Reinforcement Learning and Optimal Control for Robotics ROB-GY 6323 Exercise Series 3

November 4, 2024

Shantanu Ghodgaonkar
Univ ID: N11344563
Net ID: sng8399

Links to Jupyter Notebooks -

- Jupyter Notebook for code solution of Exercise 1
- Jupyter Notebook for code solution of Exercise 2

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.

Answer The system model and total cost have been defined as above. We can extract the terminal cost to be,

$$J_3 = x_3^2$$

And the cost function as,

$$g(x_k, u_k) = 2|x_k| + |u_k|$$

Now, we shall compute at the stage 2 for the state $x_0 = -2$ -

For, $x_0 = -2$,
With $u_0 = -1$ as $-x_0 + 1 + u_0 = -(-2) + 1 - 1 = 2 \leq 2 \Rightarrow f(x_0, u_0) = 2$
$\Rightarrow J_2(x_0) = g(x_0, u_0) + J_3(f(x_0, u_0)) = 2 * |-2| + |-1| + (2)^2 = 9$
With $u_1 = 0$ as $-x_0 + 1 + u_1 = -(-2) + 1 + 0 = 3 > 2 \Rightarrow f(x_0, u_1) = 2$
$\Rightarrow J_2(x_0) = g(x_0, u_1) + J_3(f(x_0, u_1)) = 2 * |-2| + |0| + (2)^2 = 8 \leftarrow$ **Minimum**
With $u_2 = 1$ as $-x_0 + 1 + u_0 = -(-2) + 1 + 1 = 4 > 2 \Rightarrow f(x_0, u_2) = 2$
$\Rightarrow J_2(x_0) = g(x_0, u_2) + J_3(f(x_0, u_2)) = 2 * |-2| + |1| + (2)^2 = 9$

Thus, we have computed the optimal control $u_0 = 0$ that results in minimum cost for the state x_0 at stage 2, $J_2(x_0) = 8$. By applying this same logic to all the states for all the stages,

$$J_2(x_1) = \min_{u_1} g(x_1, u_1) + J_3(f(x_1, u_1)) = 4 \quad \text{with } u_1 = -1$$
$$J_2(x_2) = \min_{u_2} g(x_2, u_2) + J_3(f(x_2, u_2)) = 1 \quad \text{with } u_2 = -1$$
$$J_2(x_3) = \min_{u_3} g(x_3, u_3) + J_3(f(x_3, u_3)) = 2 \quad \text{with } u_3 = 0$$
$$J_2(x_4) = \min_{u_4} g(x_4, u_4) + J_3(f(x_4, u_4)) = 5 \quad \text{with } u_4 = 0$$

Similarly, we can compute for all the stages, backward in time and recursively, using the generalized Dynamic Programming formula,

$$J_N(x_N) = g_N(x_N) \tag{1}$$
$$J_k(x_k) = \min_{u_k} \left(g_k(x_k, u_k) + J_{k+1}(f(x_k, u_k)) \right) \tag{2}$$

Using this idea, code has been written in the Jupyter Notebook for code solution of Exercise 1 and provides the following results -

State	J_0^*	u_0^*	J_1^*	u_1^*	J_2^*	u_2^*	J_3^*
-2	10	0	9	0	8	0	4
-1	6	-1	5	-1	4	-1	1
0	3	-1	2	-1	1	-1	0
1	4	0	3	0	2	0	1
2	7	1	6	1	5	0	4

Exercise: 1 (b) What is the sequence of control actions, states and the optimal cost if $x_0 = 0$, if $x_0 = -2$ and if $x_0 = 2$.

Answer To find the sequence of optimal control actions, we shall simply -

1. Find the optimal control for current state
2. Apply the optimal control to current state and obtain next state
3. Repeat the above steps until we reach the end

For the first case with $x_0 = 0$ -

1. For $x_0 = 0$, $u_0 = -1 \Rightarrow x_1 = f(x_0, u_0) = 0$ with $J_0 = 3$
2. For $x_1 = 0$, $u_1 = -1 \Rightarrow x_2 = f(x_1, u_1) = 0$ with $J_1 = 2$
3. For $x_2 = 0$, $u_2 = -1 \Rightarrow x_3 = f(x_2, u_2) = 0$ with $J_2 = 1$
4. For $x_3 = 0$, $J = 0$

Thus, we reached the end and minimized the cost to 0.

For the second case with $x_0 = -2$ -

1. For $x_0 = -2$, $u_0 = 0 \Rightarrow x_1 = f(x_0, u_0) = 2$ with $J_0 = 10$
2. For $x_1 = 2$, $u_1 = 1 \Rightarrow x_2 = f(x_1, u_1) = 0$ with $J_1 = 6$
3. For $x_2 = 0$, $u_2 = -1 \Rightarrow x_3 = f(x_2, u_2) = 0$ with $J_2 = 1$
4. For $x_3 = 0$, $J_3 = 0$

Thus, we reached the end and minimized the cost to 0.

For the third case with $x_0 = 2$ -

1. For $x_0 = 2$, $u_0 = 1 \Rightarrow x_1 = f(x_0, u_0) = 0$ with $J_0 = 7$
2. For $x_1 = 0$, $u_1 = -1 \Rightarrow x_2 = f(x_1, u_1) = 0$ with $J_1 = 2$
3. For $x_2 = 0$, $u_2 = -1 \Rightarrow x_3 = f(x_2, u_2) = 0$ with $J_2 = 1$
4. For $x_3 = 0$, $J_3 = 0$

Thus, we reached the end and minimized the cost to 0.

Exercise: 1 (c) Assume now that the constant term 1 in the previous dynamics can sometimes be 0 with probability 0.4. 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 $p(\omega_n = 0) = 0.4, p(\omega_n = 1) = 0.6$. The cost function to minimize becomes

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

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.

Answer The system model and total cost functions will be as given above. Firstly, let us write the cost function J_n for a state x_n at a stage n -

$$\begin{aligned} J_n(x_n) &= \mathbb{E}\left(g(x_n, u_n) + J_{n+1}(f(x_n, u_n))\right) \\ &= g(x_n, u_n) + \mathbb{E}\left(J_{n+1}(f(x_n, u_n))\right) \\ &= g(x_n, u_n) + \sum_{k=0}^{1} \left(p(\omega_k) J_{n+1}(f(x_n, u_n, \omega_k))\right) \end{aligned}$$

The above formulation works because the expectation only applies to non-deterministic terms. Using this function, we shall now compute at the stage 2 for the state $x_0 = -2$ -

$$\begin{aligned} \text{For } x_0 &= -2, \\ \text{With } u_0 &= -1 \quad \Rightarrow \quad J_2(x_0) = g(x_0, u_0) + p(0)J_3(f(x_0, u_0, 0)) + p(1)J_3(f(x_0, u_0, 1)) \\ &\qquad = 5 + 0.4(J_3(1)) + 0.6(J_3(2)) = 5 + 0.4(1) + 0.6(4) = 7.8 \leftarrow \textbf{Minimum} \\ \text{With } u_1 &= 0 \quad \Rightarrow \quad J_2(x_0) = g(x_0, u_1) + p(0)J_3(f(x_0, u_1, 0)) + p(1)J_3(f(x_0, u_1, 1)) \\ &\qquad = 4 + 0.4 \times J_3(2) + 0.6 \times J_3(2) = 4 + 0.4 \times 4 + 0.6 \times 4 = 9 \\ \text{With } u_2 &= 1 \quad \Rightarrow \quad J_2(x_0) = g(x_0, u_2) + p(0)J_3(f(x_0, u_2, 0)) + p(1)J_3(f(x_0, u_2, 1)) \\ &\qquad = 5 + 0.4 \times J_3(2) + 0.6 \times J_3(2) = 5 + 0.4 \times 4 + 0.6 \times 4 = 9 \end{aligned}$$

Thus, we have computed the optimal control $u_0 = -1$ that results in minimum cost for the state x_0 at stage 2, $J_2(x_0) = 7.8$. By applying this same logic to all the states for all the stages,

$$\begin{aligned} J_2(x_1) &= \min_{u_1} g(x_1, u_1) + \mathbb{E}\left(p(\omega_1)J_3(f(x_1, u_1, \omega_1))\right) = 3.6 \quad \text{with } u_1 = -1 \\ J_2(x_2) &= \min_{u_2} g(x_2, u_2) + \mathbb{E}\left(p(\omega_2)J_3(f(x_2, u_2, \omega_2))\right) = 0.6 \quad \text{with } u_2 = 0 \\ J_2(x_3) &= \min_{u_3} g(x_3, u_3) + \mathbb{E}\left(p(\omega_3)J_3(f(x_3, u_3, \omega_3))\right) = 2.4 \quad \text{with } u_3 = 0 \\ J_2(x_4) &= \min_{u_4} g(x_4, u_4) + \mathbb{E}\left(p(\omega_4)J_3(f(x_4, u_4, \omega_4))\right) = 5.4 \quad \text{with } u_4 = 1 \end{aligned}$$

Similarly, we can compute for all the stages, backward in time and recursively, using the generalized Dynamic Programming formula,

$$J_N(x_N) = g_N(x_N) \tag{3}$$

$$J_k(x_k) = \min_{u_k} \mathop{\mathbb{E}}_{\omega_k} \left(g_k(x_k, u_k) + J_{k+1}(f(x_k, u_k))\right) \tag{4}$$

Using this idea, code has been written in the Jupyter Notebook for code solution of Exercise 1 and provides the following results -

State	J_0^*	u_0^*	J_1^*	u_1^*	J_2^*	u_2^*	J_3^*
-2	9.8	-1	8.8	-1	7.8	-1	4
-1	5.6	-1	4.6	-1	3.6	-1	1
0	2.6	0	1.6	0	0.6	0	0
1	5.2	0	4.2	0	2.4	0	1
2	8.2	1	7.2	1	5.4	1	4

Exercise: 1 (d) Compare the costs and optimal control from the deterministic and probabilistic models and explain where, in your opinion, the differences come from.

Answer Shown below are the results from the deterministic and stochastic case of our system -

State	J_0^*	u_0^*	J_1^*	u_1^*	J_2^*	u_2^*	J_3^*
-2	10	0	9	0	8	0	4
-1	6	-1	5	-1	4	-1	1
0	3	-1	2	-1	1	-1	0
1	4	0	3	0	2	0	1
2	7	1	6	1	5	0	4

Table 1: Deterministic Policy Values

State	J_0^*	u_0^*	J_1^*	u_1^*	J_2^*	u_2^*	J_3^*
-2	9.8	-1	8.8	-1	7.8	-1	4
-1	5.6	-1	4.6	-1	3.6	-1	1
0	2.6	0	1.6	0	0.6	0	0
1	5.2	0	4.2	0	2.4	0	1
2	8.2	1	7.2	1	5.4	1	4

Table 2: Stochastic Policy Values

Upon observing the two tables side-by-side, we can see that the costs for the stochastic case are slightly lesser as compared to the deterministic system. Accordingly, the optimal control differs, proportionally account for this. This is due to the inclusion of the random variable ω_n, that introduces the uncertainty in the system model.

In the stochastic case, we take the *Expectation* over the probability distribution, which results in slightly lower costs, thus making the system slightly more robust than the deterministic case.

Exercise: 2 Consider the grid-world shown in Figure 1. In each (non grey) cell, it is possible to perform five actions: move up, down, left, right or do nothing as long as the resulting move stays inside the grid world. Grey cells are obstacles and are not allowed. We would like to find the optimal value function and optimal policy that minimize the following cost

$$\min \sum_{n=0}^{\infty} \alpha^n g_n(x_n)$$

with discount factor $\alpha = 0.99$ and where the instantaneous cost is defined as

$$g_n(x_n) = \begin{cases} -1 & \text{if } x_n \text{ is a violet cell} \\ 0 & \text{if } x_n \text{ is a white cell} \\ 1 & \text{if } x_n \text{ is a green cell} \\ 10 & \text{if } x_n \text{ is a red cell} \end{cases}$$

In a Jupyter notebook answer the following questions:

(a) Implement the value iteration algorithm to solve the problem (initialize the value function to 0). How many iterations does it take to attain convergence? (we assume here that convergence happens when all the elements of the value function do not change more than 10^{-6} in a new iteration.

(b) Implement the policy iteration algorithm to solve the problem (use the version that solves the linear equation $(I - \alpha A)J_\mu = \bar{g}$). Start with an initial policy that does not move. How many iterations does it take to converge?

(c) Compare the solutions and convergence/complexity of each algorithm to solve this problem.

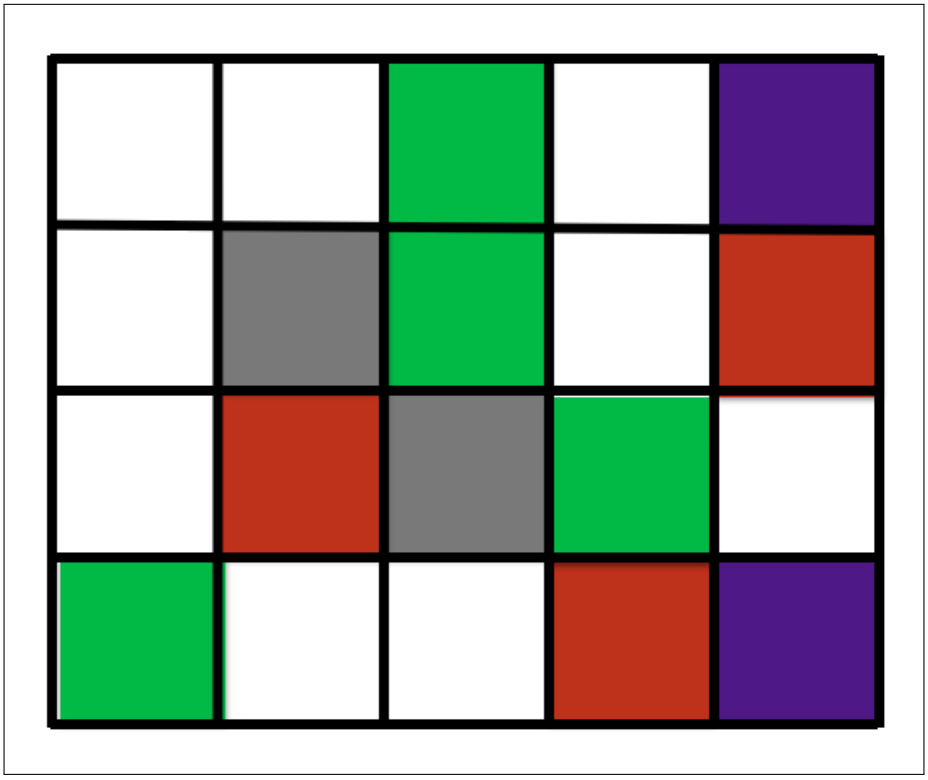

Figure 1: Grid World

Answer Please follow the link to the Jupyter Notebook for code solution of Exercise 2.