Problem Set 3

Monday, October 10, 2022 1:03 PM

Problem 1 (4 points)

Consider a *fully-actuated* cart-pole system shown in Figure 1, in which a motor is mounted in the drive-train of the cart that can apply an external force F in the horizontal direction, and another actuator is mounted in the pivot of the rod that applies a torque τ.

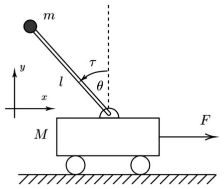


Figure 1: A fully-actuated cart-pole system.

The cart has mass M, and the pendulum is a massless rod of length l with a point mass m on top of it. The horizontal direction x denotes the displacement of the center of the cart from the origin, and θ is the angle of the pole with respect to the vertical.

The equations of motion for the system are given as:

$$(M+m)\ddot{x} - ml\ddot{\theta}\cos(\theta) + ml\dot{\theta}^2\sin(\theta) = F$$
$$ml^2\ddot{\theta} - ml\ddot{x}\cos\theta - mgl\sin\theta = \tau$$

a) Specify the generalized coordinates q and the generalized forces u for the system, and re-write the equations of motion in the manipulator equation form:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = u$$

b) Design a *symbolic* feedback linearization control law to regulate the cart-pole system so that the cart stabilizes at the point $x = 0$ and the pole is balanced perfectly vertically in an upright position.

Do not forget to suggest an expression for the virtual control input used in the feedback linearization control. Provide the overall control law for the system explicitly (that is, do not leave any matrices in the final control law, instead, provide a scalar symbolic expression for each of the control inputs). Provide all the necessary conditions for the control gains used in the control law.

A) (a) * Generalized Coordinates:

$$q = \begin{bmatrix} x \\ \theta \end{bmatrix} \quad u = \begin{bmatrix} F \\ \tau \end{bmatrix}$$

$$(M+m)\ddot{x} - ml\cos\theta\cdot\ddot{\theta} + ml\dot{\theta}^2\sin\theta = F$$
$$-ml\cos\theta\,\ddot{x} + ml^2\ddot{\theta} - mgl\sin\theta = \tau$$

* Manipulator Eqn. Form

$$\underbrace{\begin{bmatrix} (M+m) & -mlc\theta \\ -mlc\theta & ml^2 \end{bmatrix}}_{M(q)} \underbrace{\begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix}}_{\cdot\ddot{q}} + \underbrace{\begin{bmatrix} 0 & ml\dot{\theta}s\theta \\ 0 & 0 \end{bmatrix}}_{C(q,\dot{q})} \underbrace{\begin{bmatrix} \dot{x} \\ \dot{\theta} \end{bmatrix}}_{\cdot\dot{q}} + \underbrace{\begin{bmatrix} 0 \\ -mgls\theta \end{bmatrix}}_{g(q)} = \underbrace{\begin{bmatrix} F \\ \tau \end{bmatrix}}_{u}$$

(b) Designing symbolic feedback linearization Control law:

$$u = M(q)\cdot \underbrace{v(t)}_{\text{Linearized control.}} + C(q,\dot{q})\cdot\dot{q} + g(q) \quad -①$$

Equating ① w/ Manipulator Eqn.:

$$v(t) = \ddot{q} \Rightarrow \begin{cases} v_1 = \ddot{x} \\ v_2 = \ddot{\theta} \end{cases}$$

for v_1 $(v_1 = \ddot{x})$ $\Rightarrow$ $z = \begin{bmatrix} x \\ \dot{x} \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$

$$\dot{z} = \begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} z_2 \\ v_1 \end{bmatrix}$$

$$\therefore \dot{z} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix}\begin{bmatrix} v_1 \end{bmatrix}$$

$$v_1 = -Kz = -\begin{bmatrix} K_1 & K_2 \end{bmatrix}\begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

$$A' = A - BK = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \end{bmatrix}\begin{bmatrix} K_1 & K_2 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ K_1 & K_2 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 1 \\ -K_1 & -K_2 \end{bmatrix}$$

$|A' - \lambda I| = 0$

$$\Rightarrow \begin{vmatrix} -\lambda & 1 \\ -K_1 & -K_2\lambda \end{vmatrix} = 0$$

$$\lambda(\lambda + K_2) + K_1 = 0$$
$$\lambda^2 + K_2\lambda + K_1 = 0$$

Assume $\lambda = -1, -2$

$$\therefore (\lambda+1)(\lambda+2) = 0$$
$$\lambda^2 + 3\lambda + 2 = 0 \quad -②$$

from ① & ②

$$K_1 = 3 \quad K_2 = 3$$

$$\therefore v_1 = -\begin{bmatrix} 2 & 3 \end{bmatrix}\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = -2x - 3\dot{x}$$

Similarly for $v_2 \Rightarrow$ $v_2 = \ddot{\theta}$

$$z = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

$$\dot{z} = \begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} z_2 \\ v_2 \end{bmatrix}$$

$$\dot{z} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} z + \begin{bmatrix} 0 \\ 1 \end{bmatrix} v_2$$

$$v_2 = -Kz = -\begin{bmatrix} K_1 & K_2 \end{bmatrix} z$$

$$\therefore A' = A - BK = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \end{bmatrix}\begin{bmatrix} K_1 & K_2 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ K_1 & K_2 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 1 \\ -K_1 & -K_2 \end{bmatrix}$$

$$|A' - \lambda I| = 0$$

$$\begin{vmatrix} -\lambda & 1 \\ -K_1 & -K_2-\lambda \end{vmatrix} = 0$$

$$\lambda(\lambda + K_2) + K_1 = 0$$

$$\lambda^2 + K_2\lambda + K_1 = 0 \quad -(3)$$

Assume $\lambda = -3, -4$

$$\therefore (\lambda+3)(\lambda+4) = 0$$

$$\lambda^2 + 7\lambda + 12 = 0 \quad -(4)$$

from (3) & (4)

$$K_2 = 7$$

$$K_1 = 12$$

$$v_2 = -\begin{bmatrix} 12 & 7 \end{bmatrix} z$$

$$\therefore v_2 = -12\theta - 7\dot{\theta}$$

$$v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} -2x - 3\dot{x} \\ -12\theta - 7\dot{\theta} \end{bmatrix}$$

Stabilization at $q^* = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \rightarrow$ $x \rightarrow x - 0$, $\dot{x} \rightarrow \dot{x} - 0$, $\theta \rightarrow \theta - 0$, $\dot{\theta} \rightarrow \dot{\theta} - 0$

$$\therefore v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} -2x - 3\dot{x} \\ -12\theta - 7\dot{\theta} \end{bmatrix} = \begin{bmatrix} -2x - 3\dot{x} \\ -12\theta - 7\dot{\theta} \end{bmatrix}$$

$$\therefore u = M(q) \cdot v(t) + C(q, \dot{q})\dot{q} + g(q)$$

$$u = \begin{bmatrix} (M+m) & -ml c\theta \\ -ml c\theta & ml^2 \end{bmatrix}\begin{bmatrix} -2x - 3\dot{x} \\ -12\theta - 7\dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 & ml\dot{\theta} s\theta \\ 0 & 0 \end{bmatrix}\begin{bmatrix} \dot{x} \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ -mgl s\theta \end{bmatrix}$$

$$\therefore \boxed{\begin{aligned} F &= -(M+m)(2x + 3\dot{x}) + ml c\theta(12\theta + 7\dot{\theta}) + ml\dot{\theta}^2 s\theta \\ \tau &= ml c\theta(2x + 3\dot{x}) + ml^2(-12\theta - 7\dot{\theta}) - mgl s\theta \end{aligned}}$$

Problem 2 (8 points)

Vehicle platoons involve groups of vehicles travelling together at a constant inter-vehicle distance. Consider the vehicle platoon scenario in Figure 2, in which the vehicles are supposed to stop at a stop sign (considered as the origin) while maintaining a safe distance.

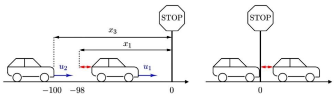


Figure 2: Vehicle platoon in stop scenario. The red arrow shows the safety region, not to be violated. Left: initial condition, Right: desired final condition.

The dynamics of vehicle 1 and vehicle 2 are derived as:

$$\dot{x}_2 = \ddot{x}_1 = u_1$$
$$\dot{x}_4 = \ddot{x}_3 = 0.5\,u_2$$

where x_1 is the distance of rear bumper of vehicle 1 from the stop sign plus an offset, and x_3 is the distance of the front bumper of vehicle 2 from the stop sign. The offset of x_1 from vehicle 1's rear bumper is the safety distance, preventing the vehicles from collision when $x_3 = x_1$. The states x_2 and x_4 represent the velocity of vehicle 1 and vehicle 2, respectively.

The inputs u_1 and u_2 are the vehicles' thrust generated by accelerating and braking.

The system outputs are defined as $y_1 = x_1$ and $y_2 = x_3 - x_1$. Note that if $y_2 > 0$ then vehicle 2 has violated the safety region of vehicle 1.

The vehicles start from initial conditions of $[-98\,(m),\ 20\,(m/s)]$ for vehicle 1, and $[-100\,(m),\ 25\,(m/s)]$ for vehicle 2.

a) Write the system dynamics in the state-space form.

b) Design an LQR controller for the system with the weight matrices given by:

$$Q = diag([2,\ 5,\ 2,\ 5]), \qquad R = diag([1,\ 1])$$

Feel free to use MATLAB to design the control law. Explicitly write down the solution P to the Riccati equation, the control gains K, and the final state-feedback control law for the system.

c) Simulate the system in MATLAB on a time span of 10 seconds, with the designed control law in part (b) and from the initial conditions specified above. Include the plots of the system response (all the states and the outputs) and control inputs, and discuss whether the safety region is violated or not (and if so by how much).

Hint: To simulate linear systems in MATLAB, you don't need to form an `ode` function or use the `ode45()` command. Instead, you can simply use the `ss()` command to create a state-space model, and the `initial()` command to obtain the unforced system response to initial states of a state-space. For more information, see the links below:
`https://www.mathworks.com/help/control/ref/ss.html`
`https://www.mathworks.com/help/control/ref/lti.initial.html`

d) In part (c), within 10 seconds both vehicles arrive within $1\,(m)$ of the stop sign at a speed less than $1\,(m/s)$. However, at some points of the stopping trajectory, vehicle 2 lagged more than 6 meters behind vehicle 1, which is not desired.

Tune the weight matrix $Q = diag([q_1,\ q_2,\ q_3,\ q_4])$ to keep vehicle 2 within $2\,(m)$ of vehicle 1 for the entire trajectory, while maintaining $y_2 < 0$ to prevent crashing (do not change the R matrix). The vehicles must again arrive at the distance of less than $1\,(m)$ from the stop sign and the velocity of less than $1\,(m/s)$ by 10 seconds.

Specify the solution P to the Riccati equation and the new gains K for the LQR control law.

Using MATLAB, plot $x(t)$ and $u(t)$ for the same initial conditions specified above and with the state feedback with new gains K. Include the plots in your submission.

Hint: set $q_1 = 2.5$ and $q_2 = 10$, and then tune q_3 and q_4.

e) Briefly discuss the trade-offs between the time response and the control effort between part (c) and part (d).

A) (a)

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \qquad \dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} x_2 \\ u_1 \\ x_4 \\ 0.5\,u_2 \end{bmatrix} \qquad u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\dot{x} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{A} x + \underbrace{\begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0.5 \end{bmatrix}}_{B} u$$

$$y_1 = x_1$$
$$y_2 = x_3 - x_1$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_3 - x_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$y = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \end{bmatrix}}_{C} x + \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}}_{D} u$$

ARE.pdf

```
clc;
clear all;
close all;
```

System Information

```
initial_conditions = [-98;20;-100;25]
```

```
initial_conditions = 4x1
   -98
    20
  -100
    25
```

```
tspan = 10
```

```
tspan = 10
```

Writing System

```
syms x1(t) x2(t) x3(t) x4(t) u1(t) u2(t) t 'real'
x = [x1;x2;x3;x4];
u = [u1;u2];
dx = [x2; u1; x4 ; u2/2]
```

dx(t) =

$$\begin{pmatrix} x_2(t) \\ u_1(t) \\ x_4(t) \\ \frac{u_2(t)}{2} \end{pmatrix}$$

Writing System Dynamics

```
A = [0 1 0 0; 0 0 0 0; 0 0 0 1; 0 0 0 0]
```

```
A = 4x4
     0     1     0     0
     0     0     0     0
     0     0     0     1
     0     0     0     0
```

```
B = [0 0; 1 0; 0 0; 0 0.5]
```

```
B = 4x2
         0         0
    1.0000         0
         0         0
         0    0.5000
```

```
C = [1 0 0 0; -1 0 1 0]
```

```
C = 2x4
```



```
     1     0     0     0
    -1     0     1     0
```

```
D = [0 0;0 0]
```

```
D = 2x2
     0     0
     0     0
```

Checking Controllability

```
CO = ctrb(A,B)
```

```
CO = 4x8
         0         0    1.0000         0         0         0         0         0
    1.0000         0         0         0         0         0         0         0
         0         0         0    0.5000         0         0         0         0
         0    0.5000         0         0         0         0         0         0
```

```
r_CO = rank(CO)
```

```
r_CO = rank(CO)
```

```
r_CO = 4
```

```
if(rank(A) == r_CO)
    disp("System is controllable")
else
    disp("System is not controllable")
end
```

```
System is not controllable
```

Designing LQR Controller with provided Q,R

```
Q = diag([2,5,2,5])
```

```
Q = 4x4
     2     0     0     0
     0     5     0     0
     0     0     2     0
     0     0     0     5
```

```
R = diag([1,1])
```

```
R = 2x2
     1     0
     0     1
```

```
[P,K,L] = icare(A,B,Q,R)
```

```
P = 4x4
    3.9569    1.4142   -0.0000   -0.0000
    1.4142    2.7979   -0.0000   -0.0000
   -0.0000   -0.0000    4.6167    2.8284
   -0.0000   -0.0000    2.8284    6.5290
K = 2x4
    1.4142    2.7979   -0.0000    0.0000
   -0.0000    0.0000    1.4142    3.2645
```

$\Rightarrow P$

$\Rightarrow$ Gain Matrix K


```
L = 4x1 complex
  -0.6622 + 0.0000i
  -0.8161 + 0.2026i
  -0.8161 - 0.2026i
  -2.1358 + 0.0000i
```

```
u1 = - K(1,:)*x;
u2 = - K(2,:)*x;
dx = [x2; u1; x4 ; u2/2]
```

$\rightarrow u = -\begin{bmatrix} 1.4142x_1 + 2.7979x_2 \\ 1.4142x_3 + 3.2645x_4 \end{bmatrix} \rightarrow$ S.F. Control law

$dx = \begin{bmatrix} x_2 \\ 1.4142x_1 + 2.7979x_2 \\ x_4 \\ 0.7071x_3 + 1.63225x_4 \end{bmatrix} \rightarrow$ State Dynamics.

```
dx(t) =
```

$$\begin{pmatrix} x_2(t) \\ \frac{1859259797328309\,x_3(t)}{5070602400912917605986812821504} - \frac{6300384224323929\,x_2(t)}{2251799813685248} - \frac{6349385536679153\,x_4(t)}{1622592768292133633915780102} \\ x_4(t) \\ \frac{8198733962548731\,x_1(t)}{40564819207303340847894502572032} - \frac{597279309511843\,x_2(t)}{20282409603651670423947251286016} - \frac{918870491619799}{5629499534213} \end{pmatrix}$$

Testing System Output

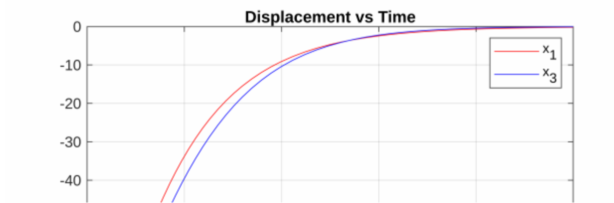
```
vars = [x1(t) x2(t) x3(t) x4(t)];
func_dx = odeFunction(dx,vars);
fdx = @(t,x)func_dx(t,x);
[t,x] = ode45(fdx,[0,tspan],initial_conditions);
for i=1:1:size(t)
    y(i,:) = (C*x(i,:)')';
    inputs(i,:) = (-K*x(i,:)')';
end
```

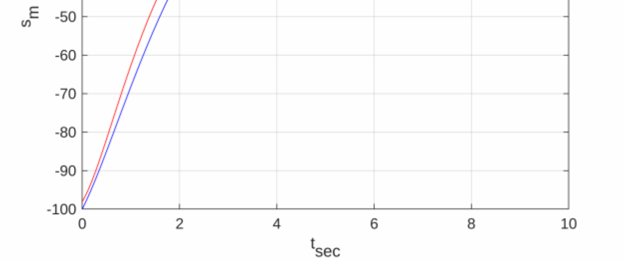
Plotting Graphs

```
disp("Plotting States")
```

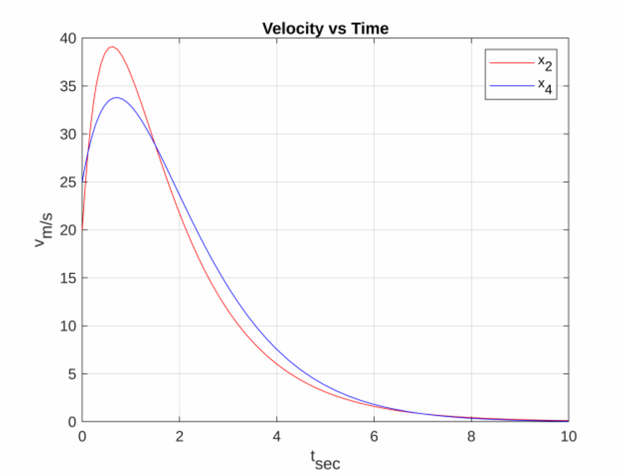
```
Plotting States
```

```
plot(t,x(:,1),'r')
hold on;
plot(t,x(:,3),'b')
hold on;
legend ('x_{1}','x_{3}');
grid on;
xlabel("t_{sec}")
ylabel("s_{m}")
title("Displacement vs Time")
hold off;
exportgraphics(gcf,'default_tuned_x1x3.png','Resolution',1200)
```



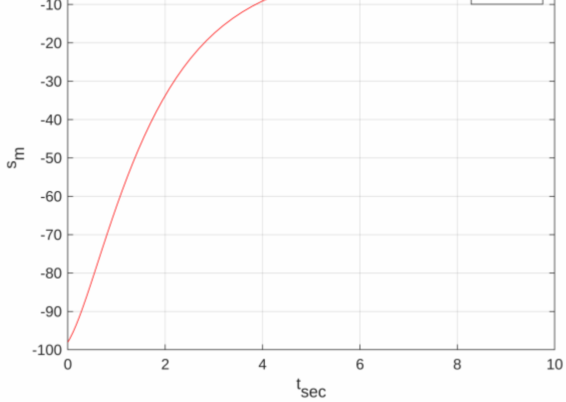

```
plot(t,x(:,2),'r')
hold on;
plot(t,x(:,4),'b')
hold on;
legend ('x_{2}','x_{4}');
grid on;
xlabel("t_{sec}")
ylabel("v_{m/s}")
title("Velocity vs Time")
hold off;
exportgraphics(gcf,'default_tuned_x2x4.png','Resolution',1200)
```

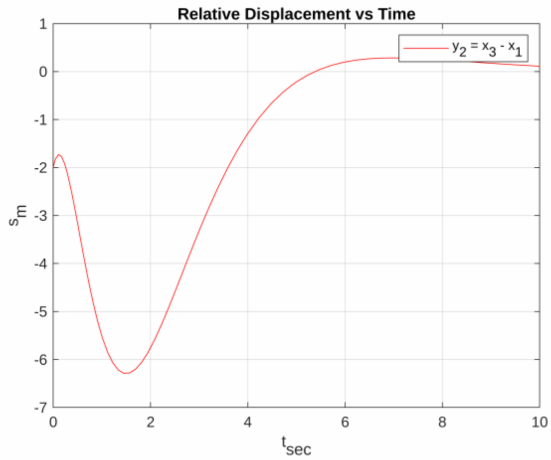

```
disp("Plotting Outputs")
```

```
Plotting Outputs
```

```
plot(t,y(:,1),'r')
legend ('y_{1}');
grid on;
xlabel("t_{sec}")
ylabel("s_{m}")
title("Car 1 Displacement vs Time")
hold off;
exportgraphics(gcf,'default_tuned_y1.png','Resolution',1200)
```

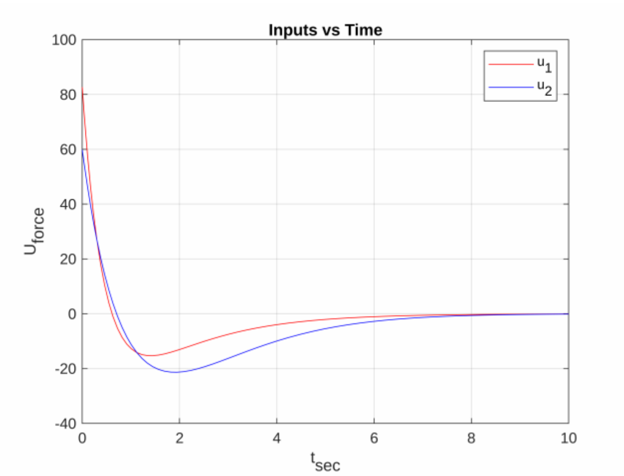

```
plot(t,y(:,2),'r')
legend ('y_{2} = x_{3} - x_{1}');
grid on;
xlabel("t_{sec}")
ylabel("s_{m}")
title("Relative Displacement vs Time")
hold off;
exportgraphics(gcf,'default_tuned_y2.png','Resolution',1200)
```


```
disp("Plotting Inputs")
```

```
Plotting Inputs
```

```
plot(t,inputs(:,1),'r')
hold on;
plot(t,inputs(:,2),'b')
hold on;
legend ('u_{1}','u_{2}');
grid on;
xlabel("t_{sec}")
ylabel("U_{force}")
title("Inputs vs Time")
hold off;
exportgraphics(gcf,'default_tuned_u1u2.png','Resolution',1200)
```



Safety Constraints Check

```
first_violation_time = tspan+1;
for i=1:1:size(t)
    if(y(i,2)>0)
        if(first_violation_time > tspan)
            first_violation_time = t(i)
            [maximum_violation,i] = max(y(:,2));
            maximum_violation
            max_violation_time = t(i)
            break;
        end
    end
end
```

```
first_violation_time = 5.4719
maximum_violation = 0.2846
max_violation_time = 6.9719
```

→ Pt. of first violation
→ Max. violation
→ Time corresponding to max violation

System is violated

Tuning LQR for minimum distance Tuning

```
a = 0.0
```

```
a = 0
```

```
b = 1
```



```
b = 1
```

```
first_entry = 1;
warning('off','all')
output = [0 0];
while(min(output(:,2)) < -2 || max(output(:,2))>0 || first_entry)
    first_entry=0;
    syms x1(t) x2(t) x3(t) x4(t) 'real';
    x = [x1;x2;x3;x4];
    a = a+0.05;
    Q = diag([2.5, 10, a, b-a]);
    [P,K,L] = icare(A,B,Q,R);
    u1 = - K(1,:)*x;
    u2 = - K(2,:)*x;
    dx = [x2; u1; x4 ; u2/2];
    vars = [x1(t) x2(t) x3(t) x4(t)];
    func_dx = odeFunction(dx,vars);
    fdx = @(t,x)func_dx(t,x);
    [t,x] = ode45(fdx,[0,tspan],initial_conditions);
    length = size(t);
    length = length(1,1);
    output = zeros(length,2);
    for i=1:1:size(t)
        output(i,:) = (C*x(i,:)')';
    end
    maximum = max(output(:,2));
    minimum = min(output(:,2));
    plot(t,output(:,2),'black')
    hold on;
    grid on;
    if(maximum > 0)
        disp('Failure at')
        b
        b = b+1;
        a = 0;
    end
end
```

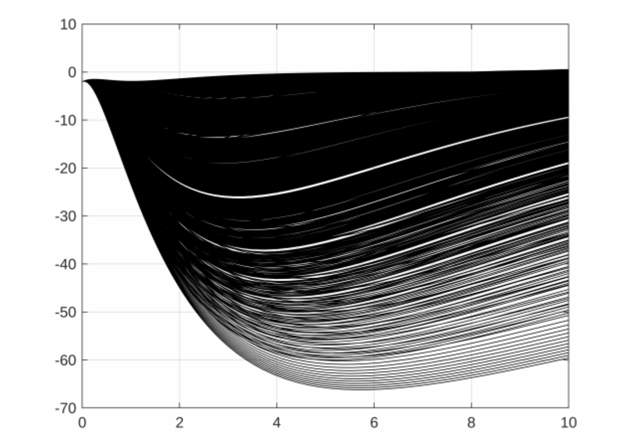
```
Failure at
b = 1
Failure at
b = 2
Failure at
b = 3
Failure at
b = 4
Failure at
b = 5
Failure at
b = 6
Failure at
```

```
Failure at
b = 7
Failure at
```



```
b = 8
Failure at
b = 9
Failure at
b = 10
Failure at
b = 11
Failure at
b = 12
Failure at
b = 13
Failure at
b = 14
Failure at
b = 15
Failure at
b = 16
Failure at
b = 17
Failure at
b = 18
Failure at
b = 19
Failure at
b = 20
Failure at
b = 21
Failure at
b = 22
Failure at
b = 23
Failure at
b = 24
Failure at
b = 25
Failure at
b = 26
Failure at
b = 27
```

```
exportgraphics(gcf,'tuning_performance.png','Resolution',1600)
hold off;
```


```
warning('on','all')
disp('Congratulation. System is tuned !!')
```

```
Congratulation. System is tuned !!
```

```
a
```

```
a = 5.5500
```

```
b
```

```
b = 28
```

Displaying Performance on new Tuning

```
syms x1(t) x2(t) x3(t) x4(t) 'real';
```

```
Warning: Can only make assumptions on variable names, not 'x1(t)'.
Warning: Can only make assumptions on variable names, not 'x2(t)'.
```

```
Warning: Can only make assumptions on variable names, not 'x2(t)'.
Warning: Can only make assumptions on variable names, not 'x3(t)'.
Warning: Can only make assumptions on variable names, not 'x4(t)'.
```

```
x = [x1;x2;x3;x4];
Q = diag([2.5, 10, a, b-a])
```

```
Q = 4x4
    2.5000         0         0         0
         0   10.0000         0         0
         0         0    5.5500         0
```

⇒ Q matrix


```
         0         0         0   22.4500
```

```
[P,K,L] = icare(A,B,Q,R)
```

```
P = 4x4
    5.7363    1.5811    0.0000    0.0000
    1.5811    3.6280    0.0000    0.0000
    0.0000    0.0000   13.3003    4.7117
    0.0000    0.0000    4.7117   11.2913
K = 2x4
    1.5811    3.6280    0.0000    0.0000
    0.0000    0.0000    2.3558    5.6457
L = 4x1
   -0.5065
   -0.5091
   -2.3137
   -3.1214
```

⇒ P tuned

⇒ K tuned

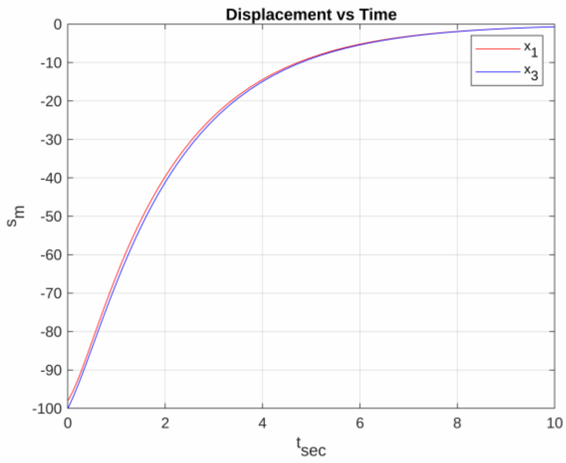
```
u1 = - K(1,:)*x;
u2 = - K(2,:)*x;
dx = [x2; u1; x4 ; u2/2];
vars = [x1(t) x2(t) x3(t) x4(t)];
func_dx = odeFunction(dx,vars);
fdx = @(t,x)func_dx(t,x);
[t,x] = ode45(fdx,[0,tspan],initial_conditions);
length = size(t);
length = length(1,1);
output = zeros(length,2);
for i=1:1:size(t)
    output(i,:) = (C*x(i,:)')';
    inputs(i,:) = (-K*x(i,:)')';
end
```

Plotting Graphs

```
disp("Plotting States")
```

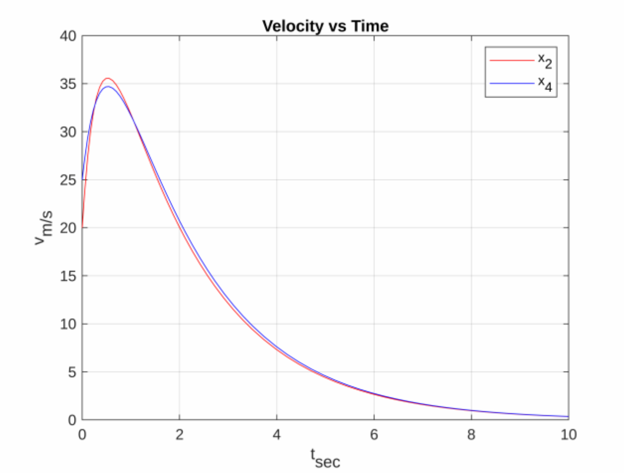
```
Plotting States
```

```
plot(t,x(:,1),'r')
hold on;
plot(t,x(:,3),'b')
hold on;
legend ('x_{1}','x_{3}');
grid on;
xlabel("t_{sec}")
ylabel("s_{m}")
title("Displacement vs Time")
hold off;
exportgraphics(gcf,'custom_tuned_x1x3.png','Resolution',1200)
```


```
plot(t,x(:,2),'r')
hold on;
plot(t,x(:,4),'b')
hold on;
legend ('x_{2}','x_{4}');
grid on;
xlabel("t_{sec}")
ylabel("v_{m/s}")
title("Velocity vs Time")
```

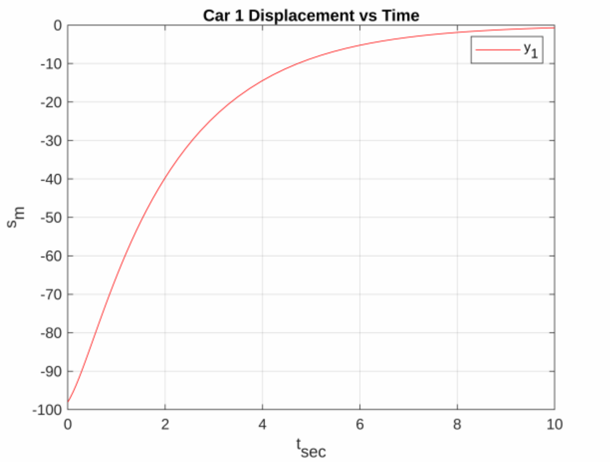
```
hold off;
exportgraphics(gcf,'custom_tuned_x2x4.png','Resolution',1200)
```


```
disp("Plotting Outputs")
```

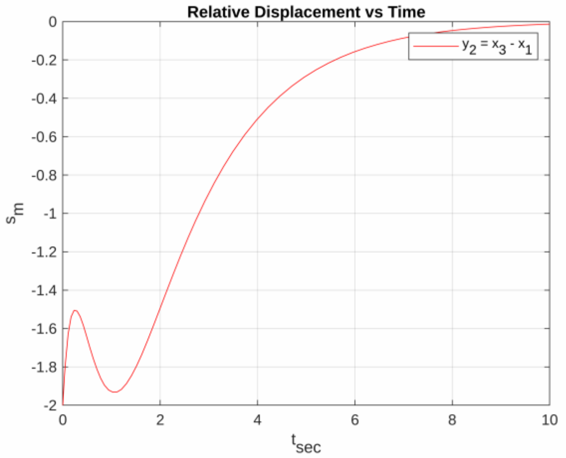
```
Plotting Outputs
```

```
plot(t,output(:,1),'r')
legend ('y_{1}');
grid on;
xlabel("t_{sec}")
ylabel("s_{m}")
title("Car 1 Displacement vs Time")
hold off;
exportgraphics(gcf,'custom_tuned_y1.png','Resolution',1200)
```


```
plot(t,output(:,2),'r')
legend ('y_{2} = x_{3} - x_{1}');
```

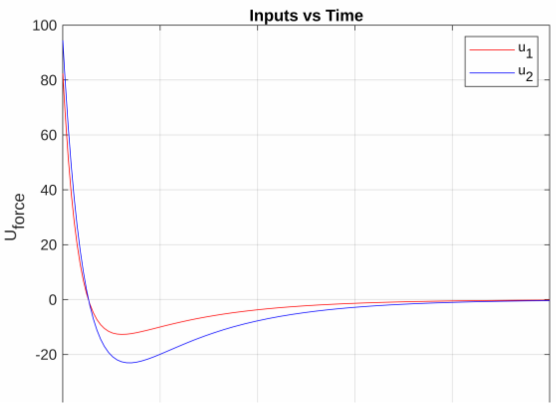
```
grid on;
xlabel("t_{sec}")
ylabel("s_{m}")
title("Relative Displacement vs Time")
hold off;
exportgraphics(gcf,'custom_tuned_y2.png','Resolution',1200)
```

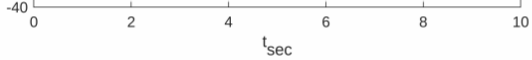




```
disp("Plotting Inputs")
```

```
Plotting Inputs
```

```
plot(t,inputs(:,1),'r')
hold on;
plot(t,inputs(:,2),'b')
hold on;
legend ('u_{1}','u_{2}');
grid on;
xlabel("t_{sec}")
ylabel("U_{force}")
title("Inputs vs Time")
hold off;
```




```
exportgraphics(gcf,'custom_tuned_u1u2.png','Resolution',1200)
```


(e.) System takes longer time to settle down if control effort is low. Inversly, it needs emense control i/p if we need relatively faster settling system.