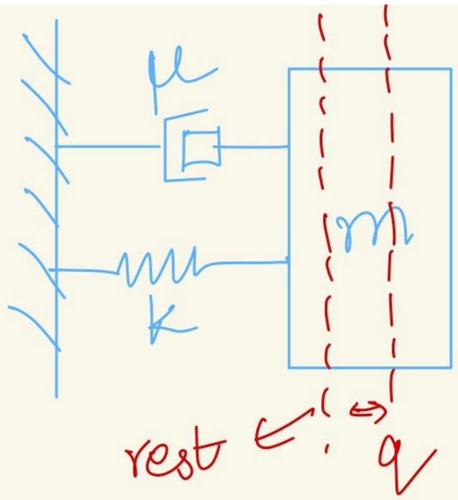
**Fall 2024 – 16-642 Manipulation, Estimation, and Control**
**Problem Set 1**

Ankit Aggarwal (ankitagg)

## Question 1

1




$$F_s = -kq$$
$$F_d = -\mu \dot{q}$$
$$F = u$$

$$F + F_s + F_d = ma$$

(a) $\Rightarrow m\ddot{q} + \mu\dot{q} + kq = u$

$$\text{Let } x = \begin{bmatrix} q \\ \dot{q} \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\Rightarrow \dot{x} = \begin{bmatrix} \dot{q} \\ \ddot{q} \end{bmatrix} = \begin{bmatrix} x_2 \\ \dfrac{u - kq - \mu\dot{q}}{m} \end{bmatrix}$$

$$\Rightarrow \dot{x} = \begin{bmatrix} x_2 \\ \dfrac{u}{m} - \dfrac{k}{m}q - \dfrac{\mu}{m}\dot{q} \end{bmatrix} = f(x,u)$$

(b) $\Rightarrow$ $$\boxed{\dot{x} = \underbrace{\begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{\mu}{m} \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_{x} + \underbrace{\begin{bmatrix} 0 \\ 1/m \end{bmatrix}}_{B} \underbrace{u}_{u}}$$

Given that $y = q$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u$$

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

(c) Stability criteria

$$A = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{\mu}{m} \end{bmatrix} \quad \det(A) \neq 0$$

A is invertible, hence $x_e = 0$ is a unique equilibrium point.

Computing eigen values of $A$, $\det(\lambda I - A) = 0$, solve for $\lambda$.

$$\lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{\mu}{m} \end{bmatrix} = 0$$

$$\Rightarrow \begin{bmatrix} \lambda & -1 \\ \frac{k}{m} & \lambda + \frac{\mu}{m} \end{bmatrix} = 0$$

$$\Rightarrow \lambda\left(\lambda + \frac{\mu}{m}\right) + \frac{k}{m} = 0$$

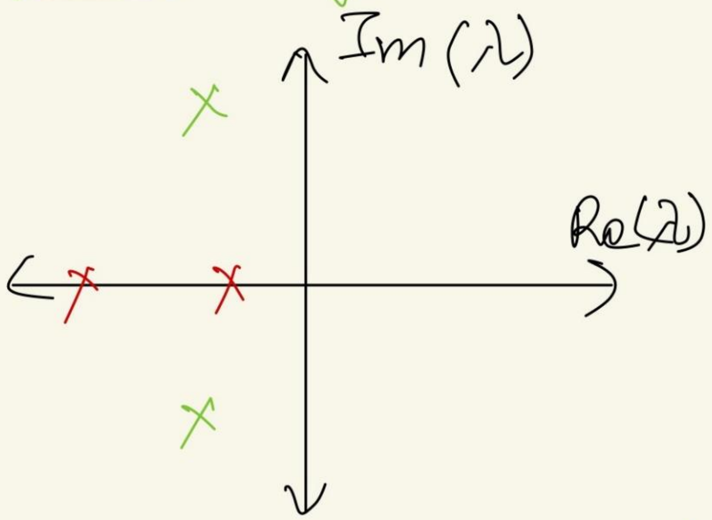
$$\Rightarrow \lambda^2 m + \lambda\mu + k = 0$$

$$\Rightarrow \lambda = \frac{\mu \pm \sqrt{\mu^2 - 4mk}}{2m}$$

Since this is a real system,

$\mu > 0$, $m > 0$, $k > 0$.

Case 1: If $\mu^2 - 4mk < 0$




For stability, $Re(\lambda) < 0$

$$\Rightarrow -\frac{\mu}{2m} < 0 \Rightarrow \frac{\mu}{2m} > 0$$

Case 2: If $\mu^2 - 4mk \geq 0$

Since $\lambda \in \mathbb{R}$,

$\lambda < 0 \Rightarrow$ stable

$$\frac{-\mu \pm \sqrt{\mu^2 - 4mk}}{2m} < 0$$

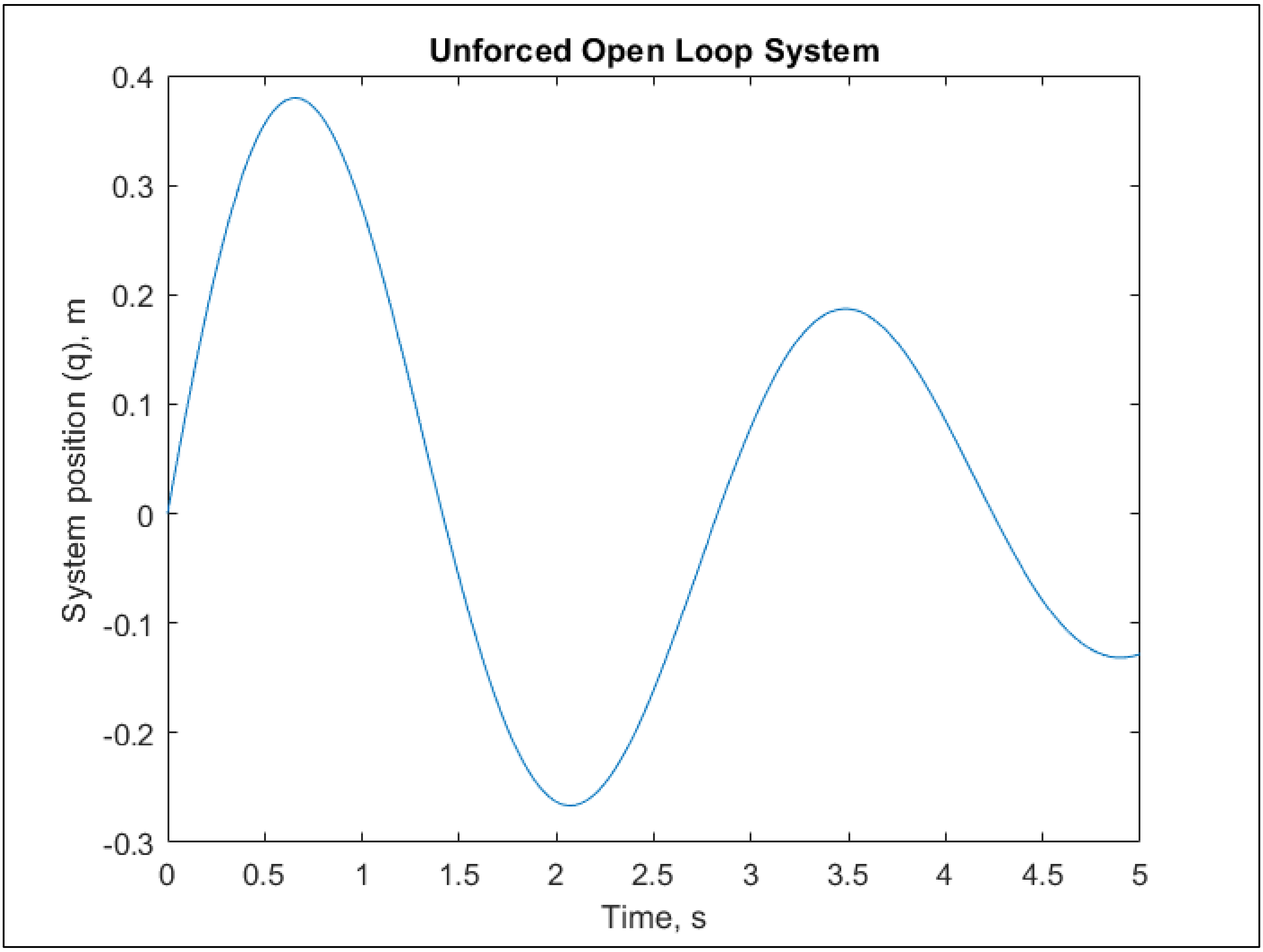
(d) Controllability

The system is controllable if the System matrix A is of full rank.

Since A is invertible, the rank of matrix A is 2 and therefore, full.

$$\boxed{\text{rank}(A) = 2}$$

(e) Plotting the output of the unforced system for t = [0,5]




(f) To find K, I used the following code:

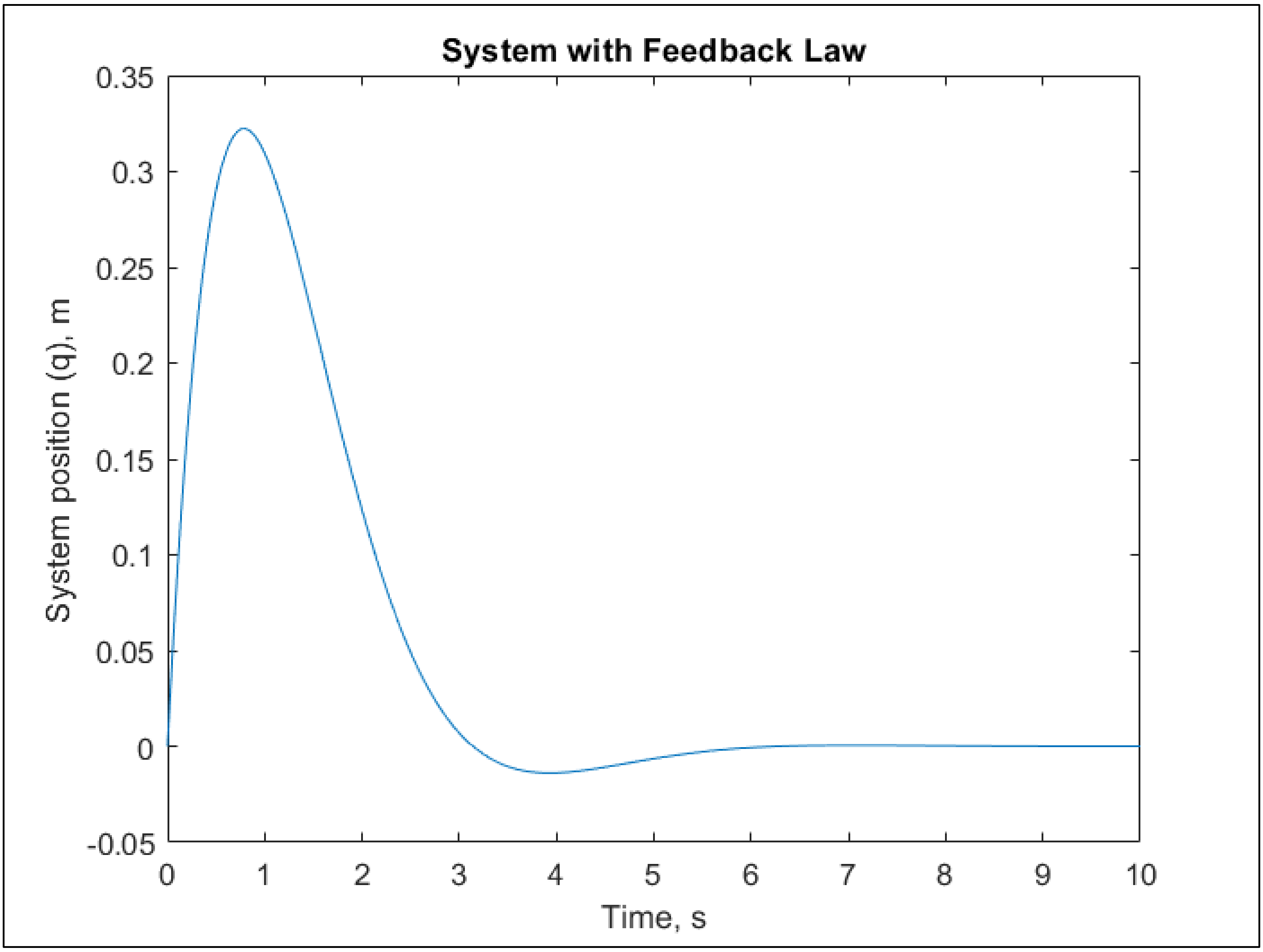
```
p = [-1 + 1i, -1 - 1i];
K = place(A,B,p);
```

The value of K is [-3, 1.5].

When all eigen values $\lambda$ are such that $Re(\lambda) < 0$. The system is asymptotically stable. This ensures that regardless of initial conditions, the system will converge to the equilibrium point.

(g) plotting the output of the system under the feedback law

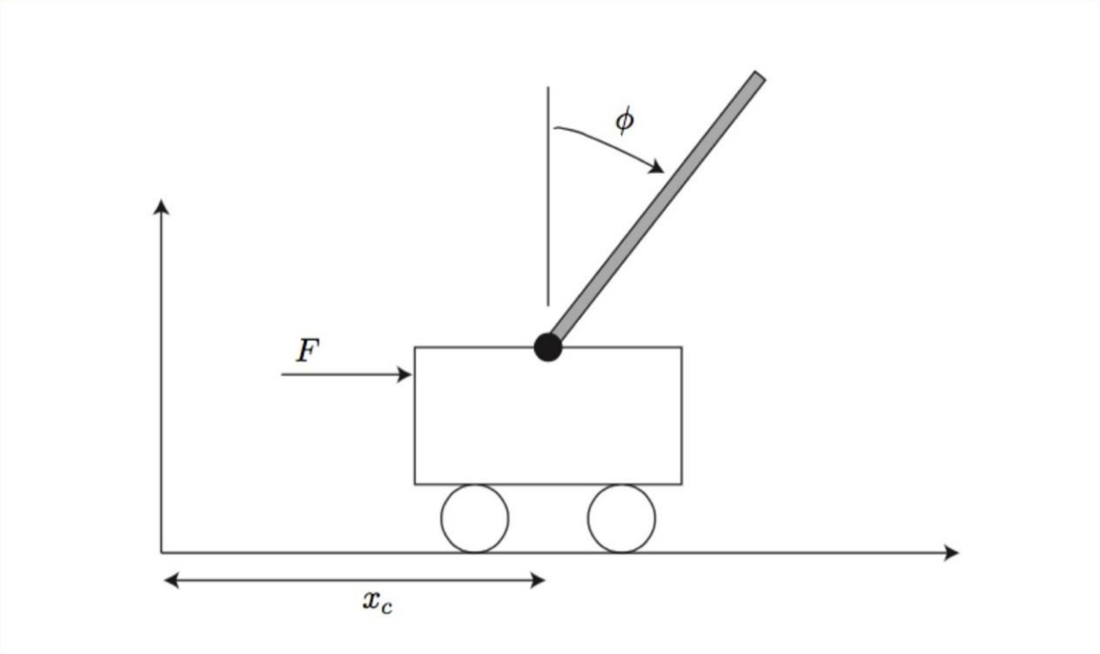
$u(t) = -Kx(t)$ for $t \in [0, 10]$.


System with Feedback Law
System position (q), m
Time, s
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-0.05
0
1
2
3
4
5
6
7
8
9
10

**Question 2**

(a)

2




$$\gamma \ddot{x}_c - \beta \ddot{\phi} \cos\phi + \beta \dot{\phi}^2 \sin\phi + \mu \dot{x}_c = F \quad \text{— (1)}$$

$$\alpha \ddot{\phi} - \beta \ddot{x}_c \cos\phi - D \sin\phi = 0, \quad \text{— (2)}$$

(a) $x = \begin{bmatrix} x_c \\ \phi \\ \dot{x}_c \\ \dot{\phi} \end{bmatrix}$ ; $u = F$

$$M = \begin{bmatrix} \gamma & -\beta\cos\phi \\ -\beta\cos\phi & \alpha \end{bmatrix}$$

$$N = M \begin{bmatrix} \ddot{x}_c \\ \ddot{\phi} \end{bmatrix}$$

$$N = \begin{bmatrix} \gamma \ddot{x}_c - \ddot{\phi}\beta\cos\phi \\ -\ddot{x}_c\beta\cos\phi + \alpha\ddot{\phi} \end{bmatrix}$$

$$N = \begin{bmatrix} u - \beta\dot{\phi}^2\sin\phi - \mu\dot{x}_c \\ D\sin\phi \end{bmatrix}$$

Since,

$$N = M \begin{bmatrix} \ddot{x}_c \\ \ddot{\phi} \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} \ddot{x}_c \\ \ddot{\phi} \end{bmatrix} = [M^{-1}][N]$$

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad-bc}\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

$$[M^{-1}] = \underbrace{\frac{1}{\gamma\alpha - \beta^2\cos^2\phi}}_{K}\begin{bmatrix} \alpha & \beta\cos\phi \\ \beta\cos\phi & \gamma \end{bmatrix}$$

$$[M^{-1}][N] = K\begin{bmatrix} \alpha & \beta\cos\phi \\ \beta\cos\phi & \gamma \end{bmatrix}\begin{bmatrix} u - \beta\dot{\phi}^2\sin\phi - \mu\dot{x}_c \\ D\sin\phi \end{bmatrix}$$

$$= K\begin{bmatrix} \alpha\left[u - \beta\dot{\phi}^2\sin\phi - \mu\dot{x}_c\right] + D\sin\phi\,\beta\cos\phi \\ \beta\cos\phi\left[u - \beta\dot{\phi}^2\sin\phi - \mu\dot{x}_c\right] + \gamma D\sin\phi \end{bmatrix}$$

$$\ddot{x}_c = K\left[(\alpha\beta\sin\phi\,\dot{\phi})\,\underline{\dot{\phi}} + (-\mu\alpha)\,\underline{\dot{x}_c} \right.$$
$$\left. \alpha\,\underline{u} + D\beta\sin\phi\cos\phi\right]$$

$$\ddot{\phi}_c = K\left[(-\beta^2\sin\phi\cos\phi\,\dot{\phi})\,\underline{\dot{\phi}} + \right.$$
$$-(\mu\beta\cos\phi)\,\underline{\dot{x}_c} + \beta\cos\phi\,\underline{u}$$
$$\left. + \gamma D\sin\phi\right]$$

---

$$\dot{x} = \begin{bmatrix} \dot{x}_c \\ \dot{\phi} \\ \ddot{x}_c \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} x_3 \\ x_4 \\ \ddot{x}_c \\ \ddot{\phi} \end{bmatrix}$$

Forming non-linear state space equation.

(b) To symbolically solve the equation for, I used the following code:

```
syms alpha beta gamma D mu xc phi xc_dot phi_dot F xc_ddot phi_ddot
equation1 = gamma * xc_ddot - beta * phi_ddot * cos(phi) + beta * phi_dot * phi_dot * sin(phi)
+ mu * xc_dot - F == 0;
equation2 = alpha * phi_ddot - beta * xc_ddot * cos(phi) - D * sin(phi) == 0;
S = solve([equation1,equation2],[xc_ddot,phi_ddot]);
```

The solutions were:

```
xc_ddot = (- alpha*sin(phi)*beta*phi_dot^2 + F*alpha - alpha*mu*xc_dot + cos(phi)*sin(phi)*D*beta)
/(alpha*gamma - beta^2*cos(phi)^2)
phi_ddot = (- cos(phi)*sin(phi)*beta^2*phi_dot^2 + F*cos(phi)*beta + sin(phi)*D*gamma -
mu*xc_dot*cos(phi)*beta) / (alpha*gamma - beta^2*cos(phi)^2)
```

Verified the solution with the one derived from hand.

(c) Equilibrium Points

To find equilibrium points,

Consider an unforced system, $u[F]=0$

$u=0, \; \dot{x}=0$

$$\dot{x} = \begin{bmatrix} x_3 \\ x_4 \\ \dot{x}_c \\ \ddot{\phi}_c \end{bmatrix} = 0$$

$$= \begin{bmatrix} x_3 \\ x_4 \\ \frac{1}{r\alpha - \beta^2\cos^2(x_2)}\left(\alpha F - \alpha\beta x_4^2 \sin(x_2) - \alpha\mu x_3 + \beta D \cos x_2 \sin x_2\right) \\ \frac{1}{r\alpha - \beta^2\cos^2(x_2)}\left(\beta F \cos x_2 - \beta^2 x_4^2 \cos x_2 \sin x_2 - \beta\mu x_3 \cos(x_2) + \gamma D \sin x_2\right) \end{bmatrix}$$

$\Rightarrow x_3 = 0, \; x_4 = 0$

$$BD\cos x_2 \sin x_2 = 0$$

$$\gamma D \sin x_2 = 0 \Rightarrow x_2 = n\pi$$

Since there are no equations to $x_1$, $x_1$ can be any real value

$$\therefore x_e = \begin{bmatrix} x_1 \\ n\pi \\ 0 \\ 0 \end{bmatrix} ; \; x_1 \in \mathbb{R}, \; n \in \mathbb{N}$$

Physically, this means that the system is at equilibrium when

i. velocity of the cart and the angular velocity of the pendulum are zero.
ii. angular position of the pendulum is nπ i.e. 0° or 180°
iii. the cart position does not affect equilibrium conditions i.e., the cart can be anywhere and still be at equilibrium.

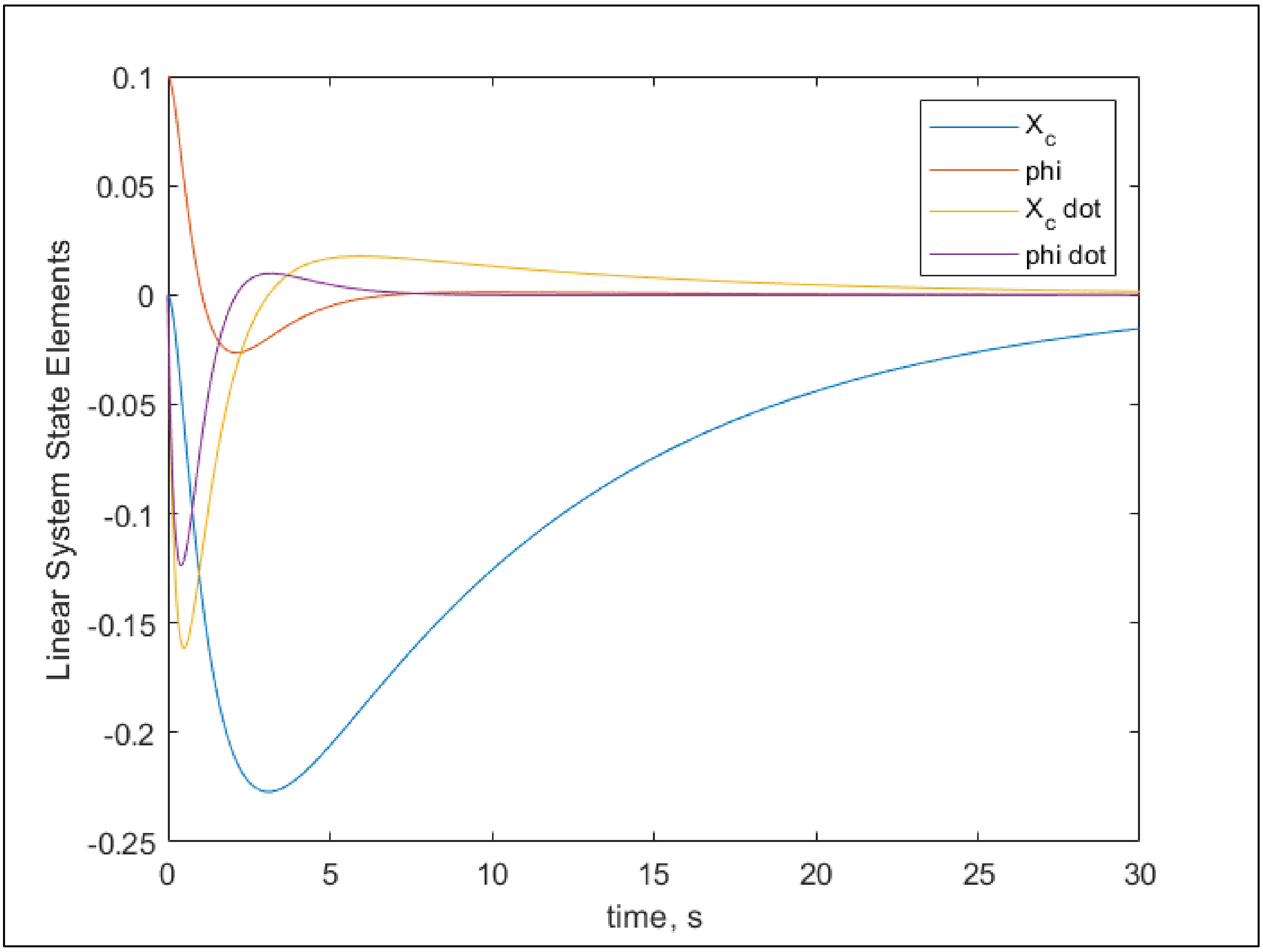
(d) eig(A_linearised) gives the following output

```
      0
  -3.3301
   1.1284
  -0.7984
```

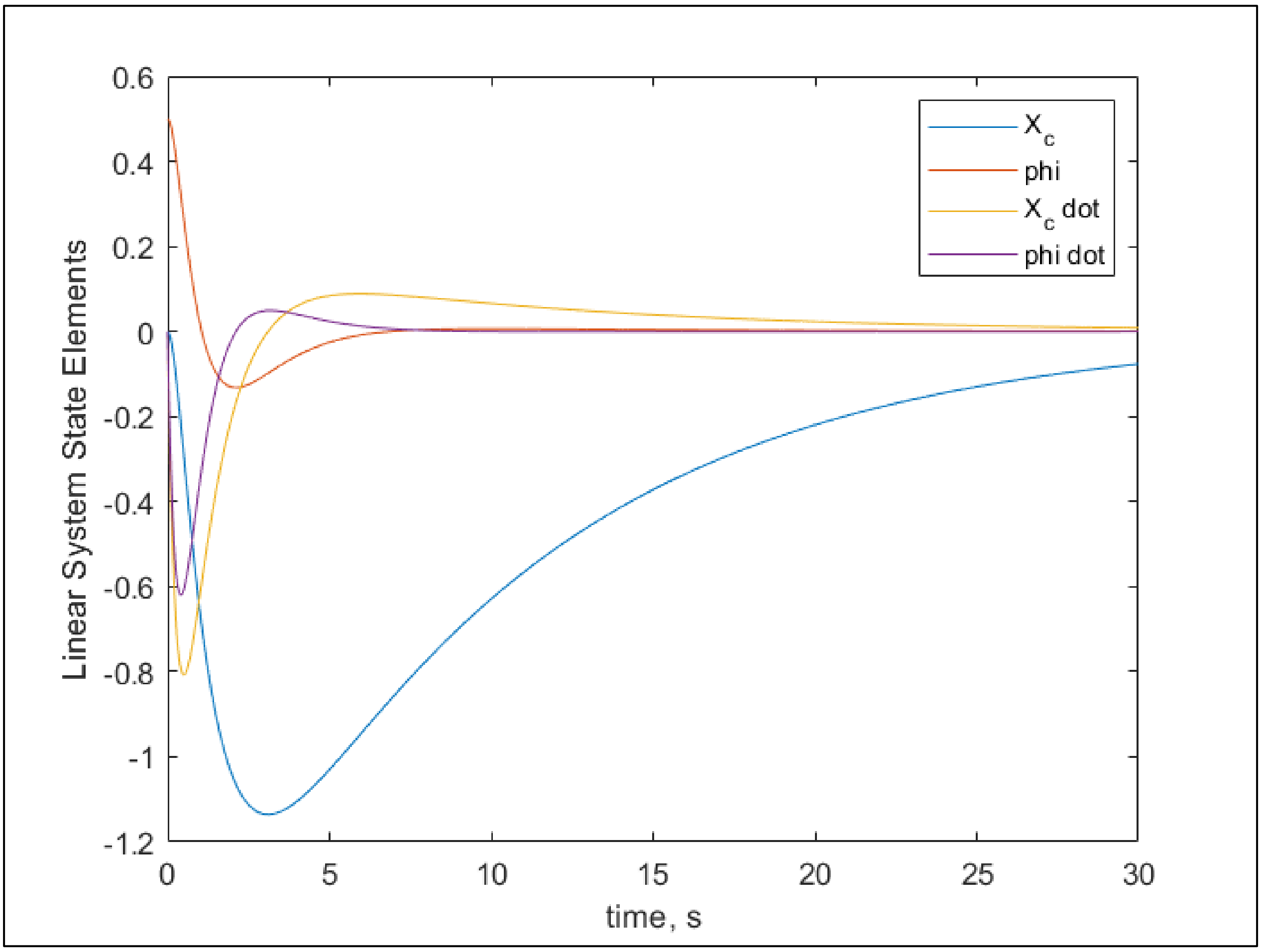
Since one of the eigen values is positive, the linearised system is unstable.
Using this, based on the Lyapunov's First Method, we can also say that the non-linear system is also unstable around the given equilibrium point $x_e = 0$.

(e) The plots for the linear system using each initial state are given as:

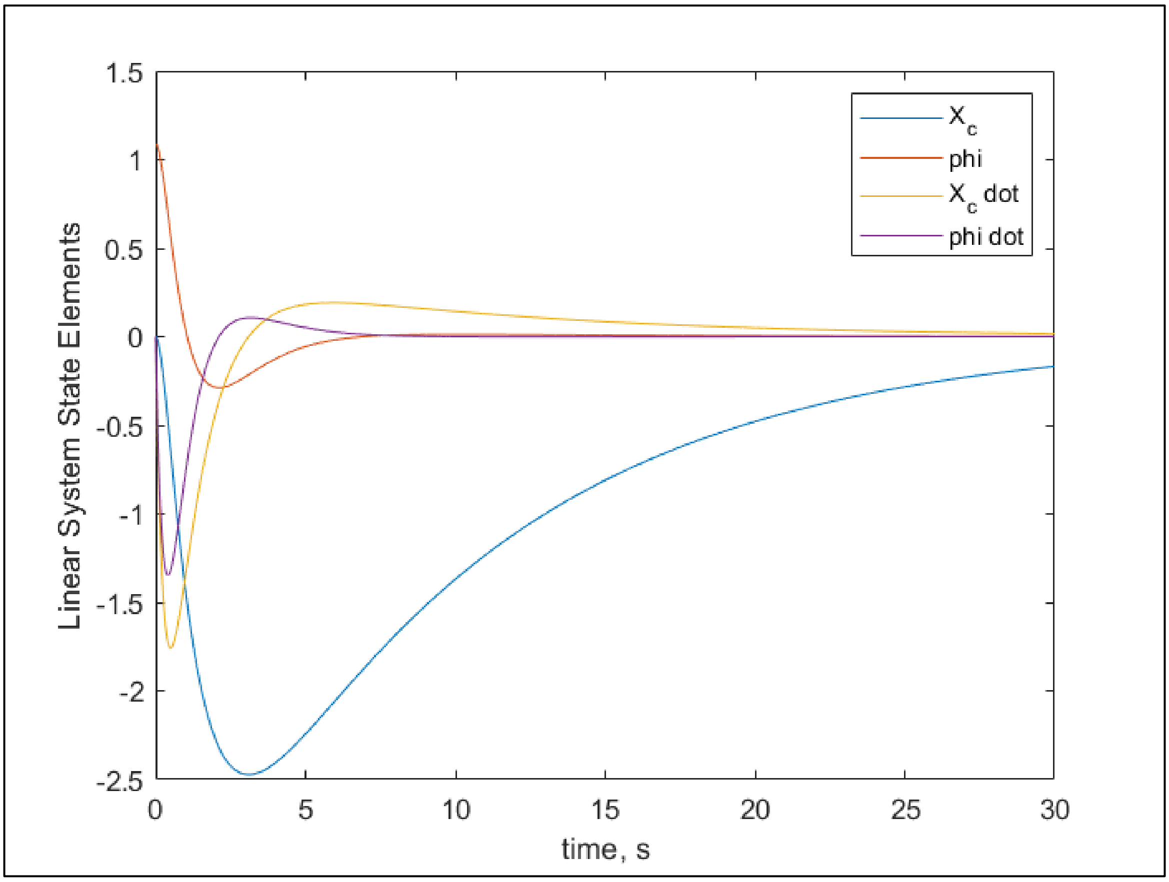
i. x_0 = [0 0.1 0 0]'


0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
Linear System State Elements
0
5
10
15
20
25
30
time, s
X c
phi
X c dot
phi dot

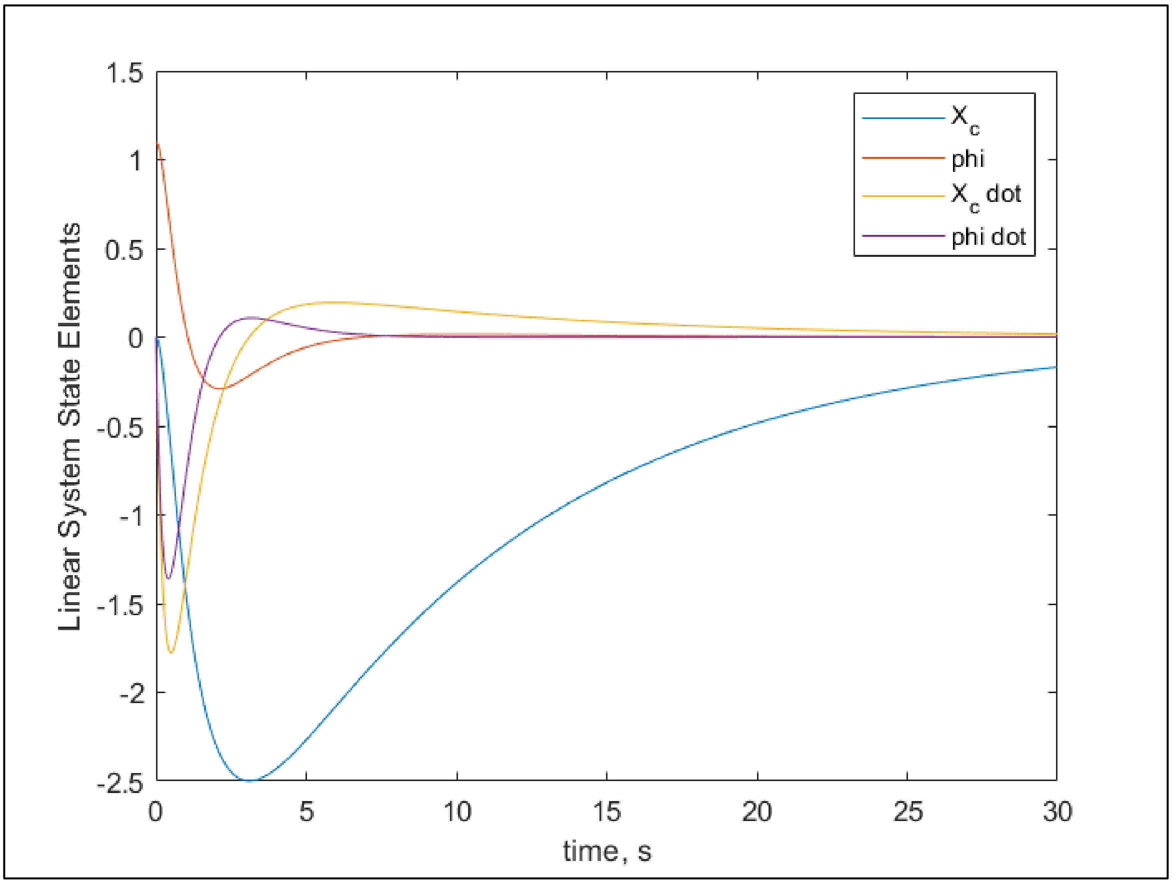
ii. x_0 = [0 0.5 0 0]'


0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1.2
Linear System State Elements
0
5
10
15
20
25
30
time, s
X c
phi
X c dot
phi dot

iii. x_0 = [0 1.0886 0 0]'

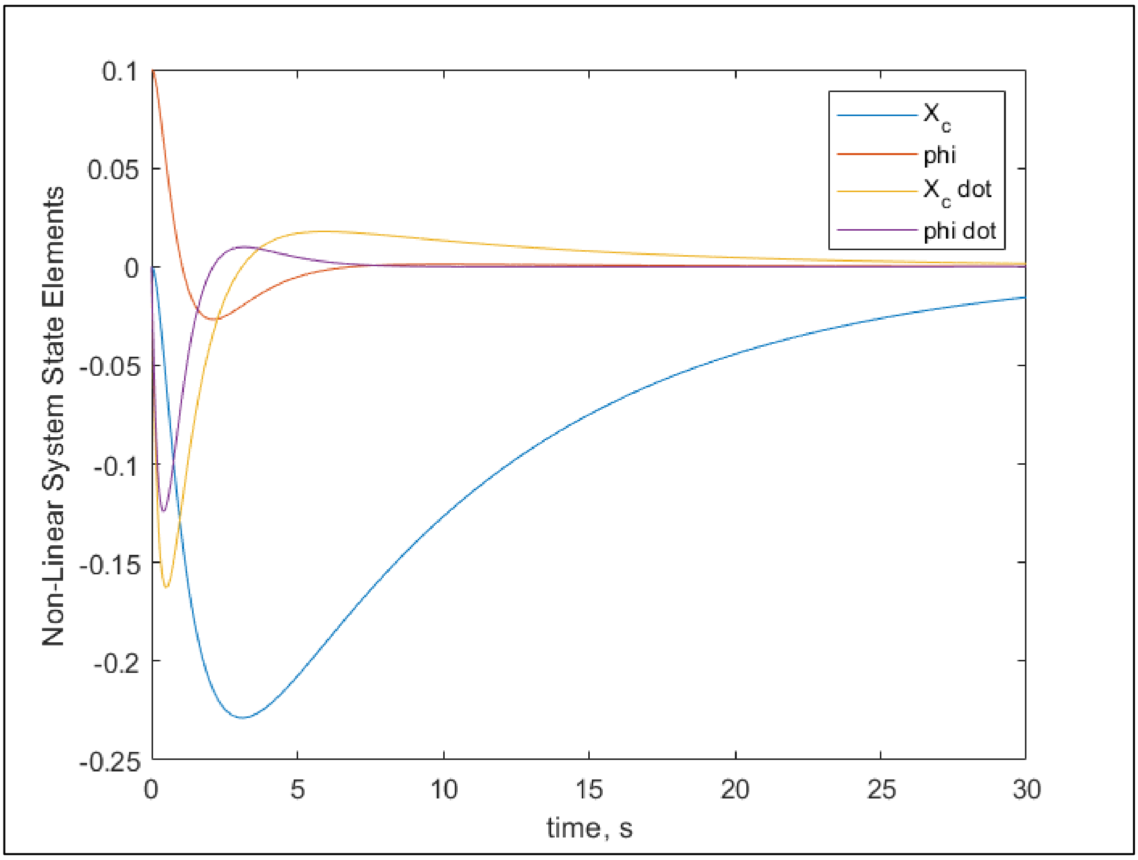



iv. x_0 = [0 1.1 0 0]'

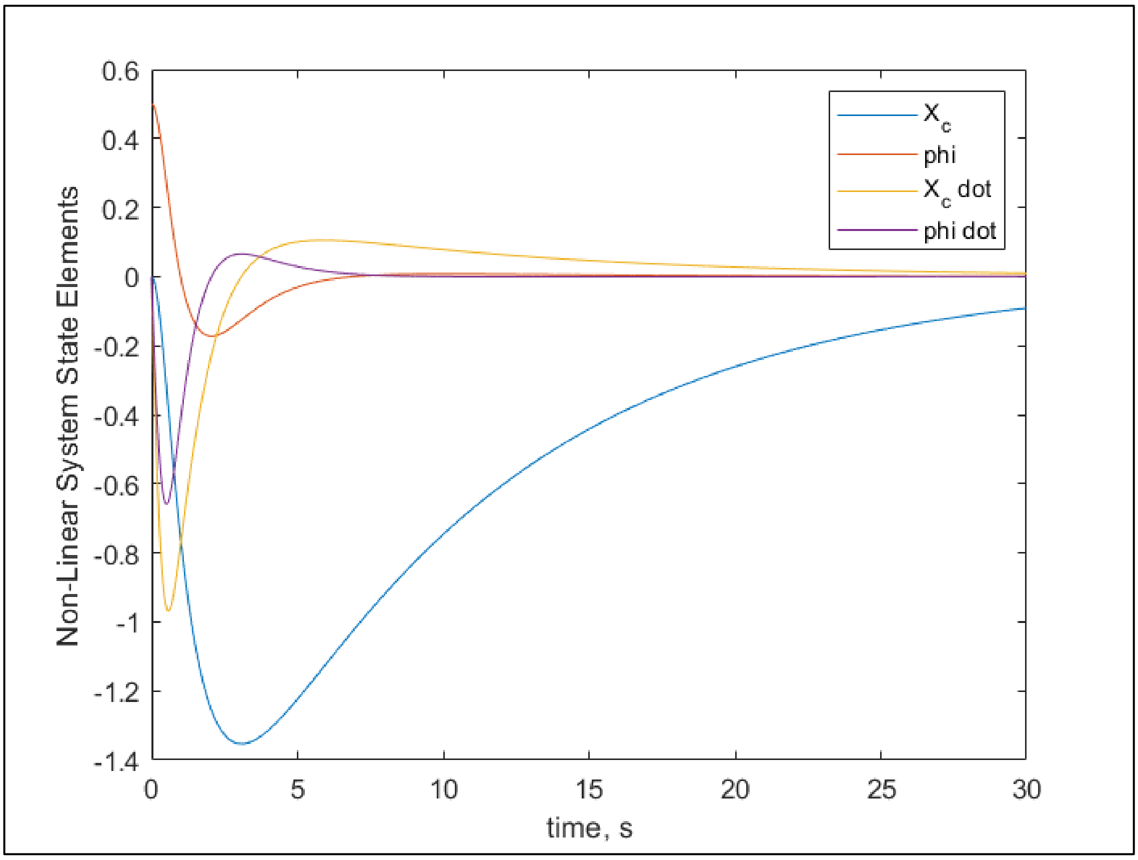


(f) The plots for the non-linear system using each initial state are given as:

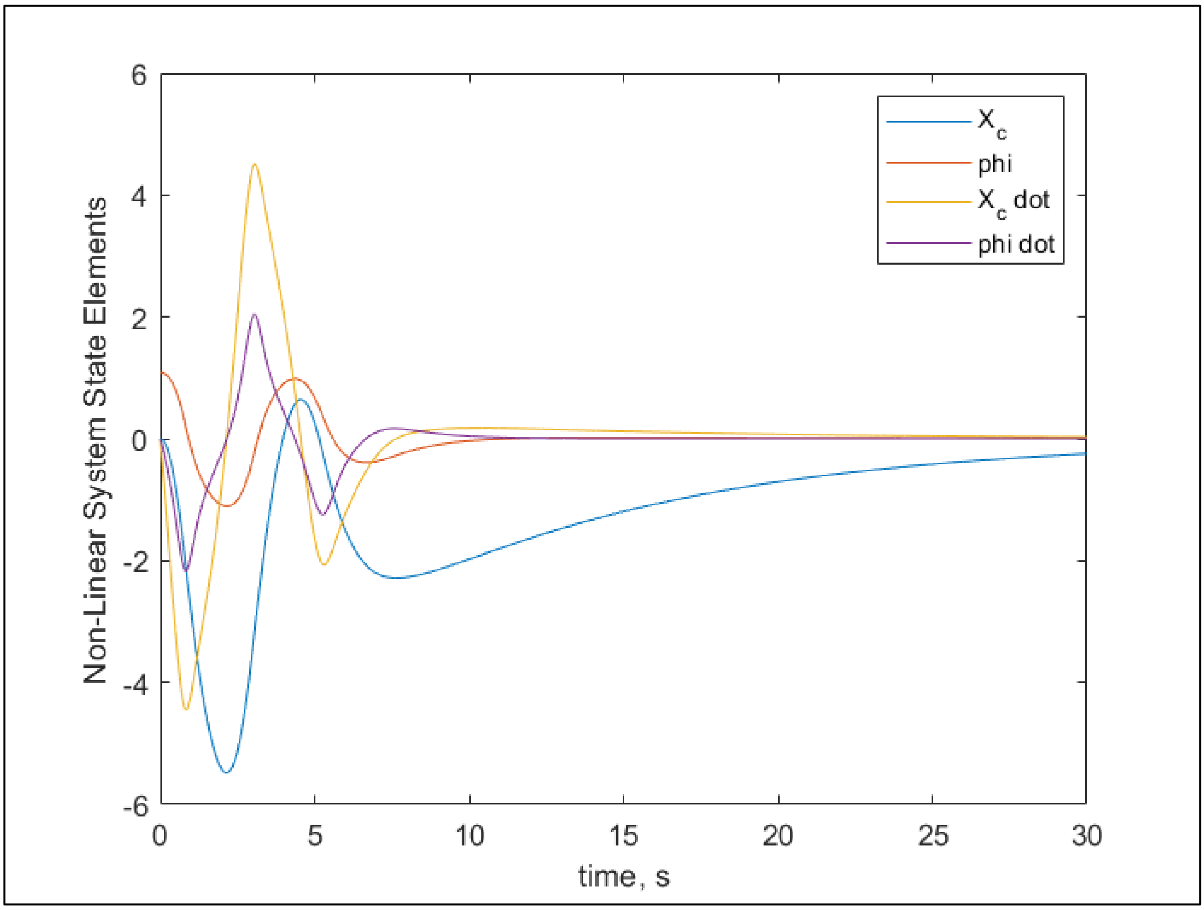
i. x_0 = [0 0.1 0 0]'



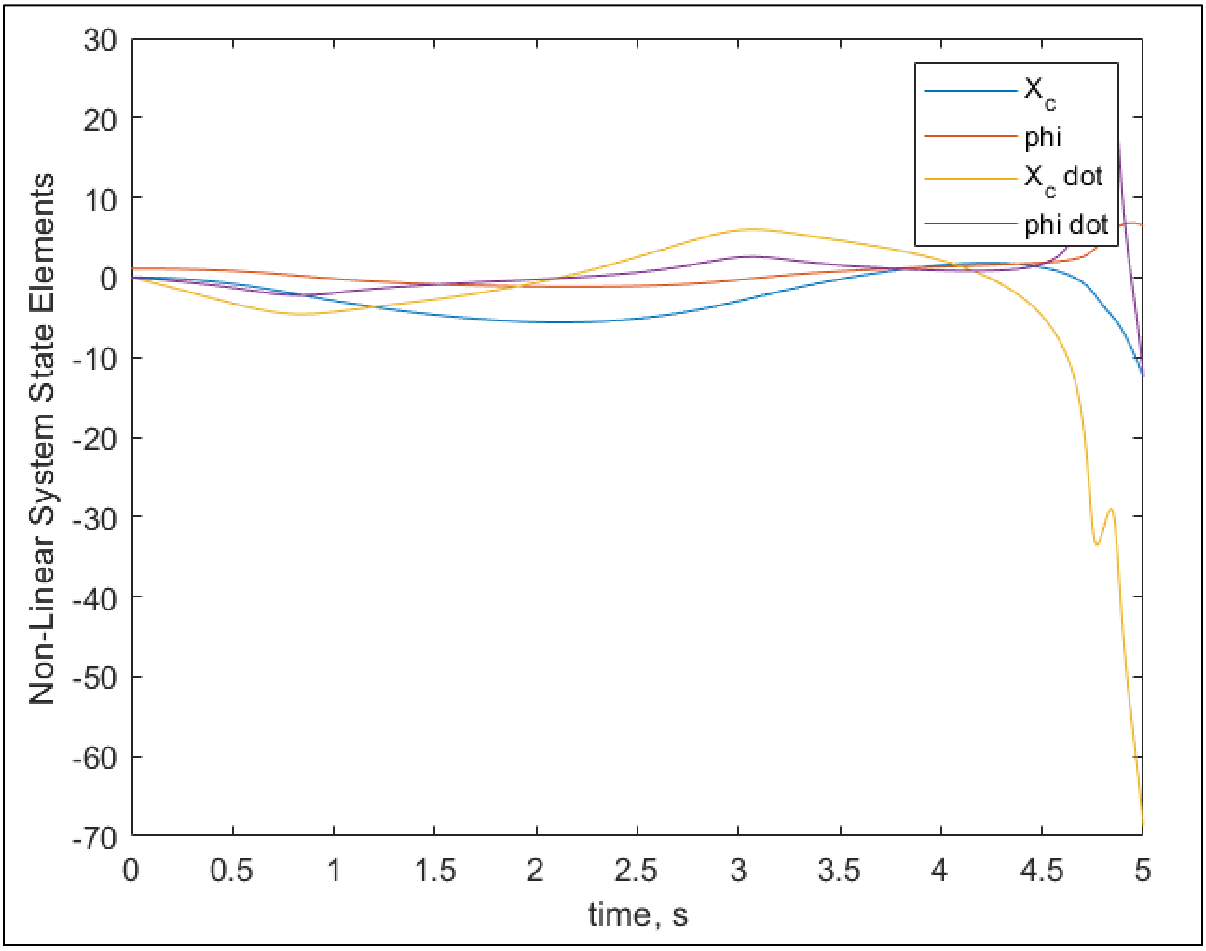

ii. x_0 = [0 0.5 0 0]



iii. x_0 = [0 1.0886 0 0]'




iv. x_0 = [0 1.1 0 0]'



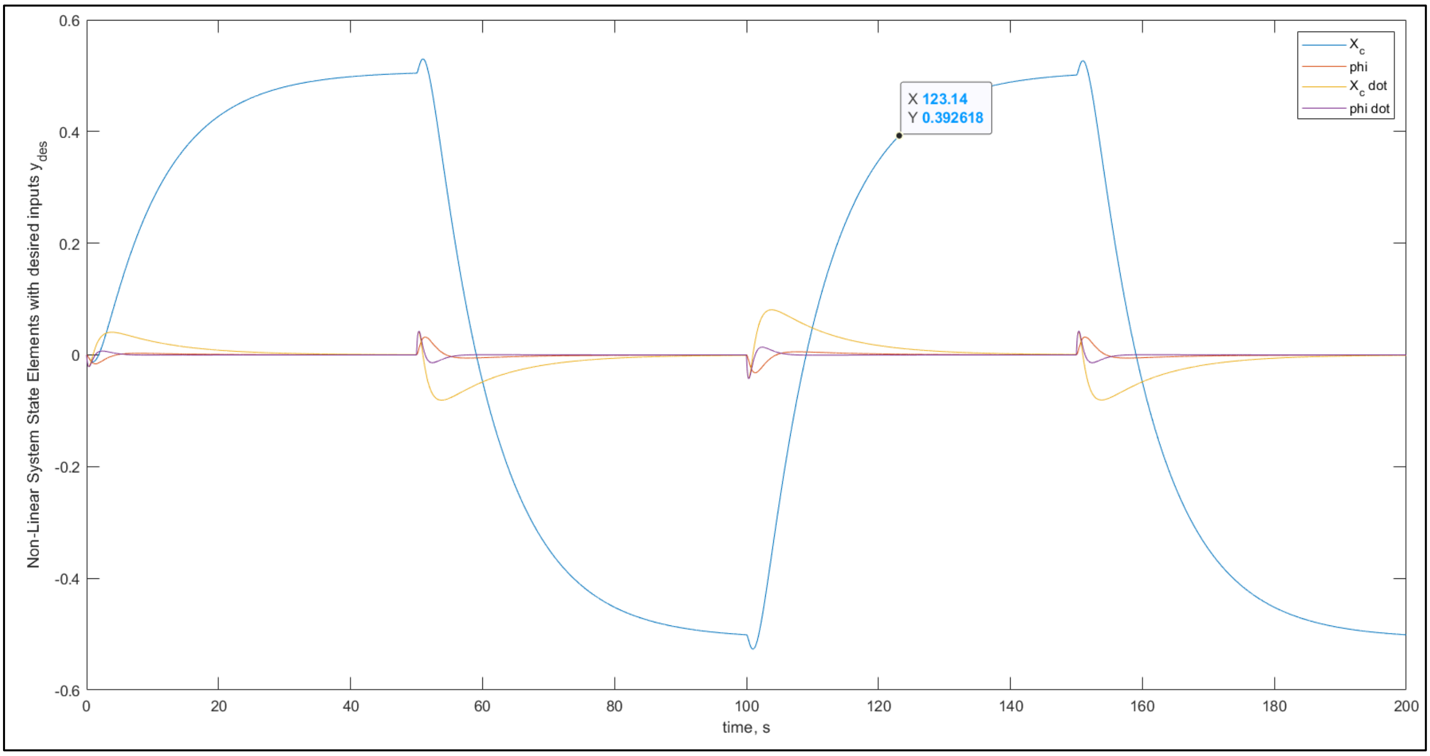
The non-linear system diverges when the initial angle phi is given as 1.1 radians while the linear system does not. This is because when a linearised system is globally asymptotically stable, the non-linear system is only locally asymptotically stable. Setting phi as 1.1 goes beyond the locally stable region, causing the system to diverge.

(g) To track position, the C matrix should extract position from X which is the first element of X.
As X is 4x1 matrix and output y is a scalar quantity. Matrix C needs to be a 1x4 matrix to achieve the right shapes.
To convert from metres to inches, the scalar value must be multiplied by 39.37008.
Hence,

$$C = [39.37008\ 0\ 0\ 0]$$

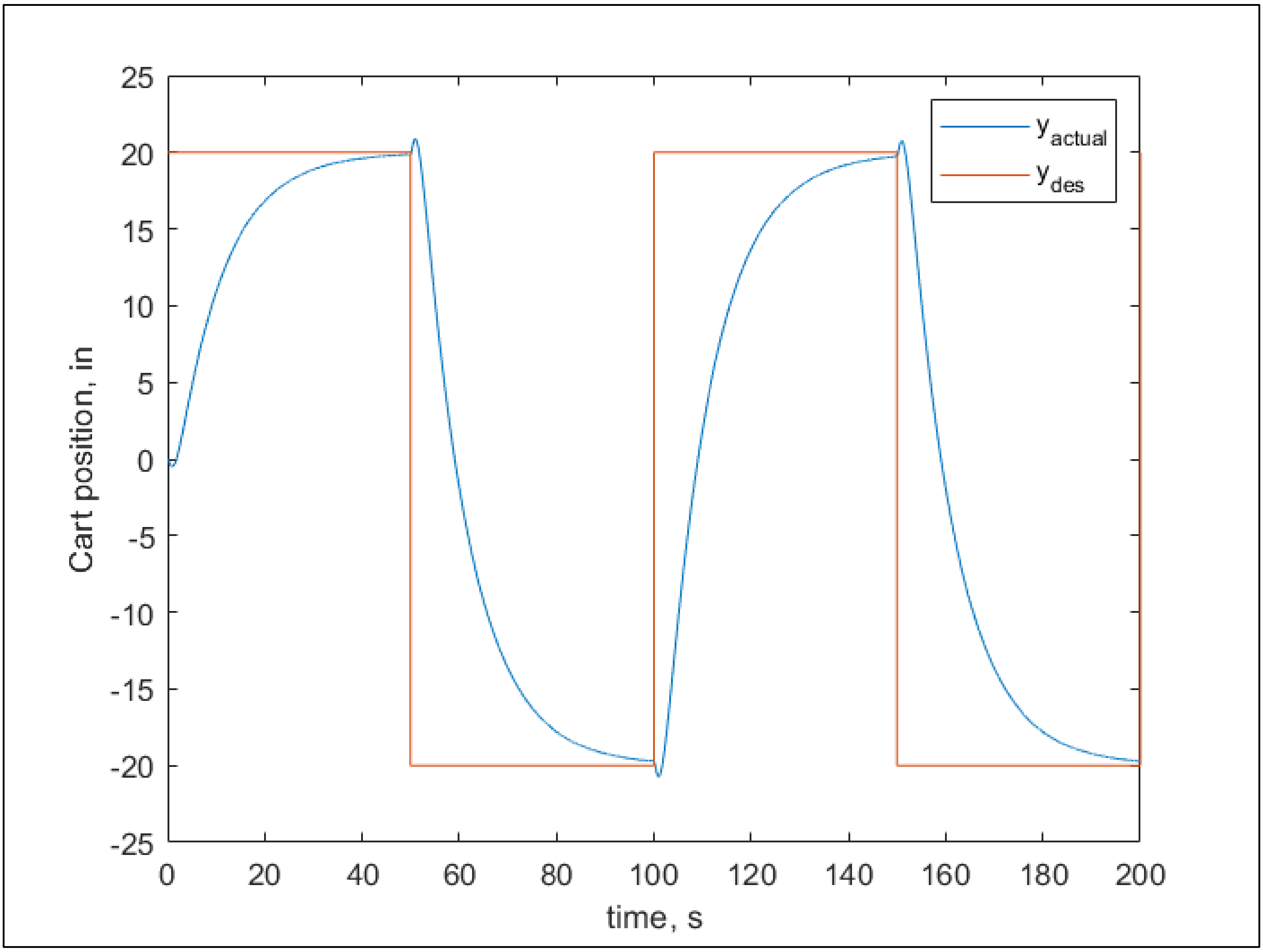
(h) To create the tracking controller, I calculated the feedforward gain $K_f$ and computed the new input value as $F = K_f * y_{des} - K * x$

Using the given Qu and Qx values, the following plots were seen:




It is evident that the system position $X_c$ is using the $y_{des}$ square wave input and trying to converge to the defined trajectory.

The value of $X_c$ in this graph is in metres. I used the C matrix to convert to the required inches unit and plotted the graph below.




This graph shows that the cart position is following the given input. However, there is error which needs to be corrected with tuning $Q_x$ and $Q_u$ parameters.

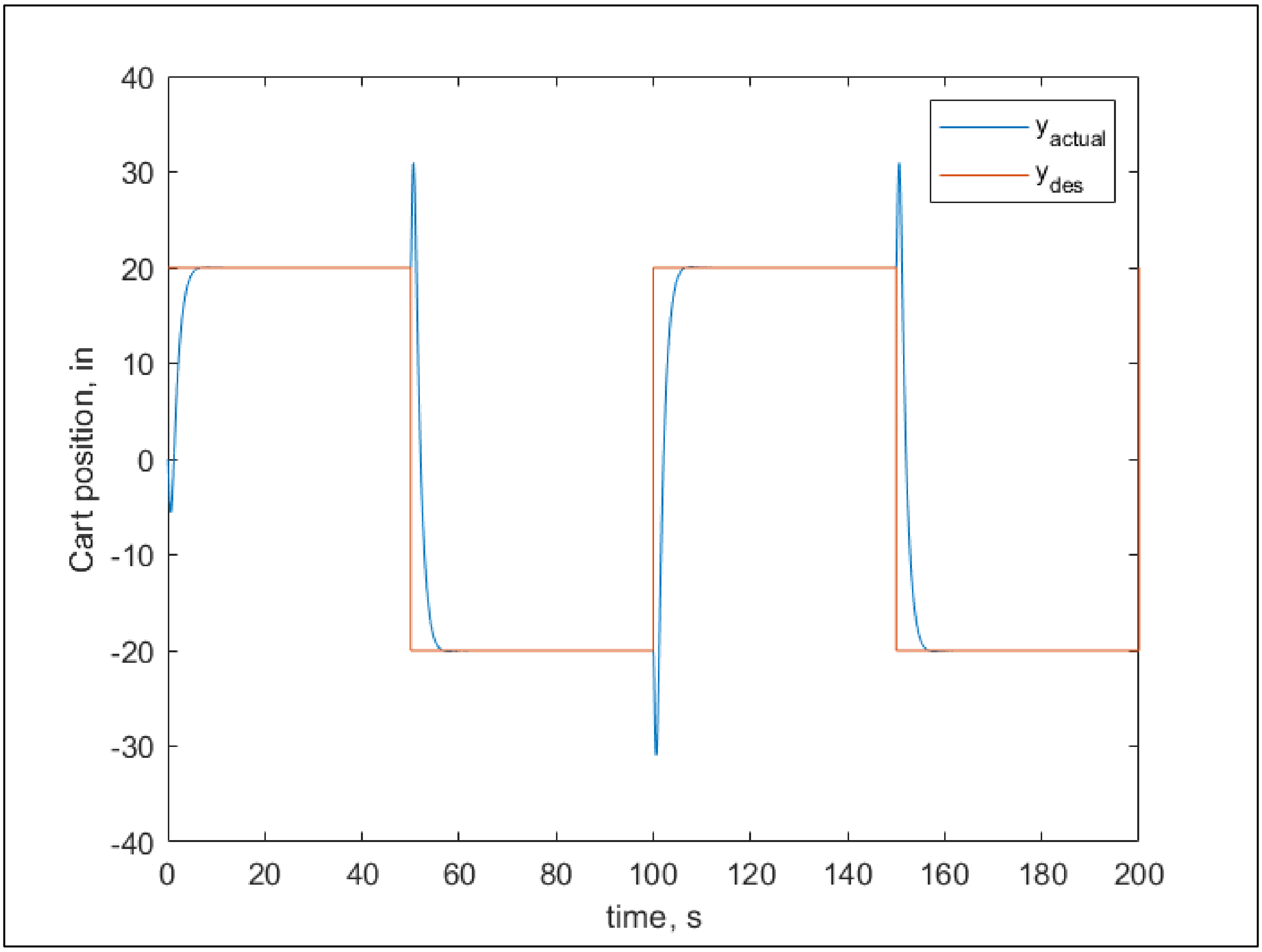
(i) There are two distinct errors in this system:

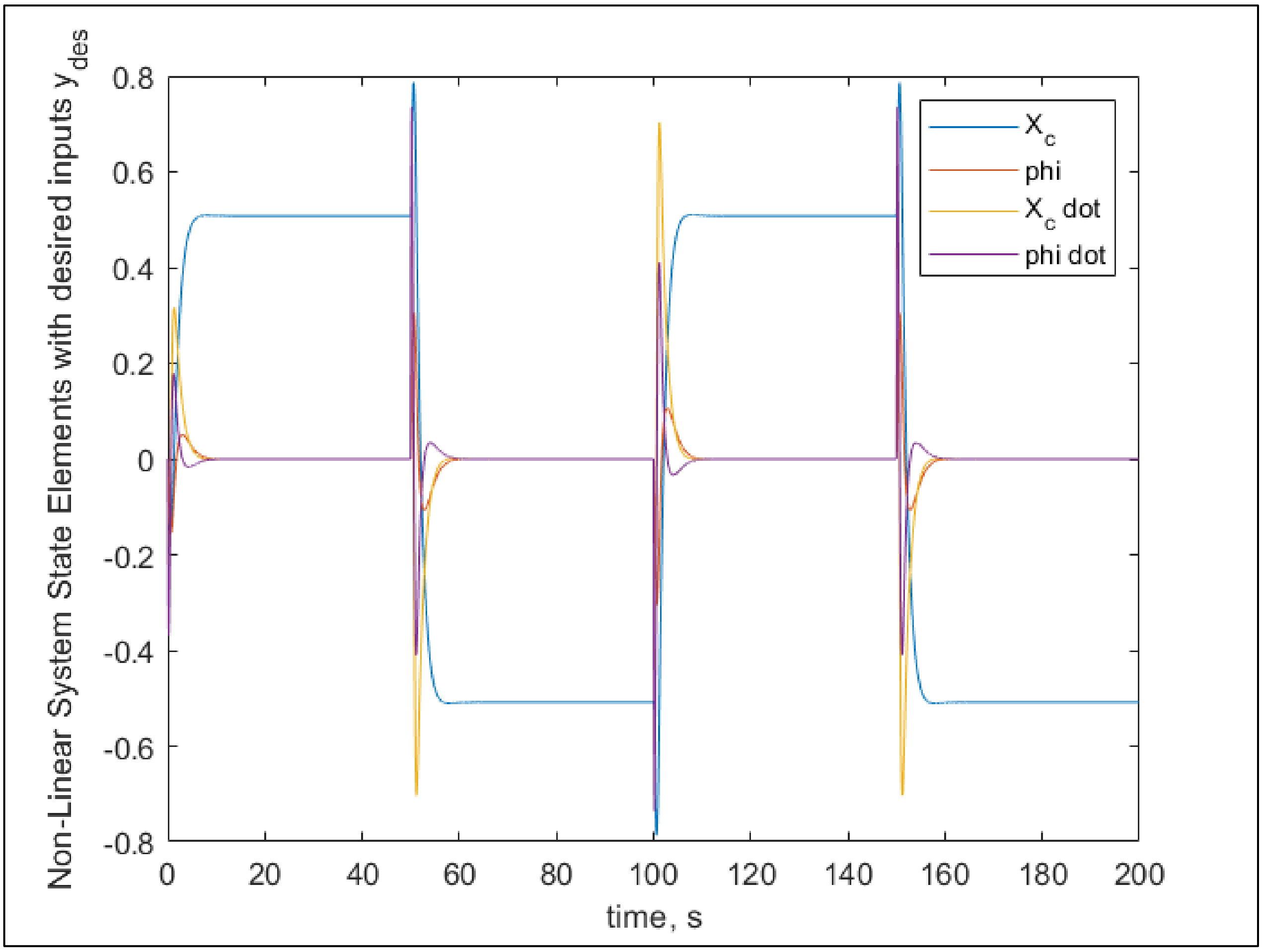
- Convergence time
- Overshoots at input change

However, these seem to be inversely related i.e., when convergence time improves, the peaks increase and vice versa.
The elements of $Q_x$ help improve convergence time and the value of $Q_u$ affects the overshoots.

**Iteration 1:**
$Q_x$ = diag (700, 700, 15, 10)
$Q_u$ = 10

Here, the time to converge improved significantly but the overshoots at the input change points have also increased.

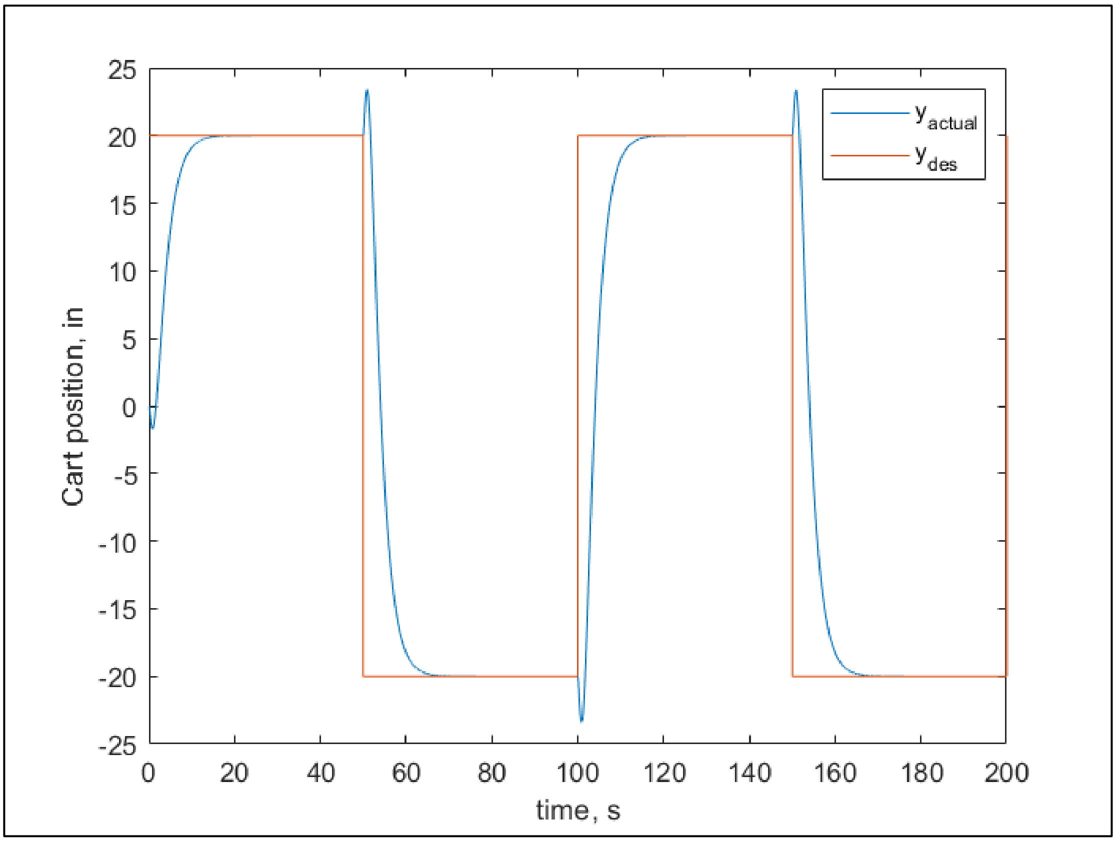

Cart position, in
time, s
y actual
y des
40
30
20
10
0
-10
-20
-30
-40
0
20
40
60
80
100
120
140
160
180
200

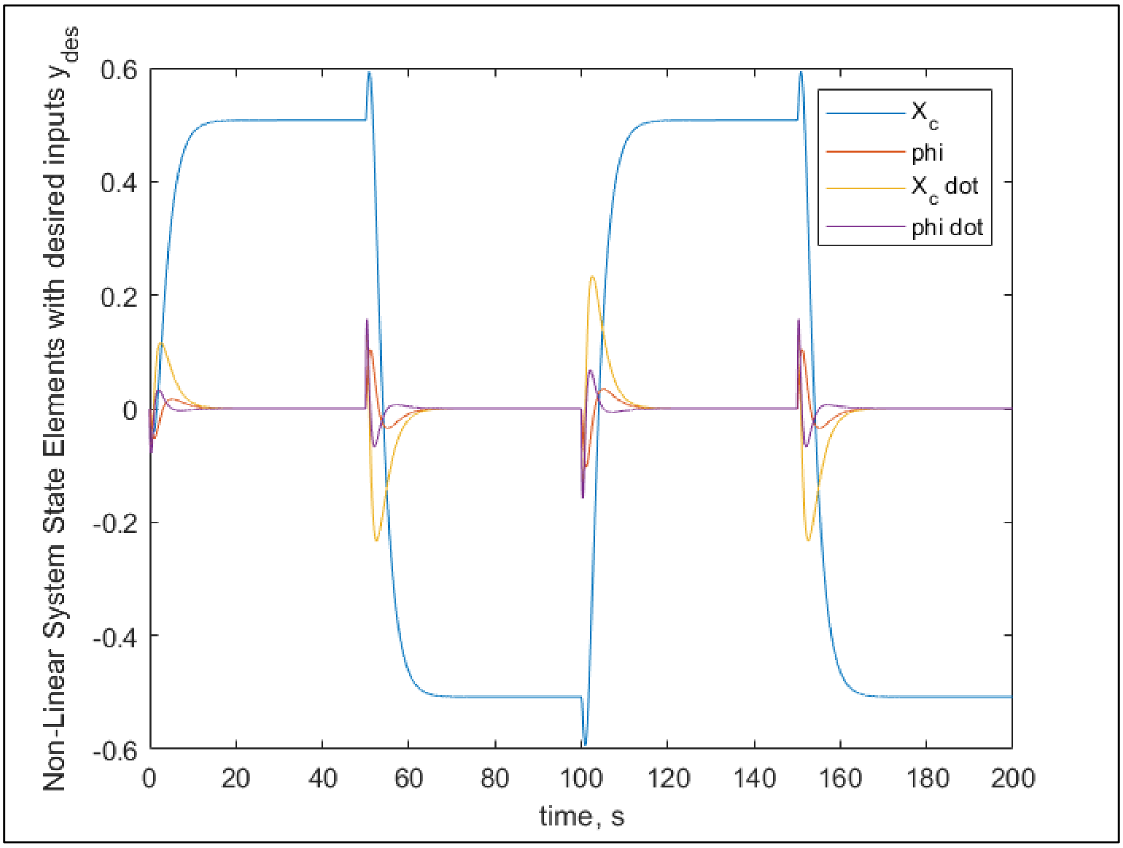

Non-Linear System State Elements with desired inputs y des
time, s
X c
phi
X c dot
phi dot
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
20
40
60
80
100
120
140
160
180
200

**Iteration 2:**

$Q_x$ = diag (700, 700, 15, 10)

$Q_u$ = 450




By increasing $Q_u$, I was able to reduce the size of the peaks while not affecting the time to converge significantly.

# CODE

**File Name: SpringMassDamper.m [Script]**

```
clear;
dt = 0.01;
t = 0:dt:5;
m = 1; mu = 0.5; k = 5;

A = [0 1;
    -k/m -mu/m];
B = [0 1/m]';
C = eye(2);
D = [0;0];

x_1 = [0 1]';

% response for 1e
x = zeros(2,length(t));
for idx = 1:length(t)
    x(:,idx) = expm(A*t(idx)) * x_1;
end
figure()
plot(t,x(1,:))
title("Unforced Open Loop System")
xlabel('Time, s')
ylabel('System position (q), m')

% 1.f
p = [-1 + 1i, -1 - 1i];
K = place(A,B,p);

% 1.g
tcl = 0:dt:10;
Acl = A - B*K;
xcl = zeros(2,length(tcl));
for j = 1:length(tcl)
    xcl(:,j) = expm(Acl*tcl(j)) * x_1;
end
figure()
plot(tcl,xcl(1,:))
title("System with Feedback Law")
xlabel('Time, s')
ylabel('System position (q), m')
```

**File Name: PendulumCart.m [Script]**

```
clear;

syms alpha beta gamma D mu xc phi xc_dot phi_dot F xc_ddot phi_ddot
equation1 = gamma * xc_ddot - beta * phi_ddot * cos(phi) + beta * phi_dot * phi_dot * sin(phi)
+ mu * xc_dot - F == 0;
equation2 = alpha * phi_ddot - beta * xc_ddot * cos(phi) - D * sin(phi) == 0;
Sol = solve([equation1,equation2],[xc_ddot,phi_ddot]);


dt = 0.01;
tspan = 0:dt:30;
%x_0 = [0,0.1,0,0]';
%x_0 = [0 0.5 0 0]';
%x_0 = [0 1.0886 0 0]';
x_0 = [0 1.1 0 0]';

A_linearised = [0 0 1 0; 0 0 0 1; 0 1 -3 0; 0 2 -3 0];
B_linearised = [0 0 1 1]';
```

```
Qu = 10;
Qx = [1 0 0 0; 0 5 0 0; 0 0 1 0; 0 0 0 5];
[K,S,P] = lqr(A_linearised,B_linearised,Qx,Qu);
eig(A_linearised)

t_nonlinear = 0:dt:5; %used for when system is not converging to 0
[t_sol_nl, x_sol_nl] = ode45(@system_def_nl_open,t_nonlinear,x_0);
figure()
plot(t_sol_nl, x_sol_nl(:,1))
hold on
plot(t_sol_nl, x_sol_nl(:,2))
hold on
plot(t_sol_nl, x_sol_nl(:,3))
hold on
plot(t_sol_nl, x_sol_nl(:,4))
legend('X_c','phi','X_c dot', 'phi dot')
xlabel('time, s')
ylabel('Non-Linear System State Elements')

[t_sol_l, x_sol_l] = ode45(@system_def_l,tspan,x_0);
figure()
plot(t_sol_l, x_sol_l(:,1))
hold on
plot(t_sol_l, x_sol_l(:,2))
hold on
plot(t_sol_l, x_sol_l(:,3))
hold on
plot(t_sol_l, x_sol_l(:,4))
legend('X_c','phi','X_c dot', 'phi dot')
xlabel('time, s')
ylabel('Linear System State Elements')
```

**File Name: TrackingController.m [Script]**

```
clear;
t_des = 0:0.01:200;
x_0 = [0,0,0,0];

[t_sol_nl, x_sol_nl] = ode45(@system_def_nl,t_des,x_0);

C = [39.37008 0 0 0];
y_actual = zeros(1,length(x_sol_nl));
x_sol_nl_t = x_sol_nl';
for i = 1:length(x_sol_nl_t)
    y_actual(i) = C * x_sol_nl_t(:,i);
end

figure()
plot(t_sol_nl,y_actual)
hold on
y_des = 20 * square(2*pi*0.01*t_sol_nl);
plot(t_sol_nl, y_des)
legend('y_{actual}', 'y_{des}')
xlabel('time, s')
ylabel('Cart position, in')

figure()
plot(t_sol_nl, x_sol_nl(:,1))
hold on
plot(t_sol_nl, x_sol_nl(:,2))
hold on
plot(t_sol_nl, x_sol_nl(:,3))
hold on
plot(t_sol_nl, x_sol_nl(:,4))
```

```
legend('X_c','phi','X_c dot', 'phi dot')
xlabel('time, s')
ylabel('Non-Linear System State Elements with desired inputs y_{des}')
```

**File Name: system_def_l.m [Function]**

```
function dx = system_def_l(t,x)
A_linearised = [0 0 1 0; 0 0 0 1; 0 1 -3 0; 0 2 -3 0];

B_linearised = [0 0 1 1]';

Qu = 10;
Qx = [1 0 0 0; 0 5 0 0; 0 0 1 0; 0 0 0 5];

[K,S,P] = lqr(A_linearised,B_linearised,Qx,Qu);
Acl = A_linearised - B_linearised* K;

dx = Acl*x;

end
```

**File Name: system_def_nl_open.m [Function]**

```
function dx = system_def_nl_open(t,x)
xc = x(1);
phi = x(2);
xc_dot = x(3);
phi_dot = x(4);

gamma = 2; alpha = 1; beta = 1; D = 1; mu = 3;

A_l = [0 0 1 0; 0 0 0 1; 0 1 -3 0; 0 2 -3 0];
B_l= [0 0 1 1]';
Qu = 10;
Qx = [1 0 0 0; 0 5 0 0; 0 0 1 0; 0 0 0 5];

[K,S,P] = lqr(A_l,B_l,Qx,Qu);

F = -K * x;

dx = [xc_dot;
    phi_dot;
    (- alpha*sin(phi)*beta*phi_dot^2 + F*alpha - alpha*mu*xc_dot +
cos(phi)*sin(phi)*D*beta)/(alpha*gamma - beta^2*cos(phi)^2);
    (- cos(phi)*sin(phi)*beta^2*phi_dot^2 + F*cos(phi)*beta + sin(phi)*D*gamma -
mu*xc_dot*cos(phi)*beta)/(alpha*gamma - beta^2*cos(phi)^2)
    ];
end
```

**File Name: system_def_nl.m [Function]**

```
function dx = system_def_nl(t,x)

xc = x(1);
phi = x(2);
xc_dot = x(3);
phi_dot = x(4);

gamma = 2; alpha = 1; beta = 1; D = 1; mu = 3;

A_l = [0 0 1 0; 0 0 0 1; 0 1 -3 0; 0 2 -3 0];
B_l= [0 0 1 1]';
C = [39.37008 0 0 0];
Qu = 450;
```

```
Qx = [700 0 0 0; 0 700 0 0; 0 0 15 0; 0 0 0 10];
%Qu = 10; Qx = [1 0 0 0; 0 5 0 0; 0 0 1 0; 0 0 0 5];
[K,S,P] = lqr(A_l,B_l,Qx,Qu);
Acl = A_l - B_l * K;
Acl_inv = inv(Acl);

y_des = 20 * square(2*pi*0.01*t);
K_f = -inv(C*Acl_inv*B_l);

F = K_f * y_des - K * x;


dx = [xc_dot;
    phi_dot;
    (- alpha*sin(phi)*beta*phi_dot^2 + F*alpha - alpha*mu*xc_dot +
cos(phi)*sin(phi)*D*beta)/(alpha*gamma - beta^2*cos(phi)^2);
    (- cos(phi)*sin(phi)*beta^2*phi_dot^2 + F*cos(phi)*beta + sin(phi)*D*gamma -
mu*xc_dot*cos(phi)*beta)/(alpha*gamma - beta^2*cos(phi)^2)
    ];

end
```