DA6400 : Reinforcement Learning (Jan-May 2026)

Programming Assignment 1

Deadline: March 5, 2026 11:59 PM

1 Instructions

1. Write complete Python code for this assignment. Include your `requirements.txt` and/or conda yaml file along with a README to reproduce your experiments. We will intimate the submission instructions in due course.
2. Submit a report containing the experimental results and answers to the questions below. The answers must be clear, concise, and to the point. A single line explanation is sufficient if it serves the purpose. Unnecessary verbose can be penalized. The report has a strict limit of 10 pages.
3. The maximum marks without the bonus question is 60. The marks distribution is provided below. 60M $\equiv$ 60 marks.
4. We expect one submission per group of 3 members.
5. **Please strictly follow the academic code of conduct. Plagiarism will be penalized**

2 Gridworld and Value Iteration

Design an MDP for a drone tasked with navigating a windy forest that has caught fire. The drone must pick up water from the lake and deliver it to the active fire zone while avoiding smoke fumes (that hinder movement and poison the water) and crashing into boulders.

For simplicity, the forest is in the form of a 5 x 5 grid with the lake at $(0, 0)$ and fire zone at $(4, 4)$. The smoke fumes are at $\{(1, 2), (3, 2)\}$ and the boulders are at $\{(2, 4), (3, 4)\}$. At any given instant, the drone either has water or does not have water. If it enters the lake empty handed, the water gets filled automatically and remains filled thereafter. Further, it can either move north, south, east, west, or hover. The drone is affected by uncertain forest wind. As a result, it moves in the intended direction with a chance of 70%, along with a 10% chance of the wind pushing the drone in either of the directions perpendicular to the intended direction and a 10% chance of staying at the same place (due to motor breaks). In the hazardous smoke-prone areas, the movement is different, where the chance of moving in the intended direction drops to 40% and the backdraft increases the chance of staying to 40%. If the drone attempts to go off the grid, it stays in the same cell. When hovering in any cell, it stays at the same place deterministically and is unaffected by the wind. All the non-terminal moves result in a per-step penalty of -1 and entering the hazardous regions causes an additional penalty of -10. Upon crashing into the boulder, a cost of -100 is incurred and the navigation terminates. Upon reaching the fire zone with water, the navigation is considered a success (and gets terminated) with a payoff of 100. Overall, the drone's objective is navigating to the fire zone carrying water from the lake.

1. Design the above problem in the form of an MDP, i.e., define the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$. What are the discrete state and action spaces? Compute the state transition probability and reward matrices and visualize their following subsets: Consider the cell $(3, 3)$. Show the transition and reward matrices for starting from $(3, 3)$ and landing in one of the eight neighboring cells + itself (total nine including the self-loop). Take $\gamma = 0.95$. **[3M]**

2. Write Python code for the above MDP and run value iteration to find the optimal value function and optimal policy. Note that the policy and value function will be two-phased – navigating to the lake and then to the fire zone (while avoiding all obstacles). Which phase the drone is in depends on the water level (empty/filled). Visualize the optimal value function and the corresponding policy for both phases[1]. Take the reference for visualization from Sutton and Barto, Pg. 77, Fig. 4.1. **[5M]**

3. Suppose we change the above MDP by setting $\gamma = 0.3$. **[9M]**

   (a) What are the resulting optimal value function and policy? Explain the difference (if any) in the solution due to changing the problem in the two MDPs. **[3M]**

   (b) What happens near the hazardous states when $\gamma = 0.3$? Can you explain the behavior? **[2M]**

   (c) Is there any case in the two MDPs where the drone prefers doing nothing (hovering) despite the per-step penalty? Explain either ways. **[2M]**

   (d) Are there any interesting states (other than the ones covered above) in which the drone behaves differently in the two MDPs? Explain either ways. **[2M]**

4. Suppose we change the MDP by penalizing the drone more for entering hazardous states (penalty $= -90$). Keep $\gamma = 0.95$. **[10M]**

   (a) What are the resulting optimal value function and policy? Explain the difference (if any) in the solution due to changing the problem in the two MDPs. Focus on the regions near hazardous cells. **[3M]**

   (b) Is there are scenario where hovering may be preferred? Explain either ways. **[2M]**

   (c) Is there are scenario when a longer path to the fire zone is preferable from a particular cell? Why or why not? **[2M]**

[1] Both phases together constitute the MDP.

(d) Given the above MDP with the -90 hazard penalty and $\gamma = 0.95$, suppose the wind becomes stronger. As a result, in all non-terminal states, the drone moves in the intended direction with a chance of 40%, in either of the perpendicular direction with a chance of 25%, and stays in same the cell rest of the times. Comment your thoughts on the solution to the resulting MDP. [**3M**]

3 TD-based Control in Acrobot

For this programming task, you should utilize the following Gymnasium environment for training and evaluating your policies. The associated link contains the environment description. Please use the exact version of the environment as specified:

Acrobot-v1: This system consists of two links connected linearly to form a chain, with one end of the chain fixed. The joint between the two links is actuated. The goal is to apply torques on the actuated joint to swing the free end of the linear chain above a given height (preset in the default implementation) while starting from the initial state of hanging downwards. Here, “height” refers to the vertical position (y-coordinate) of the tip of the second link relative to the fixed base (the first joint). Considering that both links have a length $= 1$, the minimum and maximum achievable heights are -2 and 2, respectively.

The observation space is continuous. Use binning (number of bins $= 10$) to discretize the space (what is the number of states?). Use $\gamma = 0.99$.

1. Write Python code for SARSA and Q-learning using $\epsilon-$greedy exploration in the above environment. [**3M**]

2. Plot return vs. timesteps/episodes curves comparing SARSA and Q-learning. [**10M**]

   (a) For both algorithms, tune the hyperparameters (stepsize and ϵ) and report the top three ones (resulting in highest returns). Is a constant ϵ sufficient for exploration? If not, implement an appropriate $\epsilon-$decay schedule ($\rightarrow 0$) to get a good policy. [**4M**]

   (b) Plot (with tuned hyperparameters) the mean performance along with confidence intervals over 10 random seeds/runs (sample plot shown below in Fig 1). A random seed corresponds to the initial random Q-values and the initial random start state. Explain the results along with comparing the two algorithms. [**6M**]

3. Run the two algorithms starting with $\epsilon = 1$, decaying ϵ till 0.1, and fixed thereafter. Compare (i) the online performance (performance while learning) and (ii) the performance of the policies after finishing learning (i.e., without any exploration). Explain the differences, if there exist any. [**5M**]

4. Intuitively, increasing the number of bins results in better state representation, enhanc-

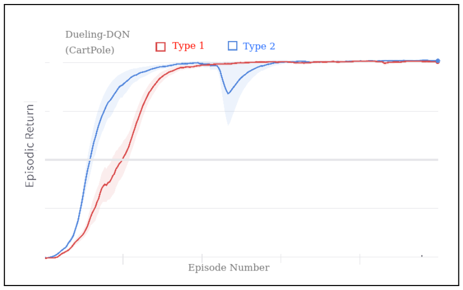

Figure 1: Sample plot for return vs. episode number of an algorithm in an environment.

ing granularity. This can result in learning a better policy. Is this always the case? Are there any downsides? Explain (try 5, 15, and 20 bins). **[3M]**

5. The classic Acrobot-v1 setup gives a per-step reward of -1 until reaching the specified height, which is when the episode terminates with a reward of 0. Suppose we modify this reward setting to a per-step reward as given by Eq. 1, where h is the height of the tip of the second link relative to the fixed base. Answer the following questions. It is not mandatory to run experiments; theoretical arguments are enough. **[12M]**

$$r = \frac{\eta h}{2} + \texttt{sign}(-1 + \eta h)\left(\frac{2 - \eta h}{2}\right), \ \eta > 0 \tag{1}$$

   (a) Will the modified reward setting result in faster/slower learning? Can the resulting behavior be something different/unexpected? Explain qualitatively. **[4M]**

   (b) Suppose $\eta = 0.5$. Explain the resulting behavior of the Acrobot. Will it learn faster/slower? **[2M]**

   (c) Suppose we increase η to $\{1, 2, 5\}$. Explain the resulting behavior. What do you recommend setting the range of η (and why)? **[3M]**

   (d) Can you give some insights on designing reward functions from this experiment? **[3M]**

6. **Bonus:** Support the arguments made in the previous question with reliable empirical evidence from either of the two algorithms. You may have to play around with the exploration rate and certain default environmental parameters of Acrobot-v1 hardcoded in its source code. **[10M]**