Exercise 7: Dynamic Programming

Prof. Dr. Moritz Diehl, Andrea Zanelli, Dimitris Kouzoupis, Florian Messerer, Yizhen Wang

In this exercise, we will use dynamic programming (DP) to implement a controller for the inverted pendulum from Exercise 6,

$$
\begin{aligned}
\dot{\theta} &= \omega \\
\dot{\omega} &= \sin(\theta) + \tau,
\end{aligned}
\tag{1}
$$

where θ is the angle describing the orientation of the pendulum, ω is its angular velocity and τ is the input torque. The goal is to design a feedback policy capable of swinging up the pendulum starting from $\theta = \pi$. Moreover, we will prove the Schur Complement Lemma, which can be used to derive the formulation for the LQR controller.

1. **Dynamic programming:** Consider the following optimal control problem,

$$
\min_{\substack{x_0,\ldots,x_N, \\ u_0,\ldots,u_{N-1}}} \quad \sum_{i=0}^{N-1} \left(x_i^T Q x_i + u_i^T R u_i \right) + x_N^T Q_N x_N
\tag{2a}
$$

$$
\text{s.t.} \quad x_0 = \bar{x}_0,
\tag{2b}
$$

$$
x_{i+1} = F(x_i, u_i), \quad i = 0, \ldots, N-1,
\tag{2c}
$$

$$
-10 \le u_i \le 10, \quad i = 0, \ldots, N-1,
\tag{2d}
$$

where $F(x, u)$ describes the discretized dynamics obtained by applying one step of the explicit RK4 integrator with step-size $h = 0.1$ to (1).

(a) Consider the unconstrained linear quadratic infinite horizon problem that is obtained from (2) by linearizing the dynamics at $x_{\text{lin}} = (0, 0)$, $u_{\text{lin}} = 0$, and dropping the control constraints,

$$
\min_{\substack{x_0,x_1,\ldots,, \\ u_0,u_1,\ldots,}} \quad \sum_{i=0}^{\infty} \left(x_i^T Q x_i + u_i^T R u_i \right)
\tag{3a}
$$

$$
\text{s.t.} \quad x_0 = \bar{x}_0,
\tag{3b}
$$

$$
x_{i+1} = A x_i + B u_i, \quad i = 0, 1, \ldots,
\tag{3c}
$$

where $A := \frac{\partial F(x,u)}{\partial x}\big|_{\substack{x=x_{\text{lin}} \\ u=u_{\text{lin}}}}$ and $B := \frac{\partial F(x,u)}{\partial u}\big|_{\substack{x=x_{\text{lin}} \\ u=u_{\text{lin}}}}$.

Complete the template `LQR_design.m / LQR_design.py` to obtain the LQR gain matrix K, which defines the optimal control at each stage as the time-independent linear feedback law $u_i^*(x) = -Kx$.

Hint: MATLAB function `dlqr` /Python function `scipy.linalg.solve_discrete_are`

(b) Complete the template `dynamic_programming.m / dynamic_programming.py` to implement the DP algorithm and use it to compute the cost-to-go associated with the initial state of (2). Choose $N = 20$, $Q = \text{diag}(100, 0.01)$, $R = 0.001$ and Q_N equal to the cost matrix associated with the LQR controller. Discretize the angle θ into 200 values between $-\frac{\pi}{2}$ and 2π. Analogously, discretize the angular velocity into 40 values between -10 and 10 and the torque τ into 20 values between 10 and -10.

Remark: in order to compute the cost-to-go you will have to project the state obtained by simulating the dynamics forward onto the defined discretization grid. To this end, use in your code the MATLAB / Python function `project` provided with this exercise.

(c) Consider the plots provided by the previous template, showing the cost of DP and LQR as well as their control policies. Where is the LQR policy similar to the one obtained with DP? Where is it different? Why?

(d) Complete the template `closed_loop.m` / `closed_loop.py` to obtain a closed loop simulation of the system, for both LQR and DP. For LQR keep in mind that it was obtained without considering the control constraints, so you have to clip the controls obtained from the LQR feedback law to the feasible control interval $[-10, 10]$. For the DP controller we will always consider the current state of the system as initial state of (2), so you can choose the control according to the cost-to-go function obtained as result of the recursion in (b).

Which of the two controllers achieves the better performance? Why?

2. **Schur Complement Lemma:** Consider the following lemma:

Lemma 1 *(Schur Complement Lemma) Let R be a positive-definite matrix. Then, the following holds:*

$$\min_{u} \begin{bmatrix} x \\ u \end{bmatrix}^T \begin{bmatrix} Q & S^T \\ S & R \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} = x^T (Q - S^T R^{-1} S) x \tag{4}$$

and the minimizer $u^*(x)$ is given by $u^*(x) = -R^{-1} S x$.

Prove Lemma 1.