Control for Robotics: From Optimal Control to Reinforcement Learning

Assignment 1 Optimal Control and Dynamic Programming

General Information

Due date:	You can find the due date in the syllabus handout. Your solution must be submitted before 23h59 on the due date.
Submission:	Please submit your solution and the requested Matlab scripts (highlighted in `blue`) as a single PDF document. Both typed and scanned handwritten solutions are accepted. It is your responsibility to provide enough detail such that we can follow your approach and judge your solution. Students may discuss assignments. However, each student must code up and write up their solutions independently. We will check for plagiarism. The points for each question are shown in the left margin.

Introduction

This assignment provides a set of exercises providing a deeper understanding of the core ideas of optimal control and illustrating how optimal control and dynamics programming algorithms can be applied to robotics problems. The assignment is comprised of four marked problems (two of them require writing code) and one additional practice problem.

Problem 1.1 Finite Horizon Dynamic Programming

In this problem, we will apply the discrete-time dynamic programming algorithm to a simple velocity-controlled 1D mobile robot moving on a line as seen in Figure 1. The dynamics of the robot can be represented as

$$x_{k+1} = x_k + u_k + w_k, \tag{1}$$

where k is the discrete-time index, $x_k \in \mathbb{R}$ is the position of the robot, $u_k \in \mathbb{R}$ is the normalized velocity input (i.e., the distance driven in one time step), and $w_k \in \mathbb{R}$ is a disturbance (e.g., caused by variations in the terrain). The initial position of the robot is $x_0 = -1$.

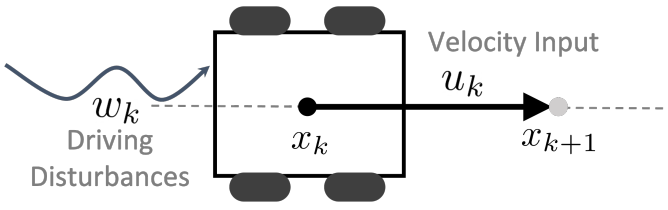

Figure 1: Velocity-controlled, 1D mobile robot.

We consider a time horizon of 2 and our aim is to minimize the cost function given by

$$\mathbb{E}_{w_k}\left[x_2^2 + \sum_{k=0}^{1}(qx_k^2 + ru_k^2)\right], \tag{2}$$

where $q > 0$ and $r > 0$ are constant scalars. No actuation constraints are imposed on u_k.

3 (a) Given the cost function and initial position, how do you expect the policy found using the dynamic programming algorithm to behave? How do you expect different values of q and r to change the robot's behaviour?

8 (b) Let $q = 5/2$ and $r = 1$ and assume that $w_k = 0$ for all k. Find the optimal policy and $J(x_0)$.

11 (c) We keep $q = 5/2$ and $r = 1$, but now assume there is some variation in the terrain causing the disturbance to follow a distribution with mean $\mathbb{E}[w_k] = 1$ and variance $\mathrm{Var}[w_k] = 1$ for $k = \{0, 1\}$. The disturbance could represent rocks, hills, and slippable terrain along the path that either help or hinder the robot's motion. Find the optimal policy and $J(x_0)$. *Hint: Recall that* $\mathrm{Var}[y] = \mathbb{E}[y^2] - \mathbb{E}[y]^2$.

3 (d) Comment on any differences in the solutions you obtained for parts (b) and (c).

Problem 1.2 Dynamic Programming for a Robot Vacuum Cleaner

Consider a robot vacuum cleaner that aims to reach its charging station while removing as much dirt as possible on the way. Additionally, the robot must avoid obstacles. This problem can be modeled as a grid world as shown in Figure 2. The grid world consists of 20 grid cells with varying costs: empty cells ($g_{\text{empty}} = 6$), dirty cells ($g_{\text{dirt}} = 1$), obstacles ($g_{\text{obstacle}} = \infty$), the charger ($g_{\text{charger}} = 0$), and cells with carpet ($g_{\text{carpet}} = 100$).

The state of the robot is its location in the grid with $x = (i, j)$ where $i \in \{1, ..., 4\}$ and $j \in \{1, ..., 5\}$. An action at time k moves the robot to the next grid cell. In each time step, the robot can go north ($u = 1$; in the code), east ($u = 2$), south ($u = 3$), or west ($u = 4$), as long as this action keeps the robot inside the grid world and does not hit an obstacle. At the charging station, the additional action of charging ($u = 0$) is available, where the robot does not move. The cost associated with this move depends on the grid it moves onto (empty, dirty, etc.). There is no terminal cost. The goal is to find the optimal policy for the robot vacuum cleaner to navigate back to the charging station.

Use the provided files to implement your solution and submit `cfr_a1_2.m` in PDF format.

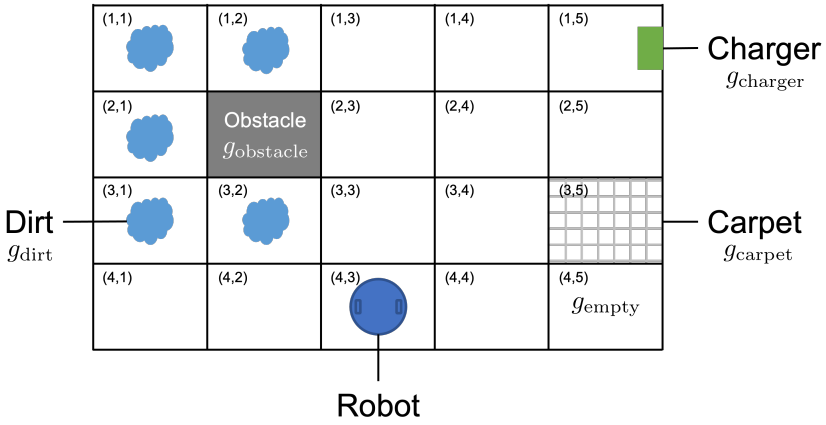

Figure 2: Illustration of the robot vacuum cleaner grid world.

14 (a) Familiarize yourself with the implementation of the class `GridWorld` in `GridWorld.m`, which models the robot vacuum cleaner problem. Take advantage of the provided methods of the class `GridWorld`

to implement the dynamic programming algorithm. Report the cost-to-go and optimal policies for the grid world.

3 (b) Provide the sequence of optimal control actions $u_0, ..., u_{N-1}$ from the intial state $x_0 = (4, 3)$. Qualitatively describe the optimal policy of the robot.

3 (c) Assume the same initial state as in part (b). Report the sequence of optimal control actions $u_0, ..., u_{N-1}$ when the cost for cleaning a dirty cell is $g_{\text{dirt}} = 5$? Qualitatively describe the optimal policy of the robot.

5 (d) Now assume that the robot is initially at the charging station. We aim to adapt the cost parameters such that the robot leaves the charging station, visits each dirty cell once, and then returns to the charging station? Describe qualitatively how you would adapt the cost parameters of the grid cells to achieve the task.

Problem 1.3 Approximate Dynamic Programming

In this problem, we use dynamic programming to stabilize an inverted pendulum at the upright position. Specifically, we will use approximate dynamic progamming over a gridded state space. Consider the inverted pendulum given by the following continuous-time nonlinear equations and shown in Figure 3:

$$\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{g}{l}\sin(x_1) + \frac{1}{ml^2}\, u \end{bmatrix},$$

where the state is the pendulum's angle $x_1 = \theta$ and its angular velocity $x_2 = \dot{\theta}$, u is the input torque, $m = 1$ kg and $l = 1$ m are the pendulum's mass and length, respectively, and $g = 9.81 \frac{\text{m}}{\text{s}^2}$ is the gravitational constant.

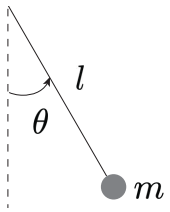

Figure 3: Illustration of the inverted pendulum problem.

The discrete-time controller shall minimize a quadratic objective function of the form

$$J(x_{\text{lin},0}) = x_{\text{lin},N}^T Q x_{\text{lin},N} + \sum_{k=0}^{N-1} x_{\text{lin},k}^T Q x_{\text{lin},k} + r u_{\text{lin},k}^2 \,,$$

where $x_{\text{lin},0}$ is the initial condition, Q is a positive definite weighting matrix, and $r > 0$ is a weighting scalar.

Use the provided files for your solution and submit `cfr_a1_3.m` in PDF format.

4 (a) Linearize the nonlinear continuous-time control system around the upright position $x_{\text{up}} = [\pi \; 0]^T$ with no control input. Find the linear system and input matrices A_c and B_c, respectively, for the linear system $\dot{x}_{\text{lin}} = A_c x_{\text{lin}} + B_c u_{\text{lin}}$ and add them to the Matlab file. Discretize the continuous-time control system using Matlab's `c2d` function and a sampling time $\Delta t = 0.1$. Report the discrete-time system and input matrices A_d and B_d, respectively, for the linear discrete-time system $x_{\text{lin},k+1} = A_d x_{\text{lin},k} + B_d u_{\text{lin},k}$, where $x_{\text{lin},k} = x_{\text{lin}}(k\Delta t)\,, k \in \mathbb{N}_{\geq 0}$. Also add the discrete-time matrices to the Matlab file.

3 (b) Add expressions for the stage cost and the initial cost-to-go to the Matlab file. *Hint: Use Matlab's anonymous functions to implement the costs, e.g.:* `stage_cost = @(x, u) <Fill in here>`.

12 (c) Implement the dynamic programming algorithm to recursively calculate the cost-to-go and the optimal control for the derived discrete-time linear system over a horizon of $N = 25$, the grid provided in the Matlab file, and $Q = \text{diag}(1, 0.1)$ and $r = 1$. Note that the intermediate cost-to-go $J_k(x_{\text{lin},k})$ might need to be evaluated at states that do not correspond to the grid values. Therefore, use a spline interpolation of the cost-to-go from the previous time step to approximately evaluate the cost-to-go. Run the script to control the system from the initial state $x_0 = x_{\text{up}} + \begin{bmatrix} -\frac{\pi}{6} & 0 \end{bmatrix}^T$. Does the controller stabilize the system at the upright position? Attach all the plots resulting from running your implemented controller. *Hint: Use the Matlab functions* `fminunc` *to solve the unconstrained nonlinear optimization problem at every time step and* `interp2` *to interpolate values of a function of two variables.*

3 (d) Vary the the values of Q and r. How do they affect the solution?

3 (e) How does this approach scale to higher-dimensional systems (i.e., states of dimension much larger than 2) with regards to computation time? Justify your proposition.

Problem 1.4 Infinite Horizon Dynamic Programming

In this problem, we will consider an infinite horizon optimal control problem. We again consider a mobile robot system similar to the first question:

$$\begin{aligned} x_{k+1} &= x_k + u_k + w_k \\ y_k &= x_k + v_k, \end{aligned}$$

where x_k is the true position of the mobile robot, $y_k \in \mathbb{R}$ is the measured position, corrupted by measurement noise v_k, and w_k represents disturbances in the terrain. We assume that the measurement noise v_k and the disturbances w_k are independent random variables with zero mean (i.e., $\mathbb{E}[v_k] = 0$ and $\mathbb{E}[w_k] = 0$) and unit variance (i.e., $\mathbb{E}[v_k^2] = 1$ and $\mathbb{E}[w_k^2] = 1$). Our goal is to minimize the following cost:

$$J = \lim_{N\to\infty} \frac{1}{N} \sum_{k=0}^{N-1} \mathbb{E}_{w_k, v_k} \left[\frac{x_k}{2}^2 + u_k^2 \right].$$

For this problem, we are given an output feedback controller $u_k = -\frac{1}{2} y_k$ that tries to move the robot's position to $x = 0$.

3 (a) Write out the closed-loop dynamics (i.e, write x_{k+1} as a function of x_k, v_k, and w_k).

3 (b) Show that this closed-loop system is stable in expectation. *Hint: Note that a discrete-time system is stable if the eigenvalues of the dynamics lie inside of the unit circle. That is, you should see that the closed-loop dynamics in expectation have the form $x_{k+1} = r x_k$. The eigenvalue of this 1D discrete-time system is r, and if $|r| < 1$, then the system is stable. Why might this be?*

10 (c) What is the infinite horizon cost J under the given feedback policy?

9 (d) Find the optimal closed-loop feedback gain α for the control law $u_k = \alpha y_k$. You can assume there is no noise for this problem. *Hint: you have to solve the discrete-time Algebraic Riccatti equation.*

Additional Practice Problem

Problem 1.5 Continuous-Time Optimal Control (Pontryagin Minimum Principle)

Not covered in SS24

Consider a quadrotor moving up and down along the vertical z-axis. Assume that this simplified 1D motion can be expressed in the following form:

$$\ddot{z}(t) = u(t), \tag{3}$$

where z is the position of the quadrotor in meters, and $u \in [-u_{\max}, u_{\max}]$ with $u_{\max} > 0$ is a scaled thrust input. The quadrotor is required to move from a given initial state to a given terminal state:

$$x(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \text{ and } x(T) = \begin{bmatrix} 2 \\ 0 \end{bmatrix}, \tag{4}$$

where $x(t) = [z(t), \dot{z}(t)]^T$ and T is the time horizon of the problem. Our goal is to design a control law that brings the quadrotor from the initial state to the terminal state with minimal total control effort:

$$\int_0^T |u(t)|\, dt. \tag{5}$$

An illustration of the problem is shown in Figure 4 (left).

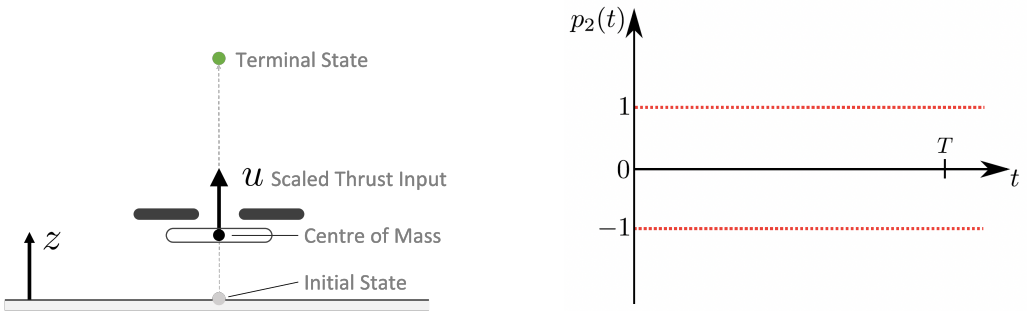

Figure 4: *Left:* Illustration of the 1D quadrotor problem. *Right:* Plot for part (d) of this problem.

3 (a) Express the dynamics of the system in the form of $\dot{x}(t) = f(x(t), u(t))$.

3 (b) Let $p = [p_1, p_2]^T$ be the co-state. Write down the Hamiltonian $H(x(t), u(t), p(t))$ of the problem.

3 (c) Write down the adjoint equation $\dot{p}(t) = -\nabla_x H(x^*, u^*, p(t))$ of this problem.

8 (d) Draw a possible solution of $p_2(t)$ for all $t \in [0, T]$ in a plot as provided in Figure 4 (right). You do not need to label the intersections of $p_2(t)$ with the horizontal and vertical axes. *Hint: Approach the problem by first writing a general solution for $p(t)$ and then writing down the $u(t)$ satisfying the Pontryagin minimum principle. You should be able to obtain three cases corresponding to $p_2(t) > 1$, $-1 \leq p_2(t) \leq 1$, and $p_2(t) < -1$. Further note that, for our problem, the quadrotor undergoes three phases: (i) accelerate from the initial state with maximum acceleration for $t \in [0, t_1]$, (ii) maintain zero acceleration for $t \in [t_1, t_2]$, and (iii) decelerate to reach the terminal position with zero velocity for $t \in [t_2, T]$, where $t_1 \leq t_2$.*

8 (e) Assume $u_{\max} = 2$ and $T = 4$. Derive the optimal control input $u^*(t)$ for all $t \in [0, T]$. *Hint: Find the optimal times t_1 and t_2 at which the quadrotor changes from phase (i) to phase (ii) and from phase (ii) to phase (iii), respectively, by solving for $x(t)$. Note that $x(t)$ should be continuous at t_1 and t_2, and satisfy the conditions in (4).*