Multivariable Control (ME-422) - Exercise session 6

Prof. G. Ferrari Trecate

1. Consider the MIMO DT LTI system

$$x^+ = \begin{bmatrix} 0 & 2 & 3 \\ 0 & -2 & 4 \\ 0 & 0 & 3 \end{bmatrix} x + \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

   (a) Is it possible to assign the eigenvalues using a scalar channel? If yes, design a state-feedback controller assigning the eigenvalues in $\frac{1}{2}$, $\frac{1}{3}$, and $\frac{1}{4}$.
   Hint: Relevant MATLAB commands are `ctrb` and `place`.

   (b) Use the probabilistic approach seen in the lectures for assigning the eigenvalues.

2. **Inverted Pendulum on Cart**
   In this exercise, you will discretize, simulate, and control a cart with an inverted pendulum on top. This exercise will be mainly based on MATLAB and Simulink. A folder containing all the necessary files for simulating the system is provided in Moodle; you will only have to familiarize with the simulation file and implement into it the controllers that you design.

   (a) Download the file *Inverted+pendulum+2018-1+no+solutions.zip* and unzip it. Read the document *Inverted+Pendulum+description.pdf*, which describes the dynamics of the system.

   (b) Run the initialization file *pendulum_sys_init.m*.

   - Open the file *pendulum_openloop_anim.mdl*. If you have the "Simulink 3D Animation" toolbox installed, you will see the pendulum animation. Otherwise, comment the block named "VR_visualization" to run a simulation. Values of variables will be still visible in the scope blocks.
     Hint: To install, under the "APPS" tab in the MATLAB windows, select "Get More Apps" and search for the toolbox.
   - Linearize the system around the obtained equilibrium. To do that, run the Simulink file *pendulum_openloop_anim.mdl*. Thanks to the "Time-Based Linearization" block you will find, in the workspace, the structure "pendulum_openloop_anim_Time_Based_Linearization" with the matrices of the linearized system.
   - Run a simulation, add a perturbation $V = 0.0001$ to the input and observe the result. Then, run a second simulation with $V = 0$ in order to store the correct linearized model in the workspace, before you proceed with the rest of the exercise.
   - Explore the Simulink blocks. In particular, zooming into the subsystem block, the initial state for a simulation is stored in the "integrator" block.

   (c) Using MATLAB, discretize the system with sampling time $T = 0.1s$ and exact discretization.
   Hint: Use the MATLAB function `c2d`.

   (d) Assume that the state is fully accessible. After checking if the discretized system is reachable, design a controller that assigns its eigenvalues in e^{-2T}, $e^{-2.5T}$, e^{-3T}, and e^{-4T}.

   Hint: To compute the reachability matrix, use the MATLAB function `ctrb(A,B)`. To calculate the pole placement gain, you can use the MATLAB function `place(A,B,P)`, where P is a vector containing the values of the desired poles.

   The simulation file *pole_placement_pendulum_anim.slx* simulates how the closed-loop system reacts to pulse disturbances (of amplitude 5) on the nominal voltage $V = 0$, for a given choice of feedback gain K. Run this model with the K matrix you calculated to check the performance of the controller against disturbance. Note that the pendulum nonlinear model is present in the loop.

- Increase the pulse amplitude to 20 and repeat the simulation.

(e) Design a new controller assigning the eigenvalues in e^{-10T}, e^{-3T}, $e^{-6.5T}$, and e^{-6T}.

Run a simulation and compare with the previous regulator. Keep increasing the disturbance amplitude. Which controller provides better performance? Can you say intuitively? Why?