ECE-GY 9243 / ME-GY 7973
Optimal and learning control for robotics

Exercise series 1

For questions requesting a written answer, please provide a detailed explanation. Typesetted answers are required (e.g. using LaTeX[1]). Include plots where requested, either in a Jupyter Notebook or in the typesetted answers. For questions requesting a software implementation, please provide your code in a python file or in a Jupyter Notebook such that it can be run directly. Include comments explaining how the functions work and how the code should be run if necessary. Any piece of code that does not run out of the box or does not contain instructions to execute it will be considered invalid.

Exercise 1

a) Consider the following dynamical system

$$x_{n+1} = \begin{cases} -x_n + 1 + u_n & \text{if } -2 \leq -x_n + 1 + u_n \leq 2 \\ 2 & \text{if } -x_n + 1 + u_n > 2 \\ -2 & \text{else} \end{cases}$$

where $x_n \in \{-2, -1, 0, 1, 2\}$ and $u_n \in \{-1, 0, 1\}$, and the cost function

$$J = \left(\sum_{k=0}^{2} 2|x_k| + |u_k| \right) + x_3^2 \tag{1}$$

Use the dynamic programming algorithm to solve the finite horizon optimal control problem that minimizes J. Show the different steps of the algorithms and present the results in a table including the cost to go and the optimal control at every stage.

b) What is the sequence of control actions, states and the optimal cost if $x_0 = 0$, if $x_0 = -2$ and if $x_0 = 2$.

c) Assume now that the constant term 1 in the previous dynamics can sometimes be 0 with probability 0.3. We can now write the dynamics as

$$x_{n+1} = \begin{cases} -x_n + \omega_n + u_n & \text{if } -2 \leq -x_n + \omega_n + u_n \leq 2 \\ 2 & \text{if } -x_n + \omega_n + u_n > 2 \\ -2 & \text{else} \end{cases}$$

where $x_n \in \{-2, -1, 0, 1, 2\}$, $u_n \in \{-1, 0, 1\}$ and $\omega_n \in \{0, 1\}$ is a random variable with probability distribution $p(\omega_n = 0) = 0.3$, $p(\omega_n = 1) = 0.7$. The cost function to minimize becomes

$$J = \mathbb{E}\left(\left(\sum_{k=0}^{2} 2|x_k| + |u_k| \right) + x_3^2 \right) \tag{2}$$

[1]`https://en.wikibooks.org/wiki/LaTeX`, NYU provides access to Overleaf to all the community `https://www.overleaf.com/edu/nyu`

Use the dynamic programming algorithm to solve the finite horizon optimal control problem that minimizes J. Show the different steps of the algorithms and present the results in a table including the cost to go and the optimal control at every stage.

d) Compare the costs and optimal control from the deterministic and probabilistic models and explain why, in your opinion, the differences come from.

Exercise 2

Consider the following dynamical system

$$x_{n+1} = x_n + u_n$$

where x_n is an integer between -4 and 4 and $u_n \in \{-2, -1, 0, 1, 2\}$ but needs to be chosen such that $-4 \leq x_{n+1} \leq 4$ (e.g. when $x_n = 4$ then u_n can only be either 0, -1 or -2 but not 1 or 2).

We would like to minimize the following cost function, for $N = 15$

$$J = \sum_{k=0}^{N-1} ((x_k + 4)^2 + u_k^2) + g_N(x_N)$$

where

$$g_N(x_N) = \begin{cases} 0 \text{ if } x_N = 0 \\ \infty \text{ else} \end{cases}$$

a) Implement the dynamic programming algorithm in Python and solve the problem above.

b) What is the cost-to-go for $x_4 = 4$?

c) In two plots, show the optimal control and state sequences for $x_0 = 0$, $x_0 = 1$ and $x_0 = -4$ as a function of the stage number (use different colors for different initial conditions). What are the associated optimal costs?

d) Answer question c) when the terminal cost g_N is 0 for all states. How does the optimal cost change compared to the previous cost function? And the optimal control/state sequence? Can you explain why?

e) Answer question c) when the cost is

$$J = \sum_{k=0}^{N-1} (x_k + 4)^2 + g_N(x_N)$$

and the terminal cost g_N is 0 for all states. How does the optimal cost change compared to the previous cost function? And the optimal control/state sequence? Can you explain why?

Exercise 3

We want to find the shortest path for the robot in various mazes (the one shown in Figure 1 and three additional mazes described in the files *maze1.npy*, *maze2.npy* and *maze3.npy*. We will use the Jupyter file *exercise 3 - shortest paths.ipynb* and the helper functions provided to implement various algorithms. In each of these questions, the starting node is the top-left one and the goal node is the bottom-right one and it is only possible to move left, right, up or down (not diagonally).

a) Implement a depth-first algorithm to find the shortest path algorithm in the 4 mazes. How many nodes are tested (in CURRENT) until the shortest paths are found? What is the length of these shortest paths? (Display the shortest paths in the Jupyter Notebook)

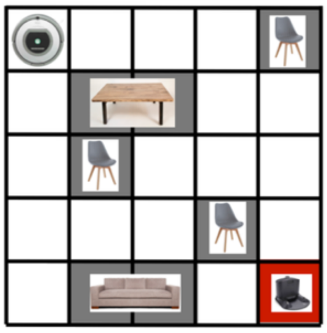

Figure 1: *Maze 0*

b) Implement a breadth-first algorithm to solve these problems and answer the same questions as in a).

c) Implement the A* algorithm and answer the same questions as a). Use as heuristic $h_{i,j} = 0$ (where i and j are the i row and j column entry) in one case and $h_{i,j} = |i - \text{goal}_i| + |j - \text{goal}_j|$ where goal_i and goal_j are the coordinates of the goal position (i.e. the sum of the horizontal distance to the goal and the vertical distance to the goal). Are these heuristics under-estimators of the cost-to-go? Why?

d) Given those results, what seem to be the pros and cons of the algorithms?