Exercises for Lecture Course on Optimal Control and Estimation (OCE)
Albert-Ludwigs-Universität Freiburg – Summer Term 2015

Exercise 4 - Linear Quadratic Problems (deadline: 25.5.2014, 2:15pm)

Prof. Dr. Moritz Diehl and Jonas Koenemann

1 Linear Quadratic Regulator

Consider the inverted pendulum problem in Exercise 3 of Exercise Sheet 3. This time, we find an approximate solution to the optimal control problem by linearising the original non-linear problem and finding a solution to the corresponding linear-quadratic problem.

Recall the system dynamics of the pendulum with state $x = [\phi, \omega]^\top$:

$$\dot{x} = \begin{bmatrix} \omega \\ 2\sin(\phi) + u \end{bmatrix}$$

and the optimal control problem

$$\begin{aligned} \underset{x_0,\dots,x_N,u_0,\dots,u_{N-1}}{\text{minimize}} \quad & \sum_{k=0}^{N-1} (\phi_k^2 + u_k^2) \\ \text{subject to} \quad & \\ f(x_k,u_k) - x_{k+1} &= 0, \quad \text{for} \quad k = 0,\dots,N-1 \\ \bar{x}_0 &= x_0 \\ u_{\min} &\le u_k \le u_{\max}, \quad \text{for} \quad k = 0,\dots,N-1 \\ \omega_{\min} &\le \omega_k \le \omega_{\max}, \quad \text{for} \quad k = 0,\dots,N \end{aligned}$$

In this sheet, our aim is to solve a linear-quadratic problem of the form:

$$\begin{aligned} \underset{x_0,\dots,x_N,u_0,\dots,u_{N-1}}{\text{minimize}} \quad & \sum_{k=0}^{N-1} \begin{bmatrix} x_k \\ u_k \end{bmatrix}^T \begin{bmatrix} Q_k & S_k^T \\ S_k & R_k \end{bmatrix} \begin{bmatrix} x_k \\ u_k \end{bmatrix} + x_N^T P_N x_N \qquad (1) \\ \text{subject to} \quad & x_0 = \bar{x}_0, \\ & x_{k+1} = A_k x_k + B_k u_k, \quad \text{for} \quad k = 0,\dots,N-1. \end{aligned}$$

We recommend using the template `lqr_template.m` from the course website to solve the tasks because some functions of the previous exercise sheet are needed.

Tasks

1. Linearise the system dynamics around the point $x = [0,0]^\top$, $u = 0$ analytically by differentiation to obtain a continuous time system of the form: $\dot{x} = A_c x + B_c u$. Then you can obtain the corresponding discrete time system $x_{k+1} = Ax_k + Bu_k$ with a timestep of 0.1 using the Matlab commands:

```
sysc = ss(Ac,Bc,eye(2),0);
sysd = c2d(sysc,0.1);
A = sysd.a;
B = sysd.b;
```

   Specify the continuous time and discrete time system matrices A_c, B_c, A, B.

   Alternatively, you can linearise the discrete time dynamic system as in Exercise Sheet 2. In this case specify the discrete time system matrices A, B. If you do the linearisation both ways you get **1 bonus point**.

   (1 point + 1 bonus point)

2. Bring the objective of the optimal control problem into a quadratic form and specify the matrices Q, R, S.

   (1 point)

3. Calculate recursively the P_k matrices (Difference Ricatti Equation) for $k = N-1,\dots,0$ with $N = 60$ starting from $P_N = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$.

   (2 points)

4. Calculate the first cost-to-go J_0 for the linear system for each state of the dynamic programming exercise of exercise sheet 3. Make a 3D contour plot with a fine level-step of J_0 using the matlab commands:

```
[C,handle] = contour3(J0);
set(handle,'LevelStep',get(handle,'LevelStep')*0.2)
```

   Compare the contour plot with the contour plot of the non-linear cost-to-go function of the dynamic programming exercise the you can get with the `get_first_J()` function in the template.

   What are the similarities/differences of the two contour plots?

   (1 point)

5. Starting from $x_0 = [-\frac{\pi}{8}, 2]^\top$, calculate the optimal feedback and the complete optimal trajectory of the linear quadratic optimal control problem by forward recursion.

   Make a plot with the evolution of the state in time and a plot of the optimal feedback controls vs. time. Does the controller bring the system to the steady state?

   (1 point)

6. Apply the same optimal controls (open-loop) to the non-linear pendulum system. Start from the same initial state x_0 and simulate the system using the `integrate_rk4` function. Make a plot of state evolution and controls as before. Does the controller bring the system to the steady state? Discuss the result.

   (1 point)

7. Implement a feedback controller, i.e. calculate the optimal feedback for the current state and simulate the non-linear system using `integrate_rk4`. Make a plot of state evolution and controls as before. Does the controller bring the system to the steady state?

   (1 point)

8. Solve the Algebraic Ricatti Equation by iteratively calculating the P_k matrices until convergence. We say, convergence is reached if the Frobenius norm of differences between the current matrix P_{cur} and the next matrix P_{next} is below 10^{-5}:

```
norm(Pnext-Pcur, 'fro') <= 1e-5
```

   (1 point)

9. Use the solution to the Algebraic Ricatti Equation to implement a Linear-Quadratic-Regulator(LQR). Simulate the system with `integrate_rk4`. Make a plot of state evolution and controls as before. Does the controller stabilize the system at the steady state?

   (1 point)

10. **Bonus question:** Try different initial states for the simulation with optimal feedback law and the LQR controller. For which initial states do the controllers stabilize?

   (1 bonus point)