**Automatic Control course project**

Yunju Li - 247402

# Project description

The project focuses on the design and analysis of a rotary inverted pendulum system. The system comprises two primary components: the inverted pendulum and the rotary arm. These components are connected via a revolute joint and are manipulated by a motor that controls the angular position of the rotary arm. The primary objective is to develop a control strategy to stabilize the inverted pendulum in the upright position.

# Mathematical model

*To describe the system's dynamics, we introduce two angles:*

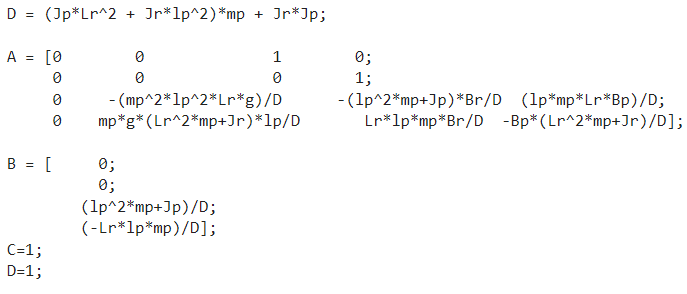
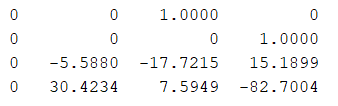
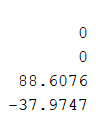
*θ: The angle of rotation of the rotary arm in the plane.*

*α: The angle of rotation of the inverted pendulum about the revolute joint. The configuration where the pendulum points directly upward is identified with α=0.*

*The relationship between the current provided to the motor and the resulting torque is modeled as a dynamical system. However, for simplification, the motor's dynamics are neglected, and we assume direct control over the torque.  
Then, for q := [θ α], the equations describing the time evolution of the angular coordinates are  
M(q) ̈q + C(q, q ̇) ̇q + fv( ̇q) + G(q) = τ.*

# Answer to the questions

*3.1 Question 1*

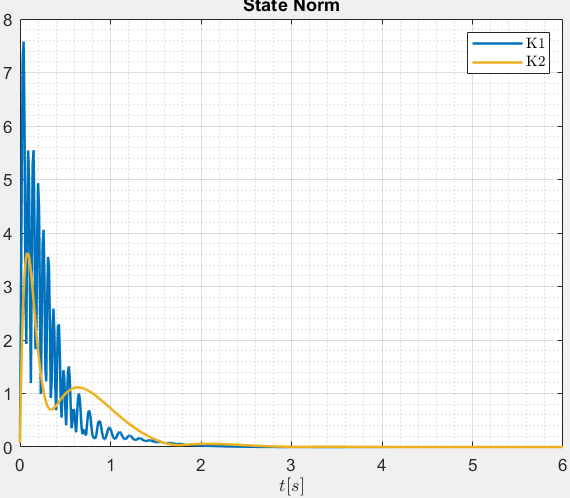
*Linearize the equations of dynamics (1) with respect to the origin x = 0 and write the linearized system in the state space representation ̇x = Ax + Bu. Is the system asymptotically stable, stable or unstable? Is the system controllable?  
  
Answer: To linearize the equations of dynamic, I get the jacobian matrix at first, then use the origin point to compute. By this I get A and B.  
  
With all the data, A and B are:  
A:   
 B:  
The real part of eigenvalues of A are: 0, -84.7869, 0.3661 and -16.0011. So the system is unstable. To figure out the controllability of the system, consider the rank of controllability matrix. The rank of controllability matrix is 4, which is full rank, so the system is controllable.*

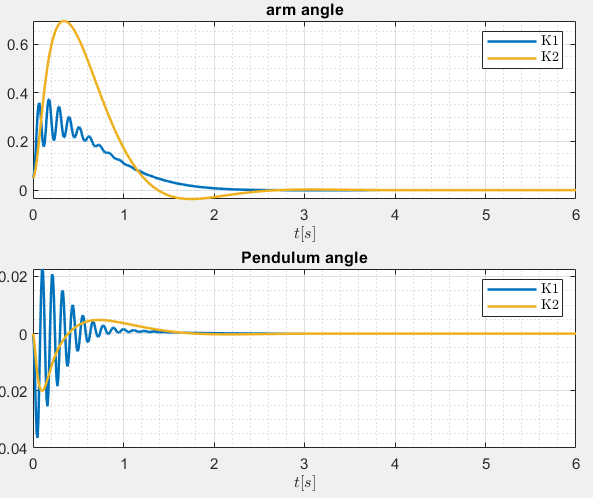
*3.2 Question 2*

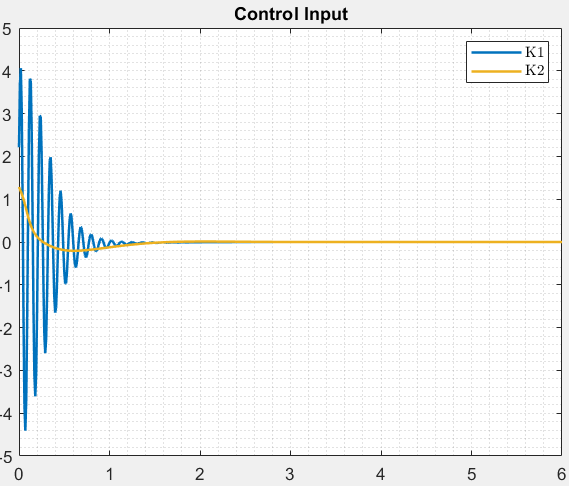
*Compute a gain matrix K1 such that the closed-loop system ̇x = A + BK1 has a convergence rate α = 2. Compute a second gain matrix K2 ensuring the same convergence rate of the closed-loop system such that K2 has minimum norm. Report the two gain matrices and simulate the two nonlinear closed-loop systems obtained by substituting in the nonlinear dynamics (1) the input selection u = Kix, i ∈ {1, 2}, for 6 seconds, starting from the initial*

*configuration. Based on the simulation results, which of the two controllers would you implement and why?  
The results of gain matrix are:  
K1 = [ -5.0814 -5.0814 -2.3464 -2.3464 ]  
K2 = [ 5.255295 3059.0724 17.070819 39.949526 ]*

*Simulation results of gain matrix K1 and K2:*

*State Norm:  
*

*Arm angle(θ) and Pendulum angle(α):  
*

*Control input:  
*

*I will choose to implement the controller with minimum control effort. Because in terms of control input, it’s more smooth. For another controller, control input oscillation is severe, which is difficult for the rotor to generate and it’s harmful for the mechanical system.*

*3.3 Question 3  
Compute the overshoot M corresponding to the closed-loop matrix A + BK for*

*the two gain matrices K computed in the previous step. Plot the exponential bound on the trajectories of the previous point. Does the exponential bound hold also for the solutions of the nonlinear system?*

*To compute the overshoot, code used in matlab is   
A\_m = A+B\*K\_b;*

*yalmip('clear');*

*% -- variables*

*P = sdpvar(n,n);*

*k = sdpvar(1,1);*

*% -- LMIs*

*constr = [ eye(n) <= P;P <= k\*eye(n);*

*A\_m'\*P + P\*A\_m <= -2\*alpha\_bar\*P ];*

*sol = solvesdp(constr, k, opts);*

*[primal\_res, dual\_res] = check(constr);*

*feas = all([primal\_res; dual\_res] > -fake\_zero);*

*disp(newline);*

*disp('-----------------------')*

*disp('Estimating overshoot M');*

*k\_val = value(k);*

*M = sqrt(k\_val);  
But in the project, the solver is not able to solve so the question is infeasible.  
To plot the exponential bound on the trajectories of the previous point, the code is:  
[t\_ode, x\_ode] = ode45( @(t,x)A\_m\*x, SimT, q0);*

*x\_norm = vecnorm(x\_ode');*

*x\_bound = M\*norm(q0)\*exp(-alpha\_bar\*SimT);*

*figure(5), clf*

*hold on*

*grid minor*

*plot(SimT, normX\_b, 'b');*

*plot(SimT, x\_bound, '--r');*

*legend('$|x(t)|$', '$M|x(0)|e^{-\bar \alpha t}$', 'interpreter', 'latex');*

*title('Trajectory');*

*xlabel('$t[s]$', 'interpreter', 'latex');*

*M is the overshoot that is supposed to get from the previous step. x\_bound is exponential bound.*