Part A Question 1

The effect of setting the prediction horizon N = 10 and N = 5 affects the states and control action in an Unconstrained MPC as shown in Figure 1 below:

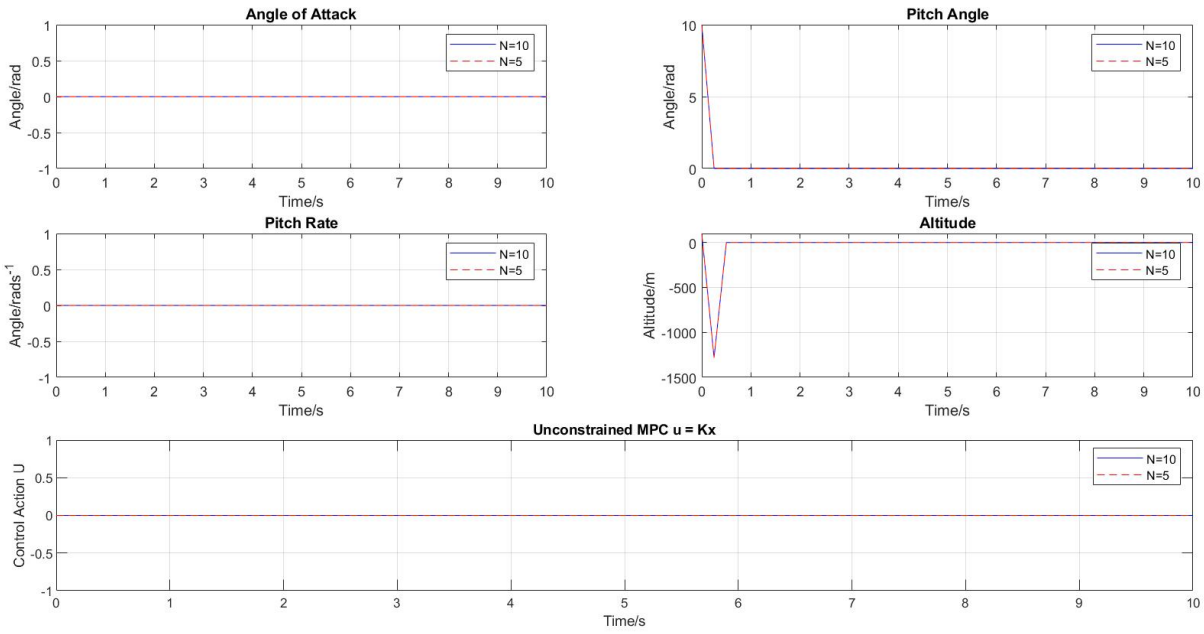

Figure 1: Part A Question 1

As seen from the figure, the pitch angle and the altitude, which are states x2 and x4 respectively, are converging to zero regardless of control action. This result comes from the fact that the F matrix contains two zeros that in turn makes the gain matrix K consists of two zeros x2 and x4 and thus, the controller is capable of stabilising the system.

Part A Question 2

The effect of changing the initial condition from x(0) = [0 10 0 100] to x(0) = [10 0 100 0] affects the states and control action in an Unconstrained MPC as shown in Figure 2 below:

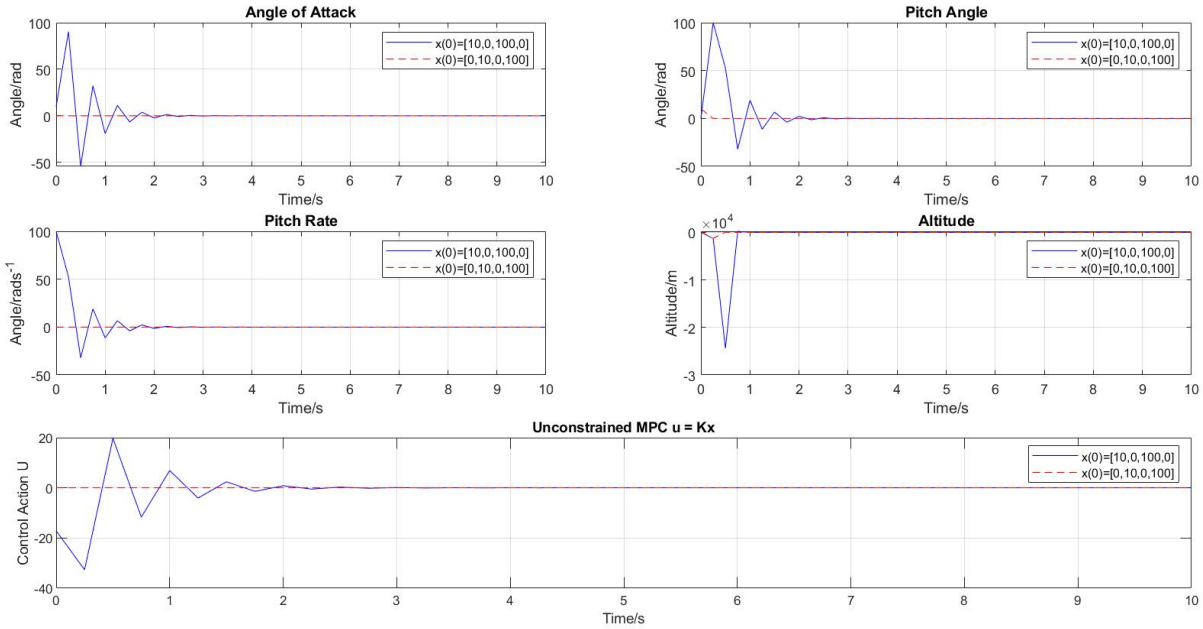

Figure 2: Part A Question 2

As seen from the figure again, similar to Part A Question 1, the pitch angle and the altitude are also converging to zero regardless of control action. Even though the initial conditions of the angle of attack and pitch rate are changed, making the system more unstable, the controller is still able to stabilise the system, although it takes a while longer and with a few more fluctuations to stabilise. This result comes from the fact that the F matrix also contains two zeros. This means that the gain matrix K still consists of 2 zeros that makes x2 and x4 zero and thus, the controller is capable of adjusting the pitch angle and altitude to zero.

Part B Question 1

The effect of setting the prediction horizon N = 10 and N = 5 affects the states and control action in an Constrained MPC as shown in Figure 3 below:

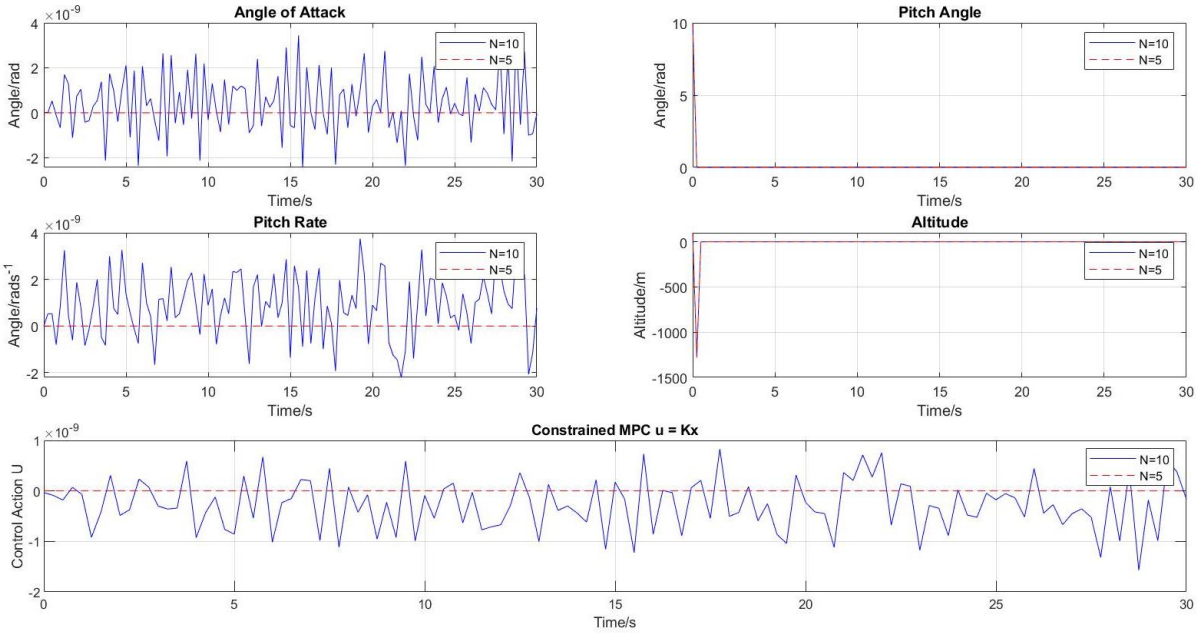

Figure 3: Part B Question 1

As seen from the figure, the pitch angle and the altitude are also converging to zero while keeping the constraint of $-0.262 \text{ rad} \leq u \leq 0.262 \text{ rad}$. Although the angle of attack and the pitch rate are slightly fluctuating, they are insignificant as the fluctuations are a maximum of 4×10^{-9} in magnitude. This result comes from the fact that the F matrix contains two zeros that in turn makes the gain matrix K consists of two zeros x2 and x4 and thus, the controller is capable of stabilising the system.

Part B Question 2

The effect of changing the initial condition from x(0) = [0 10 0 100] to x(0) = [10 0 100 0] affects the states and control action in an Constrained MPC as shown in Figure 4 below:

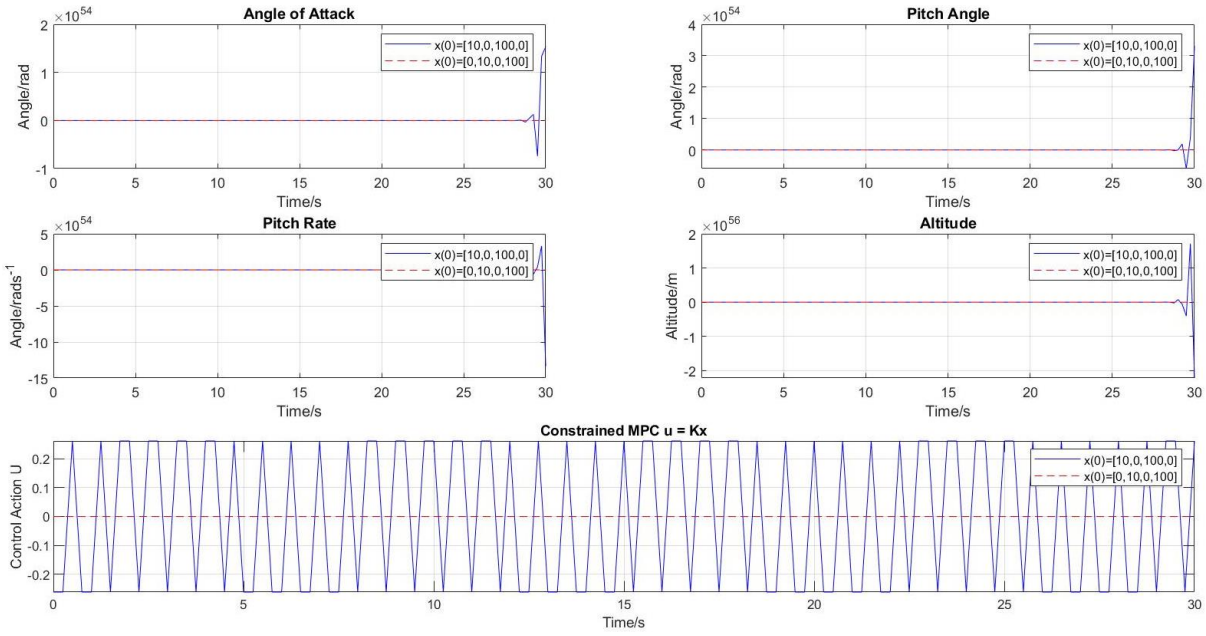

Figure 4: Part B Question 2

As seen from the Figure 4, the pitch angle and the altitude are not able to converge to zero. When this graph is compared to Question A Part 2, the Control Action reaches about u=25. However, because of the constraint of $-0.262 \text{ rad} \leq u \leq 0.262 \text{ rad}$, the system is unable to stabilise, resulting in a fluctuation of the pitch angle and the altitude. This result comes from the fact that the inputs given to the system make the system uncontrollable and thus makes the system unstable.

Modifying Q and R

By modifying the values of Q and R, some changes could be made to the graphs above as it is possible to set the values of Q and R to get the better response. The parameters Q and R can be used as design parameters to penalise the state variables and the control signals. By choosing a large value for R, the system will be stabilised with less weighted energy. However, this is an expensive control strategy. On the other hand, choosing a small value for R means less penalising of the control signal, making it a cheap control strategy. Similarly, choosing a large value for Q stabilises the system with the least possible changes in the states and small Q stabilises the system with the most possible changes.

Although it is possible to have better values for Part A Question 1and 2 and Part B Question 1, it is impossible to stabilise the unstable system in Part B Question 2. This is due to the fact the since the system is unstable and uncontrollable, setting penalty parameters will not contribute towards the system as it is saturated by the required control action. Thus, no matter how expensive the control strategy is, if there is a constraint put in place that limits the effectiveness of the penalty matrices, then it is impossible for the penalty matrices to stabilise the system.

Appendix

Below shows the code used in MATLAB for Part A:

```
clear all;
clc; close all;

%% Question 1
x = [0;10;0;100]; % Initial Condition
y = [0;0;0;5000]; % Output is x1, x2->0
dt = 0.25; % sampling Time
u = 0; % initial control signal
T_end = 10; % ending simulation

% States
A = [-1.28 0 0.98 0;0 0 1 0;-5.43 0 -1.84 0;-128.2 -128.2 0 0];
B = [-0.3;0;-17;0];
C = eye(4);

%% Unconstrained MPC
N1 = 5;
N2 = 10;
[u0,x0,y0,t0] = UnconstrainedMPC(A,B,C,x,y,N1,dt,T_end);
[u1,x1,y1,t1] = UnconstrainedMPC(A,B,C,x,y,N2,dt,T_end);

%% Plotting
figure;
subplot(3,2,1)
plot(t1,x1(1,1:end-1),'b-',t0,x0(1,1:end-1),'r--')
title('Angle of Attack');legend('N=10','N=5');
xlabel('Time/s');ylabel('Angle/rad')
grid on;
subplot(3,2,2)
plot(t1,x1(2,1:end-1),'b-',t0,x0(2,1:end-1),'r--')
title('Pitch Angle');legend('N=10','N=5');
xlabel('Time/s');ylabel('Angle/rad')
grid on;
subplot(3,2,3)
plot(t1,x1(3,1:end-1),'b-',t0,x0(3,1:end-1),'r--')
title('Pitch Rate');legend('N=10','N=5');
xlabel('Time/s');ylabel('Angle/rads^-^1')
grid on;
subplot(3,2,4)
plot(t1,x1(4,1:end-1),'b-',t0,x0(4,1:end-1),'r--')
title('Altitude');legend('N=10','N=5');
xlabel('Time/s');ylabel('Altitude/m')
grid on;
subplot(3,2,[5,6])
plot(t1,u1,'b-',t0,u0,'r--')
xlabel('Time/s'); ylabel('Control Action U')
title('Unconstrained MPC u = Kx');legend('N=10','N=5');
grid on;

%% Question 2
x0 = [0;10;0;100]; % Initial Condition
x1 = [10;0;100;0]; % Initial Condition
```

```
y = [0;0;0;5000]; % Output is x1, x2->0
dt = 0.25; % sampling Time
u = 0; % initial control signal
T_end = 10; % ending simulation

% States
A = [-1.28 0 0.98 0;0 0 1 0;-5.43 0 -1.84 0;-128.2 -128.2 0 0];
B = [-0.3;0;-17;0];
C = eye(4);

%% Unconstrained MPC
N = 10;
[u0,x0,y0,t0] = UnconstrainedMPC(A,B,C,x0,y,N,dt,T_end);
[u1,x1,y1,t1] = UnconstrainedMPC(A,B,C,x1,y,N,dt,T_end);

%% Plotting
figure;
subplot(3,2,1)
plot(t1,x1(1,1:end-1),'b-',t0,x0(1,1:end-1),'r--')
title('Angle of
Attack');legend('x(0)=[10,0,100,0]','x(0)=[0,10,0,100]');
xlabel('Time/s');ylabel('Angle/rad')
grid on;
subplot(3,2,2)
plot(t1,x1(2,1:end-1),'b-',t0,x0(2,1:end-1),'r--')
title('Pitch
Angle');legend('x(0)=[10,0,100,0]','x(0)=[0,10,0,100]');
xlabel('Time/s');ylabel('Angle/rad')
grid on;
subplot(3,2,3)
plot(t1,x1(3,1:end-1),'b-',t0,x0(3,1:end-1),'r--')
title('Pitch Rate');legend('x(0)=[10,0,100,0]','x(0)=[0,10,0,100]');
xlabel('Time/s');ylabel('Angle/rads^-^1')
grid on;
subplot(3,2,4)
plot(t1,x1(4,1:end-1),'b-',t0,x0(4,1:end-1),'r--')
title('Altitude');legend('x(0)=[10,0,100,0]','x(0)=[0,10,0,100]');
xlabel('Time/s');ylabel('Altitude/m')
grid on;
subplot(3,2,[5,6])
plot(t1,u1,'b-',t0,u0,'r--')
xlabel('Time/s'); ylabel('Control Action U')
title('Unconstrained MPC u =
Kx');legend('x(0)=[10,0,100,0]','x(0)=[0,10,0,100]');
grid on;

%% Function
function [u,x,y,t] = UnconstrainedMPC(A,B,C,x,y,N,dt,T_end)
    t = 0:dt:T_end; % total time duration
    Nsim = length(t) - 1; % Total Samples

    nx=size(A,2); % number of states
    nu=size(B,2); % number of inputs
    R = 10; %Input penalization
    Q = diag([0,1,0,1]); %State Penalization
    Q_bar = kron(eye(N),Q); %Running over the Prediction Horizon
```

```
    R_bar = kron(eye(N),R); %Running over the Preidction Horizon

    % Build Sx Build Su for Equation 5 (X = x_0*Sx + u*Su)
    Su=zeros(N*nx,N*nu);
    for i=1:N
       Sx((i-1)*nx+1:i*nx,:)=A^i; %Sx
       for j=1:i
          Su((i-1)*nx+1:(i)*nx,(j-1)*nu+1:j*nu)=A^(i-j)*B; %Su
       end
    end

    F = Su'*Q_bar*Sx; %Matrix F for the Equation 20-21
    G = R_bar + Su'*Q_bar*Su; %Matrix G
    z = zeros(1,N-1); %Zeros for gain multiplication
    K = -[1 z]*inv(G)*F; %MPC Gain

    %Simulation
    for k = 1:Nsim+1
        u(:,k) = K*x(:,k); %control action u = kx
        x(:,k+1) = A*x(:,k) + B*u(:,k); %state equation
        y(:,k) = C*x(:,k); %output equation
    end
end
```

Below shows the code used in MATLAB for Part B:

```
clear all;
clc; close all;

%% Question 1
x = [0;10;0;100]; % Initial Condition
y = [0;0;0;5000]; % Output is x1, x2->0
dt = 0.25; % sampling Time
u = 0; % initial control signal
T_end = 30; % ending simulation

% States
A = [-1.28 0 0.98 0;0 0 1 0;-5.43 0 -1.84 0;-128.2 -128.2 0 0];
B = [-0.3;0;-17;0];
C = eye(4);

%% Constrained MPC
N1 = 5;
N2 = 10;
[u0,x0,y0,t0] = ConstrainedMPC(A,B,C,x,y,N1,dt,T_end);
[u1,x1,y1,t1] = ConstrainedMPC(A,B,C,x,y,N2,dt,T_end);

%% Plotting
figure;
subplot(3,2,1)
plot(t1,x1(1,1:end-1),'b-',t0,x0(1,1:end-1),'r--')
title('Angle of Attack');legend('N=10','N=5');
xlabel('Time/s');ylabel('Angle/rad')
grid on;
subplot(3,2,2)
```

```
plot(t1,x1(2,1:end-1),'b-',t0,x0(2,1:end-1),'r--')
title('Pitch Angle');legend('N=10','N=5');
xlabel('Time/s');ylabel('Angle/rad')
grid on;
subplot(3,2,3)
plot(t1,x1(3,1:end-1),'b-',t0,x0(3,1:end-1),'r--')
title('Pitch Rate');legend('N=10','N=5');
xlabel('Time/s');ylabel('Angle/rads^-^1')
grid on;
subplot(3,2,4)
plot(t1,x1(4,1:end-1),'b-',t0,x0(4,1:end-1),'r--')
title('Altitude');legend('N=10','N=5');
xlabel('Time/s');ylabel('Altitude/m')
grid on;
subplot(3,2,[5,6])
plot(t1,u1,'b-',t0,u0,'r--')
xlabel('Time/s'); ylabel('Control Action U')
title('Constrained MPC u = Kx');legend('N=10','N=5');
grid on;

%% Question 2
x0 = [0;10;0;100]; % Initial Condition
x1 = [10;0;100;0]; % Initial Condition
y = [0;0;0;5000]; % Output is x1, x2->0
dt = 0.25; % sampling Time
u = 0; % initial control signal
T_end = 30; % ending simulation

% States
A = [-1.28 0 0.98 0;0 0 1 0;-5.43 0 -1.84 0;-128.2 -128.2 0 0];
B = [-0.3;0;-17;0];
C = eye(4);

%% Constrained MPC
N = 10;
[u0,x0,y0,t0] = ConstrainedMPC(A,B,C,x0,y,N,dt,T_end);
[u1,x1,y1,t1] = ConstrainedMPC(A,B,C,x1,y,N,dt,T_end);

%% Plotting
figure;
subplot(3,2,1)
plot(t1,x1(1,1:end-1),'b-',t0,x0(1,1:end-1),'r--')
title('Angle of
Attack');legend('x(0)=[10,0,100,0]','x(0)=[0,10,0,100]');
xlabel('Time/s');ylabel('Angle/rad')
grid on;
subplot(3,2,2)
plot(t1,x1(2,1:end-1),'b-',t0,x0(2,1:end-1),'r--')
title('Pitch
Angle');legend('x(0)=[10,0,100,0]','x(0)=[0,10,0,100]');
xlabel('Time/s');ylabel('Angle/rad')
grid on;
subplot(3,2,3)
plot(t1,x1(3,1:end-1),'b-',t0,x0(3,1:end-1),'r--')
title('Pitch Rate');legend('x(0)=[10,0,100,0]','x(0)=[0,10,0,100]');
xlabel('Time/s');ylabel('Angle/rads^-^1')
```

```
grid on;
subplot(3,2,4)
plot(t1,x1(4,1:end-1),'b-',t0,x0(4,1:end-1),'r--')
title('Altitude');legend('x(0)=[10,0,100,0]','x(0)=[0,10,0,100]');
xlabel('Time/s');ylabel('Altitude/m')
grid on;
subplot(3,2,[5,6])
plot(t1,u1,'b-',t0,u0,'r--')
xlabel('Time/s'); ylabel('Control Action U')
title('Constrained MPC u =
Kx');legend('x(0)=[10,0,100,0]','x(0)=[0,10,0,100]');
grid on;

%% Function
function [u,x,y,t] = ConstrainedMPC(A,B,C,x,y,N,dt,T_end)
t = 0:dt:T_end; % total time duration
    Nsim = length(t) - 1; % Total Samples

    nx=size(A,2); % number of states
    nu=size(B,2); % number of inputs
    R = 10; % Input penalization
    Q = diag([0,1,0,1]); %State Penalization
    Q_bar = kron(eye(N),Q); %Running over the Prediction Horizon
    R_bar = kron(eye(N),R); %Running over the Preidction Horizon

    % Build Sx Build Su for Equation 5 (X = x_0*Sx + u*Su)
    Su=zeros(N*nx,N*nu);
    for i=1:N
       Sx((i-1)*nx+1:i*nx,:)=A^i; %Sx
       for j=1:i
          Su((i-1)*nx+1:(i)*nx,(j-1)*nu+1:j*nu)=A^(i-j)*B; %Su
       end
    end

    F = Su'*Q_bar*Sx; %Matrix F for the Equation 20-21
    G = R_bar + Su'*Q_bar*Su; %Matrix G
    z = zeros(1,N-1); %Zeros for gain multiplication
    K = -[1 z]*inv(G)*F; %MPC Gain

    % Constrained MPC Part
    H = 2*G; % Cost vector H
    E1 = eye(N,N); E2 = eye(N,N);
    E = [E1;-E2]; % For Constraints Ez <= b
    u_max = 0.262;
    u_min = -0.262;

    b_unit = [u_max; -u_min]; % Constraints on Input
    b = [];

    % Create vector b based of N
    for i = 1:N
        b = [b;b_unit];
    end

    % Simulation
    for k = 1 : Nsim + 1
```

```
        xp = x(:,k); % Measure the state x0 at time instant k
        f = 2*(F *xp); %update cost vector
        options = optimoptions('quadprog','MaxIterations',300);
        Aeq = [];
        beq = [];
        lb = [];
        ub = [];
        x0 = [];
        uopt = quadprog(H,f,E,b,Aeq,beq,lb,ub,x0,options); %Compute
optimal control U^*
        u(:,k) = uopt(1); %Apply the first element u^*[0] of U to
the platform
        x(:, k + 1) = A*x(:,k) + B*u(:, k);
        y(:,k) = C*x(:,k);
    end
end
```