q^{\mu^k}(s, \mu_\theta(s))$$

Show that this equals:

$$\nabla_\theta \mu_\theta(s) \nabla_a Q^{\mu^k}(s, a) \big|_{a=\mu_\theta(s)}$$

## 1.3 Motivation for Deterministic Policy Gradients

The action-value gradient formula reveals a crucial advantage: we can compute policy gradients without integrating over the action space. Instead, we only need:

1. the policy Jacobian  $\nabla_\theta \mu_\theta(s)$  from the actor;
2. the action-value gradient  $\nabla_a Q^\mu(s, a) \big|_{a=\mu_\theta(s)}$  from the critic.

This suggests that for deterministic policies, we can derive a policy-gradient theorem that avoids the  $\int_{\mathcal{A}}$  term entirely. This is the essence of the Deterministic Policy Gradient theorem.

## 1.4 The Deterministic Policy Gradient Theorem

Let the deterministic policy be  $\mu_\theta : \mathcal{S} \rightarrow \mathcal{A}$ . The state-value and action-value functions are

$$V^{\mu_\theta}(s) = Q^{\mu_\theta}(s, \mu_\theta(s)),$$

and

$$Q^{\mu_\theta}(s, a) = r(s, a) + \gamma \int_{\mathcal{S}} p(s'|s, a) V^{\mu_\theta}(s') ds'.$$

Let the unnormalized discounted state-visitation measure be

$$\rho^{\mu_\theta}(s) = \sum_{t=0}^{\infty} \gamma^t \Pr(s_t = s \mid \mu_\theta).$$

### Deterministic Policy Gradient (DPG) Theorem

Consider a deterministic policy  $\mu_\theta : \mathcal{S} \rightarrow \mathcal{A}$  and the performance objective:

$$J(\mu_\theta) = \int_{\mathcal{S}} \rho^\mu(s) r(s, \mu_\theta(s)) ds = \mathbb{E}_{s \sim \rho^\mu} [r(s, \mu_\theta(s))]$$

where  $\rho^\mu(s)$  is the discounted state visitation distribution under policy  $\mu$ . Then:

$$\nabla_\theta J(\mu_\theta) = \int_S \rho^\mu(s) \nabla_\theta \mu_\theta(s) \nabla_a Q^\mu(s, a) \big|_{a=\mu_\theta(s)} ds$$

or equivalently:

$$\nabla_\theta J(\mu_\theta) = \mathbb{E}_{s \sim \rho^\mu} \left[ \nabla_\theta \mu_\theta(s) \nabla_a Q^\mu(s, a) \big|_{a=\mu_\theta(s)} \right]$$

### Question 2: Proof of the DPG Theorem

[8 Points]

Prove the Deterministic Policy Gradient Theorem by following these steps:

**Step 1** Express  $J(\mu_\theta)$  in terms of the state-value function  $V^\mu(s)$ . Show that:

$$J(\mu_\theta) = \int_S p_0(s_0) V^{\mu_\theta}(s_0) ds_0$$

where  $p_0(s_0)$  is the initial state distribution.

**Step 2** Using the Bellman equation for  $V^\mu$ , show that:

$$\nabla_\theta V^{\mu_\theta}(s) = \nabla_\theta Q^{\mu_\theta}(s, \mu_\theta(s))$$

**Step 3:** Express the gradient of the state-value function  $\nabla_\theta V^{\mu_\theta}(s)$  recursively by applying the chain rule to the Bellman equation  $V^{\mu_\theta}(s) = Q^{\mu_\theta}(s, \mu_\theta(s))$ , showing that:

$$\nabla_\theta V^{\mu_\theta}(s) = \nabla_\theta \mu_\theta(s)^\top \nabla_a Q^{\mu_\theta}(s, a) \big|_{a=\mu_\theta(s)} + \gamma \int_S p(s'|s, \mu_\theta(s)) \nabla_\theta V^{\mu_\theta}(s') ds'$$

**Step 4:** By repeatedly unrolling this recursive equation down the trajectory timeline, show how the discounted state visitation distribution  $\rho^{\mu_\theta}(s)$  naturally emerges to yield the final Deterministic Policy Gradient theorem formula.

## 1.5 Benefits of Deterministic Policy Gradients

The DPG theorem reveals several key advantages over stochastic policy gradients:

### 1. Computational Efficiency

The DPG gradient

$$\nabla_\theta J(\mu_\theta) = \mathbb{E}_{s \sim \rho^\mu} \left[ \nabla_\theta \mu_\theta(s) \nabla_a Q^\mu(s, a) \big|_{a=\mu_\theta(s)} \right]$$

requires only a **single action evaluation** per state, whereas the SPG gradient

$$\nabla_\theta J(\theta) = \mathbb{E}_{s \sim \rho^{\pi_\theta}, a \sim \pi_\theta} [\nabla_\theta \log \pi_\theta(a|s) Q^{\pi_\theta}(s, a)]$$

requires sampling actions from  $\pi_\theta(a|s)$  and estimating an integral over the action space.

In high-dimensional continuous action spaces, this difference is dramatic. For example, in a 20-dimensional action space, SPG may require many action samples per state to estimate the gradient

accurately, while DPG evaluates only the deterministic action.

## 2. Relationship to Stochastic Policy Gradients

Silver et al. show that DPG is the limiting case of SPG as the policy becomes increasingly deterministic. Specifically, consider

$$\pi_{\mu_\theta, \sigma}(a|s) = \mathcal{N}(a; \mu_\theta(s), \sigma^2 I).$$

As  $\sigma \rightarrow 0$ ,

$$\lim_{\sigma \rightarrow 0} \nabla_\theta J(\pi_{\mu_\theta, \sigma}) = \nabla_\theta J(\mu_\theta).$$

## 1.6 Deterministic Actor-Critic Algorithms

Just as with stochastic policies, we can derive actor-critic algorithms for deterministic policies by combining the DPG theorem with function approximation for the critic.

### Question 3: Deterministic On-Policy Actor-Critic

[2 Points]

Derive the **on-policy deterministic actor-critic** algorithm. Assume:

- The actor is a deterministic policy  $\mu_\theta(s)$
- The critic is a function approximator  $Q^w(s, a) \approx Q^\mu(s, a)$
- Transitions are generated by following  $\mu_\theta$  with exploration noise:  $a_t = \mu_\theta(s_t) + \mathcal{N}(0, \sigma^2)$

Provide the update rules for both the critic parameters  $w$  and the actor parameters  $\theta$ .

### Question 4: Deterministic Off-Policy Actor-Critic

[3 Points]

Derive the **off-policy deterministic actor-critic** algorithm. Assume:

- The target policy is  $\mu_\theta(s)$
- The behavior policy is  $\beta(a|s)$  (e.g.,  $\mu_\theta$  with added noise)
- Transitions  $(s_t, a_t, r_t, s_{t+1})$  are generated by  $\beta$

Show that, unlike the stochastic case, the off-policy deterministic actor-critic does **not** require importance sampling corrections in the actor update. Explain why.

### Implementation 1: Deterministic Policy Gradient (DPG)

[5 Points]

Implement a near-on-policy deterministic actor-critic baseline for the custom moving-target Reacher environment. The environment code and helper utilities are provided in `moving_target_reacher_env.py`.

The actor should be updated using the deterministic policy-gradient direction:

$$\nabla_\theta J(\mu_\theta) \approx \mathbb{E}_s \left[ \nabla_\theta \mu_\theta(s) \nabla_a Q_\phi(s, a) \Big|_{a=\mu_\theta(s)} \right].$$

The critic should be trained with a one-step bootstrapped target:

$$y_t = r_t + \gamma(1 - d_t)Q_\phi(s_{t+1}, \mu_\theta(s_{t+1})).$$

Since deterministic policies do not explore by themselves, collect data using noisy actions

$$a_t = \mu_\theta(s_t) + \epsilon_t.$$

Run the training pipeline, plot the learning curves, and briefly discuss why this baseline is less stable and less sample-efficient than DDPG and TD3.

## 1.7 Compatible Function Approximation

A critical question in actor-critic methods is: when does using a function approximator  $Q^w(s, a)$  instead of the true  $Q^\mu(s, a)$  introduce bias in the policy gradient?

The answer is given by the Compatible Function Approximation theorem, which provides conditions under which the approximate gradient is unbiased.

### Compatible Function Approximation Theorem for Deterministic Policies

A function approximator  $Q^w(s, a)$  is **compatible** with a deterministic policy  $\mu_\theta(s)$  if:

1.  $\nabla_a Q^w(s, a)|_{a=\mu_\theta(s)} = \nabla_\theta \mu_\theta(s)^\top w$
2. The parameter vector  $w$  of the approximate critic function  $Q^w(s, a)$  must minimize the mean-squared error of its action gradients relative to the true action-value function gradients:

$$\epsilon^2(\theta, w) = \mathbb{E}_{s \sim \rho^\mu} \left[ \left( \nabla_a Q^w(s, a)|_{a=\mu_\theta(s)} - \nabla_a Q^\mu(s, a)|_{a=\mu_\theta(s)} \right)^2 \right]$$

Then:

$$\mathbb{E}_{s \sim \rho^\mu} \left[ \nabla_\theta \mu_\theta(s) \nabla_a Q^w(s, a)|_{a=\mu_\theta(s)} \right] = \mathbb{E}_{s \sim \rho^\mu} \left[ \nabla_\theta \mu_\theta(s) \nabla_a Q^\mu(s, a)|_{a=\mu_\theta(s)} \right]$$

### Question 5: Proof of Compatible Function Approximation

[3 Points]

Prove the Compatible Function Approximation theorem by following these steps:

**Step 1** Using condition 1, show that:

$$\nabla_a Q^w(s, a)|_{a=\mu_\theta(s)} = \nabla_\theta \mu_\theta(s)^\top w$$

**Step 2** Show that minimizing  $\epsilon^2$  with respect to  $w$  yields the first-order condition:

$$\mathbb{E}_{s \sim \rho^\mu} [\nabla_\theta \mu_\theta(s) (Q^w(s, \mu_\theta(s)) - Q^\mu(s, \mu_\theta(s)))] = 0$$

**Step 3** Combine Steps 1 and 2 to prove:

$$\mathbb{E}_{s \sim \rho^\mu} \left[ \nabla_\theta \mu_\theta(s) \nabla_a Q^w(s, a)|_{a=\mu_\theta(s)} \right] = \mathbb{E}_{s \sim \rho^\mu} \left[ \nabla_\theta \mu_\theta(s) \nabla_a Q^\mu(s, a)|_{a=\mu_\theta(s)} \right]$$

## 1.8 From DPG to DDPG: Practical Challenges

While the DPG theorem provides a sound theoretical foundation, applying it to complex, high-dimensional problems with deep neural networks faces several challenges:

### 1. Sample Efficiency

On-policy methods like vanilla DPG require generating new experience after every policy update. This is sample-inefficient, especially when:

- interactions with the environment are expensive;
- the policy changes slowly;
- the state and action spaces are high-dimensional.

## 2. Stability with Deep Networks

Training deep neural networks with temporal-difference learning is unstable due to:

- **correlation in sequential data:** consecutive transitions are highly correlated;
- **moving targets:** the target  $r + \gamma Q^w(s', \mu_\theta(s'))$  changes as  $w$  and  $\theta$  update;
- **overestimation bias:** value-based backups may overestimate action values, and the actor can exploit these errors.

## 3. Exploration

Deterministic policies provide no inherent exploration mechanism. Without exploration, the agent may get stuck in local optima, fail to discover high-reward regions, and overfit to narrow regions of the replay buffer.

# 1.9 Deep Deterministic Policy Gradient (DDPG)

DDPG mitigates these challenges through four key techniques[?]:

### 1. Experience Replay Buffer

DDPG stores transitions  $(s_t, a_t, r_t, s_{t+1})$  in a replay buffer  $\mathcal{D}$  and samples random minibatches for training.

### 2. Target Networks

DDPG maintains target networks  $Q^{w'}$  and  $\mu_{\theta'}$  with parameters  $w'$  and  $\theta'$  updated by

$$w' \leftarrow \tau w + (1 - \tau)w', \quad \theta' \leftarrow \tau \theta + (1 - \tau)\theta',$$

where  $\tau \ll 1$ .

The TD target is computed using target networks:

$$y_t = r_t + \gamma Q^{w'}(s_{t+1}, \mu_{\theta'}(s_{t+1})).$$

### 3. Batch Normalization

DDPG applies batch normalization to state inputs and intermediate layers of both actor and critic networks.

### 4. Exploration Noise

DDPG adds temporally correlated noise to the deterministic policy during training:

$$a_t = \mu_\theta(s_t) + \mathcal{N}_t,$$

where  $\mathcal{N}_t$  is often an Ornstein-Uhlenbeck process.

## Question 6: DDPG Component Analysis

[3 Points]

For each DDPG component, explain how it addresses the practical challenges identified in Section 1.8:

- Experience Replay Buffer**
- Target Networks with Soft Updates**
- Batch Normalization**
- Ornstein-Uhlenbeck Exploration Noise**

For each component, also mention one limitation or possible failure mode.

## Implementation 2: Deep Deterministic Policy Gradient (DDPG)

[5 Points]

Extend your DPG implementation to the off-policy DDPG algorithm on the custom moving-target Reacher environment.

The critic should be trained using samples from the replay buffer and the target-network bootstrapped target:

$$y_t = r_t + \gamma(1 - d_t)Q_{\bar{\phi}}(s_{t+1}, \mu_{\bar{\theta}}(s_{t+1})),$$

where  $Q_{\bar{\phi}}$  and  $\mu_{\bar{\theta}}$  are slowly updated target networks.

The critic loss is

$$L_Q(\phi) = \mathbb{E}_{(s,a,r,s',d) \sim \mathcal{D}} [(Q_{\phi}(s, a) - y)^2].$$

The actor should be updated using the deterministic policy-gradient objective:

$$J(\theta) = \mathbb{E}_{s \sim \mathcal{D}} [Q_{\phi}(s, \mu_{\theta}(s))],$$

or equivalently by minimizing

$$L_{\pi}(\theta) = -\mathbb{E}_{s \sim \mathcal{D}} [Q_{\phi}(s, \mu_{\theta}(s))].$$

During data collection, use noisy actions

$$a_t = \mu_{\theta}(s_t) + \epsilon_t$$

to ensure exploration, but evaluate the final policy without exploration noise.

Run the training pipeline, plot the learning curves, and compare DDPG against your DPG baseline.

In your report, explain how replay buffers and target networks address the sample-efficiency and moving-target instability problems of the near-on-policy DPG baseline.

## 1.10 Function Approximation Error in Deterministic Actor-Critic

Consider a deterministic policy  $\mu_{\theta} : \mathcal{S} \rightarrow \mathcal{A}$  and a learned critic  $Q_{\phi}(s, a)$ . In standard deterministic actor-critic methods, the actor is updated by

$$\nabla_{\theta} J(\theta) \approx \mathbb{E}_{s \sim \rho^{\mu}} \left[ \nabla_{\theta} \mu_{\theta}(s) \nabla_a Q_{\phi}(s, a) \Big|_{a=\mu_{\theta}(s)} \right].$$

However, when  $Q_{\phi}$  has approximation error, the actor gradient can be biased.

## Question 7: Critic Gradient Error and Actor Bias

[2 Points]

Assume

$$\left\| \nabla_a Q_{\phi}(s, a) \Big|_{a=\mu_{\theta}(s)} - \nabla_a Q^{\mu}(s, a) \Big|_{a=\mu_{\theta}(s)} \right\|_2 \leq \epsilon$$

for every  $s$ , and

$$\|\nabla_{\theta} \mu_{\theta}(s)\|_{\text{op}} \leq M.$$

Prove that the actor-gradient error satisfies

$$\left\| \mathbb{E}_{s \sim \rho^{\mu}} \left[ \nabla_{\theta} \mu_{\theta}(s) (\nabla_a Q_{\phi}(s, a) - \nabla_a Q^{\mu}(s, a)) \Big|_{a=\mu_{\theta}(s)} \right] \right\|_2 \leq M\epsilon.$$

Explain why this makes deterministic actor-critic especially sensitive to critic errors.

## 1.11 TD3: Addressing Function Approximation Error

Twin Delayed Deep Deterministic Policy Gradient (TD3) builds on DDPG and introduces three mechanisms to reduce overestimation bias and improve stability:

1. clipped double Q-learning;
2. delayed policy updates;
3. target policy smoothing.

Consider a deterministic policy  $\mu_\theta : \mathcal{S} \rightarrow \mathcal{A}$  and a Q-function approximator  $Q_\phi(s, a)$ . In standard actor-critic methods, the policy is updated by:

$$\nabla_\theta J(\theta) = \mathbb{E}_{s \sim \rho^\mu} [\nabla_\theta \mu_\theta(s) \nabla_a Q_\phi(s, a)|_{a=\mu_\theta(s)}]$$

However, when  $Q_\phi$  overestimates the true Q-values, the policy gradient becomes biased, which can lead to suboptimal policies.

### Question 8: Overestimation and Policy Performance

[2 Points]

Explain intuitively why overestimation in the critic leads to poor policy performance. What happens when the actor exploits overestimated Q-values?

### Question 9: Sources of Overestimation Bias

[2 Points]

List and briefly explain three sources of overestimation bias in value-based reinforcement learning with function approximation.

## 1.12 Clipped Double Q-Learning

TD3 uses two critics and forms the target

$$y = r + \gamma \min_{i=1,2} Q_{\phi'_i}(s', \mu_{\theta'}(s')).$$

### Question 10: Minimum of Two Q-Estimates

[3 Points]

While standard Double Q-learning assumes independent critics to achieve an unbiased estimate, TD3 acknowledges that critics in actor-critic frameworks are not entirely independent. Explain why this lack of independence causes standard Double Q-learning to fail in continuous control, and show mathematically how taking the minimum  $y = r + \gamma \min_{i=1,2} Q_{\phi'_i}(s', a')$  forces an underestimation bias assuming the critics' errors can be modeled with symmetric noise.

### Question 11: Underestimation vs. Overestimation

[2 Points]

Explain the trade-off: why might a slight underestimation bias be less harmful than overestimation bias in actor-critic methods?

## 1.13 Delayed Policy Updates

In TD3, the policy is updated less frequently than the Q-functions. Specifically, the policy is updated once every  $d$  critic updates, typically with  $d = 2$ .

### Question 12: Delayed Policy Updates

[2 Points]

Explain why updating the policy too frequently relative to the critic can be harmful. Interpret delayed policy updates as an approximate two-time-scale method.

## 1.14 Target Policy Smoothing

TD3 adds noise to the target policy action when computing the target Q-value:

$$y = r + \gamma \min_{i=1,2} Q_{\phi'_i}(s', \mu_{\theta'}(s') + \epsilon),$$

where

$$\epsilon \sim \text{clip}(\mathcal{N}(0, \sigma), -c, c).$$

**Note:** The noise  $\epsilon$  is sampled exactly once per target computation and applied equally to the action evaluated by both critics.

### Question 13: Target Policy Smoothing Intuition

[2 Points]

Explain the intuition behind target policy smoothing. Why does adding clipped noise to the target action help?

### Question 14: Target Smoothing and Regularization

[3 Points]

Ignoring clipping, suppose  $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$  and  $Q(s, a)$  is twice differentiable in  $a$ . Derive a second-order approximation for

$$\mathbb{E}_\epsilon[Q(s', \mu(s') + \epsilon)].$$

Explain how this relates to local regularization of the critic target.

### Implementation 3: Twin Delayed Deep Deterministic Policy Gradient (TD3)

[10 Points]

Extend your DDPG implementation to TD3 on the custom moving-target Reacher environment. TD3 uses two critics  $Q_{\phi_1}$  and  $Q_{\phi_2}$  to reduce overestimation bias. The critic target should be

$$y_t = r_t + \gamma(1 - d_t) \min_{i=1,2} Q_{\phi_i}(s_{t+1}, \mu_{\bar{\theta}}(s_{t+1}) + \epsilon),$$

where

$$\epsilon \sim \text{clip}(\mathcal{N}(0, \sigma^2), -c, c).$$

The perturbed target action should also be clipped to the valid action range of the environment. The critic loss is

$$L_Q(\phi_1, \phi_2) = \mathbb{E}_{(s,a,r,s',d) \sim \mathcal{D}} [(Q_{\phi_1}(s, a) - y)^2 + (Q_{\phi_2}(s, a) - y)^2].$$

The actor should be updated less frequently than the critics. When an actor update is performed, use

$$L_{\pi}(\theta) = -\mathbb{E}_{s \sim \mathcal{D}} [Q_{\phi_1}(s, \mu_{\theta}(s))] .$$

Implement the three main TD3 modifications:

1. **Clipped double Q-learning:** use  $\min(Q_{\bar{\phi}_1}, Q_{\bar{\phi}_2})$  in the target.
2. **Delayed policy updates:** update the actor and target networks once every  $d$  critic updates.
3. **Target policy smoothing:** add clipped Gaussian noise to the target action.

Run the training pipeline and compare TD3 against DDPG and the near-on-policy DPG baseline. Include learning curves and at least one ablation, such as removing clipped double Q-learning, removing delayed actor updates, or removing target policy smoothing. In your report, explain which TD3 component had the largest effect and relate your observations to critic overestimation and actor exploitation of critic errors.

## 2 Soft Actor-Critic

### 2.1 Maximum Entropy Framework

Maximum-entropy RL augments return maximization with an entropy regularizer:

$$J(\pi) = \sum_{t=0}^T \mathbb{E}_{(s_t, a_t) \sim \rho_\pi} [r(s_t, a_t) + \alpha \mathcal{H}(\pi(\cdot|s_t))], \quad (1)$$

where  $\alpha > 0$  is the temperature parameter and

$$\mathcal{H}(\pi(\cdot|s_t)) = -\mathbb{E}_{a \sim \pi(\cdot|s_t)} [\log \pi(a|s_t)].$$

Equivalently, in discounted infinite-horizon form, the maximum-entropy objective can be written as

$$J(\pi) = \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t (r(s_t, a_t) - \alpha \log \pi(a_t|s_t)) \right].$$

For a fixed policy  $\pi$ , the soft value function is

$$V^\pi(s_t) = \mathbb{E}_{a_t \sim \pi(\cdot|s_t)} [Q^\pi(s_t, a_t) - \alpha \log \pi(a_t|s_t)]. \quad (2)$$

The soft state-action value satisfies the soft Bellman equation

$$Q^\pi(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s, a)} [V^\pi(s')].$$

#### Question 1: Maximum Entropy Framework

[3 Points]

Explain intuitively why adding entropy improves exploration and robustness.

Your answer should discuss:

1. why entropy prevents premature collapse to deterministic policies;
2. why stochasticity is useful in environments with multiple near-optimal actions;
3. why the maximum-entropy objective can be more robust to approximation error than pure reward maximization.

For a fixed function  $Q(s, a)$ , define the entropy-regularized value induced by optimizing over the action distribution:

$$\mathcal{V}_Q(s) = \sup_{\pi(\cdot|s)} \mathbb{E}_{a \sim \pi(\cdot|s)} [Q(s, a) - \alpha \log \pi(a|s)].$$

#### Question 2: Soft Value Function and the Variational Formula

[5 Points]

Show that the entropy-optimized soft value function can be written as

$$\mathcal{V}_Q(s) = \alpha \log \int_{\mathcal{A}} \exp \left( \frac{1}{\alpha} Q(s, a) \right) da. \quad (3)$$

Also show that the maximizing policy is the Boltzmann policy

$$\pi_Q^*(a|s) = \frac{\exp\left(\frac{1}{\alpha}Q(s, a)\right)}{\int_{\mathcal{A}} \exp\left(\frac{1}{\alpha}Q(s, \bar{a})\right) d\bar{a}}.$$

Finally, explain why Equation (3) is not generally equal to  $V^\pi(s)$  for an arbitrary fixed policy  $\pi$ .

### Question 3: Analysis on $\alpha$

[2 Points]

Analyze the limiting behavior as  $\alpha \rightarrow 0$  and  $\alpha \rightarrow \infty$ .

Your answer should discuss both the soft value

$$\alpha \log \int_{\mathcal{A}} \exp\left(\frac{1}{\alpha}Q(s, a)\right) da$$

and the Boltzmann policy  $\pi_Q^*(a|s)$ .

## 2.2 Soft Policy Evaluation

For a fixed policy  $\pi$ , define

$$V_Q^\pi(s) = \mathbb{E}_{a \sim \pi(\cdot|s)}[Q(s, a) - \alpha \log \pi(a|s)].$$

The soft Bellman backup operator is

$$\mathcal{T}^\pi Q(s, a) \triangleq r(s, a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s, a)}[V_Q^\pi(s')]. \quad (4)$$

### Soft Policy Evaluation Lemma

Consider the soft Bellman backup operator  $\mathcal{T}^\pi$  in Equation (4). Let  $Q^0 : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$  be arbitrary. Define  $Q^{k+1} = \mathcal{T}^\pi Q^k$ . Then  $Q^k$  converges to the soft  $Q$ -value  $Q^\pi$  as  $k \rightarrow \infty$ .

### Question 4: Soft Policy Evaluation Lemma

[3 Points]

Prove the Soft Policy Evaluation Lemma.

### Question 5: Soft Bellman Residual Bound

[4 Points]

Let  $Q^\pi$  be the fixed point of  $\mathcal{T}^\pi$ . Suppose an approximate critic  $Q$  satisfies

$$\|\mathcal{T}^\pi Q - Q\|_\infty \leq \epsilon.$$

Prove that

$$\|Q - Q^\pi\|_\infty \leq \frac{\epsilon}{1 - \gamma}.$$

## 2.3 Soft Policy Improvement

### Soft Policy Improvement Lemma

Let  $\pi_{old} \in \Pi$  and define:

$$\pi_{new} = \arg \min_{\pi' \in \Pi} D_{KL} \left( \pi'(\cdot|s_t) \parallel \frac{\exp \left( \frac{1}{\alpha} Q^{\pi_{old}}(s_t, \cdot) \right)}{Z^{\pi_{old}}(s_t)} \right). \quad (5)$$

Then  $Q^{\pi_{new}}(s_t, a_t) \geq Q^{\pi_{old}}(s_t, a_t)$  for all  $(s_t, a_t)$  with  $|\mathcal{A}| < \infty$ .

### Question 6: Soft Policy Improvement Lemma

[3 Points]

Prove the Soft Policy Improvement Lemma.

## 2.4 Soft Policy Iteration

Now that we have established both soft policy evaluation and soft policy improvement, we can combine them into a complete algorithm.

### Soft Policy Iteration Theorem

Repeated application of soft policy evaluation and soft policy improvement from any  $\pi \in \Pi$  converges to a policy  $\pi^*$  such that  $Q^{\pi^*}(s_t, a_t) \geq Q^\pi(s_t, a_t)$  for all  $\pi \in \Pi$  and  $(s_t, a_t) \in \mathcal{S} \times \mathcal{A}$ , assuming  $|\mathcal{A}| < \infty$ .

### Question 7: Soft Policy Iteration Theorem

[3 Points]

Prove the Soft Policy Iteration Theorem.

## 2.5 Soft Actor-Critic Losses

In this SAC variant, we use one soft Q-function  $Q_\phi$ , a target network  $Q_{\bar{\phi}}$ , and a stochastic actor  $\pi_\theta$ .

For a sampled transition  $(s, a, r, s') \sim \mathcal{D}$ , SAC uses the target

$$y = r + \gamma (Q_{\bar{\phi}}(s', a') - \alpha \log \pi_\theta(a'|s')), \quad a' \sim \pi_\theta(\cdot|s').$$

The critic loss is

$$J_Q(\phi) = \mathbb{E}_{(s,a,r,s') \sim \mathcal{D}} \left[ \frac{1}{2} (Q_\phi(s, a) - y)^2 \right].$$

The actor loss is

$$J_\pi(\theta) = \mathbb{E}_{s \sim \mathcal{D}, a \sim \pi_\theta(\cdot|s)} [\alpha \log \pi_\theta(a|s) - Q_\phi(s, a)].$$

**Question 8: SAC Actor Objective as KL Minimization****[3 Points]**

For a fixed critic  $Q_\phi$ , show that minimizing

$$J_\pi(\theta) = \mathbb{E}_{s \sim \mathcal{D}, a \sim \pi_\theta(\cdot|s)} [\alpha \log \pi_\theta(a|s) - Q_\phi(s, a)]$$

is equivalent, up to a state-dependent constant, to minimizing

$$\mathbb{E}_{s \sim \mathcal{D}} \left[ \text{D}_{\text{KL}} \left( \pi_\theta(\cdot|s) \parallel \frac{\exp \left( \frac{1}{\alpha} Q_\phi(s, \cdot) \right)}{Z_\phi(s)} \right) \right].$$

Explain the connection between this actor update and soft policy improvement.

**2.6 Reparameterized Gradient**

For continuous actions, we parameterize

$$a_t = f_\theta(\epsilon_t; s_t), \quad \epsilon_t \sim \mathcal{N}(0, I).$$

**Question 9: REINFORCE Variance****[2 Points]**

Explain why REINFORCE has high variance.

Your answer should compare score-function gradients with pathwise gradients and should mention why long-horizon returns make the variance problem worse.

**Question 10: Reparameterized Gradient****[5 Points]**

The SAC actor objective can be written as

$$J_\pi(\theta) = \mathbb{E}_{s_t \sim \mathcal{D}, \epsilon_t \sim \mathcal{N}} [\alpha \log \pi_\theta(a_t|s_t) - Q_\phi(s_t, a_t)], \quad a_t = f_\theta(\epsilon_t; s_t).$$

Derive the full reparameterized gradient and verify that the following expression is correct:

$$\nabla_\theta J_\pi(\theta) = \mathbb{E}_{s_t \sim \mathcal{D}, \epsilon_t \sim \mathcal{N}} [\nabla_\theta \alpha \log \pi_\theta(a_t|s_t) - \nabla_{a_t} Q_\phi(s_t, a_t) \nabla_\theta f_\theta] \quad (6)$$

**Question 11: Squashed Gaussian Policy and Change of Variables****[3 Points]**

In SAC, continuous actions are often sampled using a squashed Gaussian policy:

$$u_\theta(s, \epsilon) = \mu_\theta(s) + \sigma_\theta(s) \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I),$$

$$a = \tanh(u_\theta(s, \epsilon)).$$

Derive the log-probability correction

$$\log \pi_\theta(a|s) = \log \mathcal{N}(u; \mu_\theta(s), \sigma_\theta(s)^2) - \sum_{j=1}^d \log(1 - \tanh^2(u_j)),$$

where  $d$  is the action dimension.

## 2.7 Temperature Tuning

The temperature  $\alpha$  controls the trade-off between reward maximization and entropy maximization. Instead of manually tuning  $\alpha$ , SAC often learns it automatically using a target entropy  $\bar{\mathcal{H}}$ .

### Question 12: Automatic Temperature Tuning

[3 Points]

Consider the entropy constraint

$$\mathbb{E}_{s \sim \mathcal{D}, a \sim \pi_\theta(\cdot|s)}[-\log \pi_\theta(a|s)] \geq \bar{\mathcal{H}}.$$

Treating the policy  $\pi_\theta$  as fixed for the temperature update, derive the temperature loss

$$J(\alpha) = \mathbb{E}_{s \sim \mathcal{D}, a \sim \pi_\theta(\cdot|s)}[-\alpha(\log \pi_\theta(a|s) + \bar{\mathcal{H}})], \quad \alpha > 0,$$

which is minimized with respect to  $\alpha$ .

Compute  $\nabla_\alpha J(\alpha)$  and explain how gradient descent on  $J(\alpha)$  changes  $\alpha$  when the policy entropy is below or above the target entropy.

### 3 Asynchronous Advantage Actor-Critic (A3C)

Study A2C and its asynchronous extension A3C, focusing on variance reduction, multi-step bootstrapping, entropy regularization, stale-gradient effects, and parallel optimization behavior [6].

#### 3.1 Variance Reduction in Policy Gradients

Recall the standard REINFORCE policy gradient estimator:

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

where

$$R(\tau) = \sum_{t=0}^{T-1} \gamma^t r_t$$

is the total return of trajectory  $\tau$ .

A well-known variance reduction technique introduces a **baseline**  $b(s_t)$ :

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

The **advantage function** is defined as

$$A^{\pi}(s, a) = Q^{\pi}(s, a) - V^{\pi}(s),$$

where

$$V^{\pi}(s) = \mathbb{E}_{a \sim \pi(\cdot | s)} [Q^{\pi}(s, a)].$$

#### Question 1: Baselines, Bias, and the Optimal Baseline

[7 Points]

For a fixed state  $s$ , define the score vector

$$g_{\theta}(s, a) = \nabla_{\theta} \log \pi_{\theta}(a | s).$$

**Part (a)** Prove that subtracting any state-dependent baseline  $b(s)$  does not introduce bias into the policy-gradient estimator. Specifically, show that

$$\mathbb{E}_{a \sim \pi_{\theta}(\cdot | s)} [g_{\theta}(s, a) b(s)] = 0.$$

**Part (b)** Now consider the single-state gradient estimator

$$X_b = g_{\theta}(s, a) (Q^{\pi_{\theta}}(s, a) - b(s)).$$

Derive the scalar baseline  $b^*(s)$  that minimizes

$$\mathbb{E}_{a \sim \pi_{\theta}(\cdot | s)} [\|X_b\|_2^2].$$

**Part (c)** Is the optimal baseline  $b^*(s)$  always equal to  $V^{\pi_{\theta}}(s)$ ? If not, state a sufficient condition under which they are equal.

## 3.2 Multi-Step Advantage Estimation

A3C uses multi-step returns to trade off variance and bootstrapping bias. Consider the  $n$ -step return estimator

$$R_t^{(n)} = \sum_{k=0}^{n-1} \gamma^k r_{t+k} + \gamma^n V_\phi(s_{t+n}).$$

The corresponding  $n$ -step advantage estimator is

$$\hat{A}_t^{(n)} = R_t^{(n)} - V_\phi(s_t).$$

The one-step temporal-difference error is

$$\delta_t = r_t + \gamma V_\phi(s_{t+1}) - V_\phi(s_t).$$

### Question 2: Multi-Step Advantage as a Sum of TD Errors

[3 Points]

Show that the  $n$ -step advantage estimator can be written as

$$\hat{A}_t^{(n)} = R_t^{(n)} - V_\phi(s_t) = \sum_{k=0}^{n-1} \gamma^k \delta_{t+k}.$$

Then explain how increasing  $n$  affects the bias and variance of the advantage estimator.

### Question 3: Critic-Induced Bias in Multi-Step Actor-Critic

[5 Points]

Assume that the learned critic differs from the true value function by an approximation error

$$V_\phi(s) = V^\pi(s) + e(s).$$

For  $n \geq 1$ , derive an explicit expression for the bias

$$\mathbb{E} \left[ \hat{A}_t^{(n)} \mid s_t = s, a_t = a \right] - A^\pi(s, a).$$

Your final answer should be written in terms of  $e(s)$ , the one-step transition kernel  $P(\cdot|s, a)$ , and the policy-induced transition matrix  $P_\pi$ .

Then answer the following:

1. What does the bias become when  $n = 1$ ?
2. What happens to the bias as  $n \rightarrow \infty$ , assuming  $\gamma^n e(s_{t+n}) \rightarrow 0$ ?
3. Why does the term  $-e(s)$  alone not bias the policy gradient?
4. Why can the next-state error term bias the actor?

## 3.3 Synchronous Advantage Actor-Critic (A2C)

The **Advantage Actor-Critic (A2C)** algorithm maintains two networks:

- **Actor:** Policy network  $\pi_\theta(a|s)$  parameterized by  $\theta$ .
- **Critic:** Value network  $V_\phi(s)$  parameterized by  $\phi$ .

The critic minimizes the value-function loss

$$L_{\text{critic}}(\phi) = \mathbb{E} \left[ \frac{1}{2} \left( V_{\phi}(s_t) - \hat{R}_t \right)^2 \right],$$

where

$$\hat{R}_t = \sum_{k=0}^{n-1} \gamma^k r_{t+k} + \gamma^n V_{\phi}(s_{t+n})$$

is the  $n$ -step return target.

The actor maximizes the advantage-weighted policy objective

$$J_{\text{actor}}(\theta) = \mathbb{E} \left[ \log \pi_{\theta}(a_t | s_t) \hat{A}_t^{(n)} \right],$$

where

$$\hat{A}_t^{(n)} = \hat{R}_t - V_{\phi}(s_t).$$

#### Question 4: A2C Update Equations and Semi-Gradient Critic Learning

[3 Points]

**Part (a)** Derive the semi-gradient update for the critic:

$$\nabla_{\phi} L_{\text{critic}}(\phi).$$

In this derivation, treat the bootstrap target  $\hat{R}_t$  as a constant with respect to  $\phi$ .

**Part (b)** Derive the policy-gradient update for the actor:

$$\nabla_{\theta} J_{\text{actor}}(\theta).$$

In this derivation, treat the advantage estimate  $\hat{A}_t^{(n)}$  as a constant with respect to  $\theta$ .

**Part (c)** Explain why the critic update above is called a **semi-gradient** update. What term would appear if we fully differentiated through the bootstrap target?

### 3.4 Entropy Regularization

To encourage exploration, A2C and A3C often add an entropy bonus to the policy objective:

$$J_{\text{total}}(\theta) = J_{\text{actor}}(\theta) + \beta H(\pi_{\theta}(\cdot | s_t)),$$

where

$$H(\pi_{\theta}(\cdot | s)) = -\mathbb{E}_{a \sim \pi_{\theta}(\cdot | s)} [\log \pi_{\theta}(a | s)].$$

For a discrete action space, this can be written as

$$H(\pi_{\theta}(\cdot | s)) = -\sum_{a \in \mathcal{A}} \pi_{\theta}(a | s) \log \pi_{\theta}(a | s).$$

**Question 5: Entropy as KL Regularization****[2 Points]**

**Part (a)** Show that maximizing entropy is equivalent to minimizing the KL divergence from the policy to the uniform distribution over actions:

$$D_{\text{KL}}(\pi_{\theta}(\cdot|s) \parallel \text{Unif}(\mathcal{A})).$$

**Part (b)** For a discrete action space with  $|\mathcal{A}|$  actions, derive

$$\nabla_{\theta} H(\pi_{\theta}(\cdot|s)) = - \sum_{a \in \mathcal{A}} \nabla_{\theta} \pi_{\theta}(a|s) (\log \pi_{\theta}(a|s) + 1).$$

**Part (c)** Explain why entropy regularization is particularly important in A3C, where each worker collects on-policy data without an experience replay buffer.

**3.5 Asynchronous Advantage Actor-Critic (A3C)**

The key innovation of **A3C** is to train multiple actor-learners in parallel on different CPU threads, each interacting with its own copy of the environment. This asynchronous framework addresses two critical problems:

**1. Correlated Experience**

Sequential on-policy data is highly correlated, leading to unstable learning. Traditional solutions use experience replay, but replay is naturally off-policy.

**2. Wall-Clock Efficiency**

Multiple workers collect data in parallel, improving wall-clock training time on CPU-based systems.

A3C addresses these problems as follows:

- Run  $N$  parallel actor-learners, each with its own environment instance.
- Each worker maintains local copies of the global parameters, denoted by  $\theta'$  and  $\phi'$ .
- Periodically, each worker:
  1. collects up to  $t_{\max}$  steps of experience;
  2. computes gradients  $\nabla_{\theta'} J_{\text{actor}}$  and  $\nabla_{\phi'} L_{\text{critic}}$ ;
  3. asynchronously updates the global parameters  $\theta$  and  $\phi$  using Hogwild!-style updates;
  4. synchronizes its local parameters by setting  $\theta' \leftarrow \theta$  and  $\phi' \leftarrow \phi$ .

**Question 6: Stale Gradients in Asynchronous Actor-Critic****[3 Points]**

In A3C, a worker may compute a gradient using stale local parameters  $\theta_{t-\tau}$  and apply it to the current global parameter  $\theta_t$ .

Let

$$G(\theta) = \nabla_{\theta} J(\theta)$$

denote the exact policy-gradient field. Assume  $G$  is  $L$ -Lipschitz:

$$\|G(\theta) - G(\theta')\|_2 \leq L \|\theta - \theta'\|_2.$$

Assume each global parameter update satisfies

$$\|\theta_{k+1} - \theta_k\|_2 \leq \alpha C.$$

Prove that the stale-gradient error is bounded by

$$\|G(\theta_t) - G(\theta_{t-\tau})\|_2 \leq L\alpha C\tau.$$

Then explain what this bound implies about the effect of the learning rate  $\alpha$ , the delay  $\tau$ , and the number of parallel workers.

### Question 7: Decorrelation Through Parallel Workers

[5 Points]

Suppose each A3C worker produces a stochastic gradient estimate  $g_i$  satisfying

$$\mathbb{E}[g_i] = g, \quad \text{Var}(g_i) = \Sigma.$$

Assume that for  $i \neq j$ ,

$$\text{Cov}(g_i, g_j) = \rho \Sigma,$$

where  $\rho \in [0, 1]$  measures correlation between worker gradients.

Let

$$\bar{g} = \frac{1}{N} \sum_{i=1}^N g_i.$$

**Part (a)** Compute  $\text{Var}(\bar{g})$  in terms of  $N$ ,  $\rho$ , and  $\Sigma$ .

**Part (b)** Analyze the special cases  $\rho = 0$  and  $\rho = 1$ .

**Part (c)** Use this calculation to explain why running multiple parallel actors can stabilize on-policy learning without experience replay.

## References

- [1] Based on INF8250AE - Introduction to Reinforcement Learning. Fall 2025.
- [2] Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., & Riedmiller, M. (2014). *Deterministic Policy Gradient Algorithms*. Proceedings of the 31st International Conference on Machine Learning (ICML).
- [3] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., & Wierstra, D. (2016). *Continuous Control with Deep Reinforcement Learning*. Proceedings of the International Conference on Learning Representations (ICLR).
- [4] Fujimoto, S., van Hoof, H., & Meger, D. (2018). *Addressing Function Approximation Error in Actor-Critic Methods*. Proceedings of the 35th International Conference on Machine Learning (ICML).
- [5] Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). *Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor*. Proceedings of the 35th International Conference on Machine Learning (ICML).
- [6] Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., & Kavukcuoglu, K. (2016). *Asynchronous Methods for Deep Reinforcement Learning*. Proceedings of the 33rd International Conference on Machine Learning (ICML).
- [7] Thomas Degris, Martha White, & Richard S. Sutton (2012). *Off-Policy Actor-Critic*. Proceedings of the 29th International Conference on Machine Learning (ICML).