THE UNIVERSITY OF TEXAS AT ARLINGTON, TEXAS
DEPARTMENT OF ELECTRICAL ENGINEERING

EE 5321 - 001

OPTIMAL CONTROL

HW # 6
ASSIGNMENT

by

SOUTRIK MAITI

1001569883

Presented to

Prof. Michael Niestroy

April 26,2018

Problem 1:

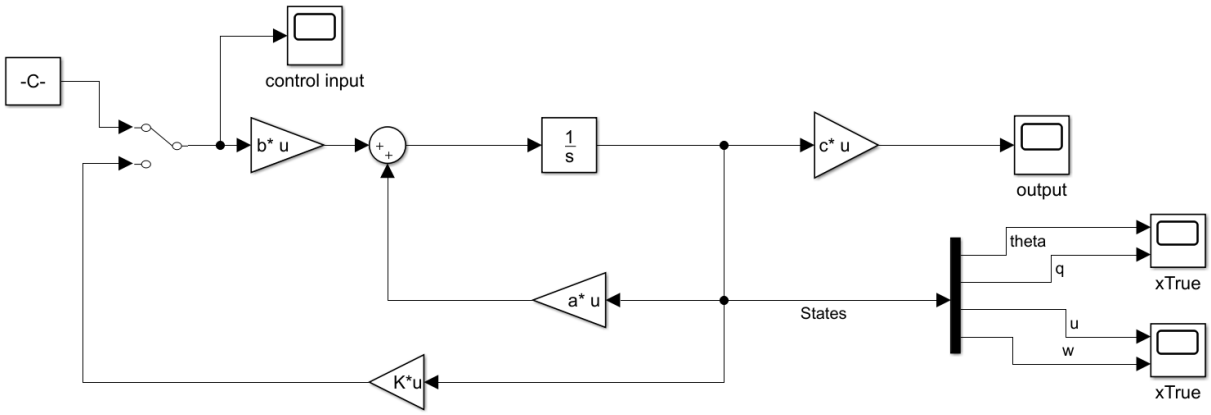

Fig1.1 Simulink Diagram for problem 1.

MATLAB Code:

```
%% The system matrices
a = [ -0.0000 1.0000 -0.0000 -0.0000;
    -0.0000 -0.5004 -0.0022 0.0064;
    -30.8405 -66.2409 0.0089 0.0199;
    -8.8533 224.6563 -0.0360 -0.8097 ];

b = [ -0.0000 0.0000;
    -0.0226 0.0212;
    0.0010 0.0035;
    -0.1601 0.2148 ];

c = [ 1.0000 -0.0050 0.0000 0.0000;
    0.0000 1.0025 0.0000 -0.0000;
    0.1604 0.0084 0.9612 0.2769;
    0.0000 -0.0049 -0.0011 0.0040 ];

d = zeros(4,2);
%% The State space system
system = ss(a,b,c,d);
%% Poles of the system
poles = eig(system)
%% Pole zero plot of the system
pzplot(system);
%% Damping of the system
damp(system);
```

a) Pole location, damping, and frequencies of the open loop system

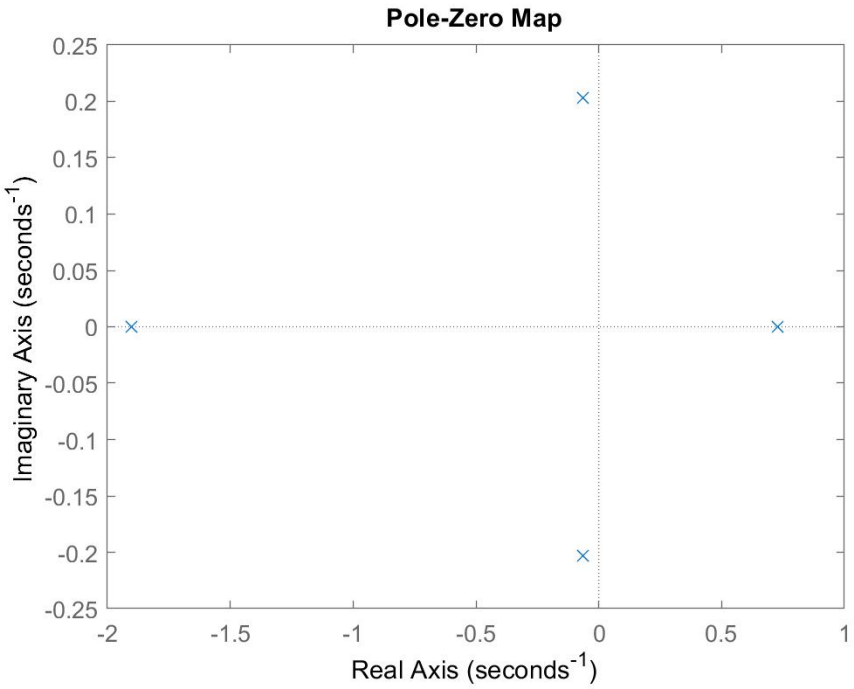

Fig1.2 Pole zero plot of the system

```
poles =

   0.7288 + 0.0000i
  -0.0643 + 0.2032i
  -0.0643 - 0.2032i
  -1.9013 + 0.0000i
```

Pole	Damping	Frequency (rad/seconds)	Time Constant (seconds)
-6.43e-02 + 2.03e-01i	3.02e-01	2.13e-01	1.55e+01
-6.43e-02 - 2.03e-01i	3.02e-01	2.13e-01	1.55e+01
7.29e-01	-1.00e+00	7.29e-01	-1.37e+00
-1.90e+00	1.00e+00	1.90e+00	5.26e-01

b) Since one of the poles of the system lies in the right half plane, the open loop system is unstable.

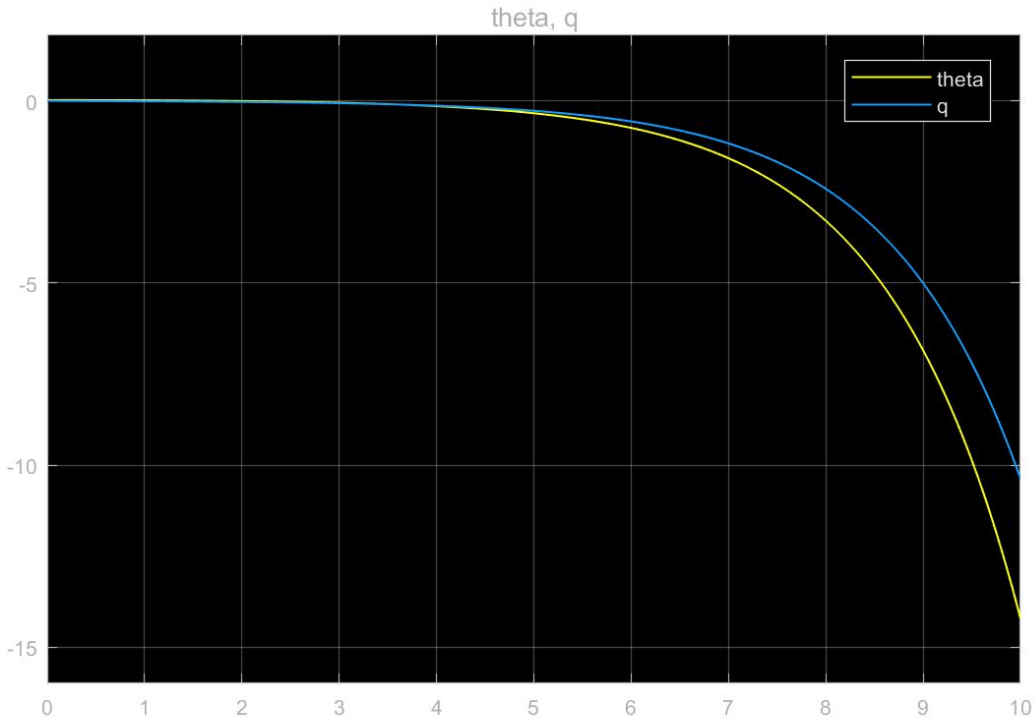

Fig1.3 Unstable states theta and q

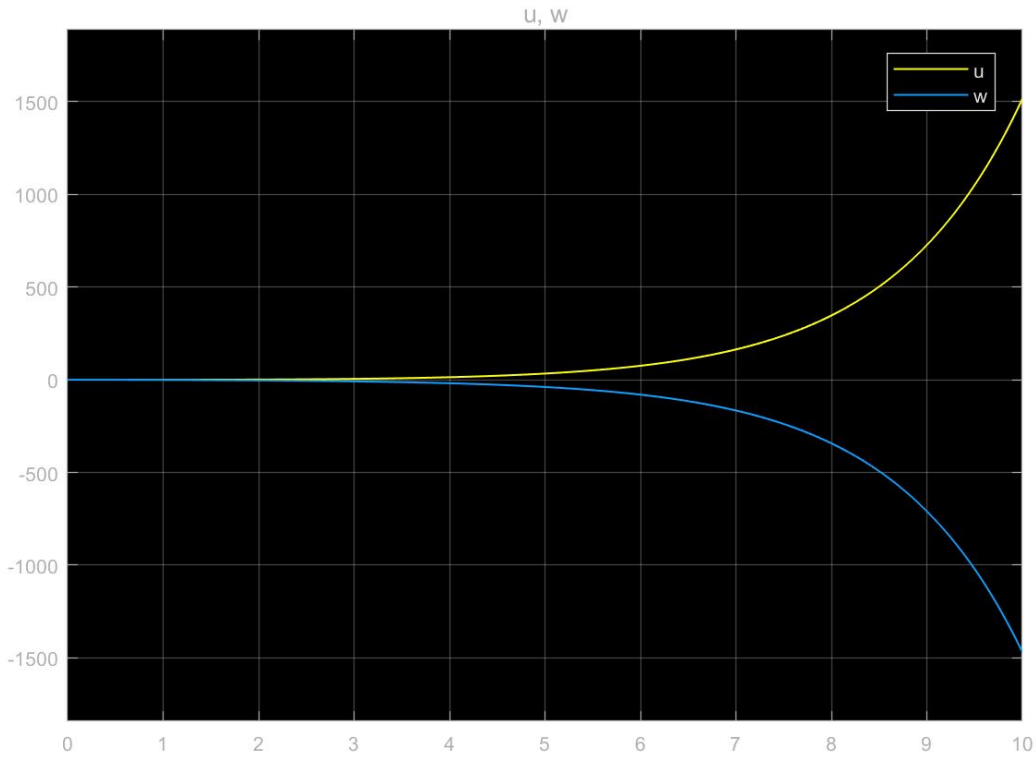

Fig1.4 Unstable states u and w

c) To show that the system is controllable and observable, we implement the following in MATLAB

```
if rank(ctrb(system))== 4
    disp('System is Controllable. Since, Rank of Controllability matrix
= no. of states');
end
if rank(obsv(system))== 4
    disp('System is Observable. Since, Rank of Observability matrix =
no. of states');
end
```

Results:

```
System is Controllable. Since, Rank of Controllability matrix = no. of states
System is Observable. Since, Rank of Observability matrix = no. of states
```

d) MATLAB Code:

```
%% Implementing state feedback
Q = 10 * eye(4);
R = eye(2);
[K,S,e] = lqr(a,b,Q,R)
controlledSystem = ss(a-b*K,b,c,d)
newpoles = eig(controlledSystem)
figure
pzplot(controlledSystem);
```

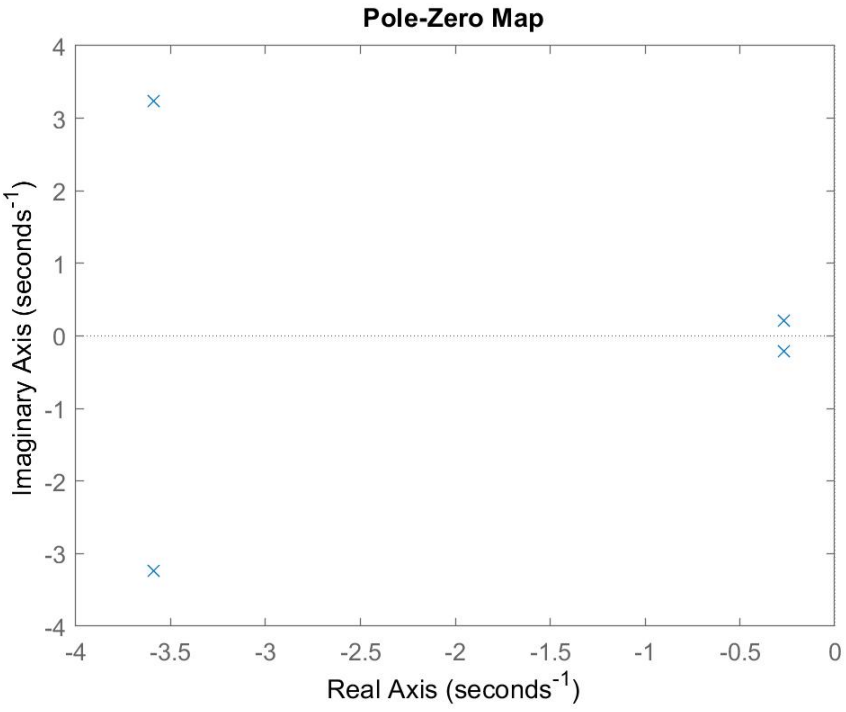

Fig1.5 Pole zero plot of the stabilized system

We can see from the results that the poles lie in the left half plane so the system is stable.

e) After implementing negative feedback by flipping the manual switch, the Simulink diagram looks like this:

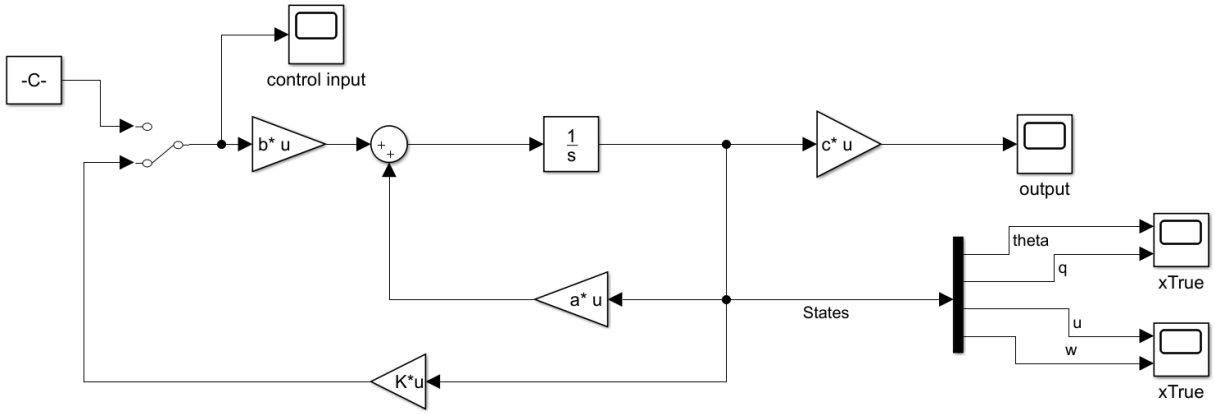

Fig1.6 Implementing state feedback in the same Simulink diagram

The results are as follows:

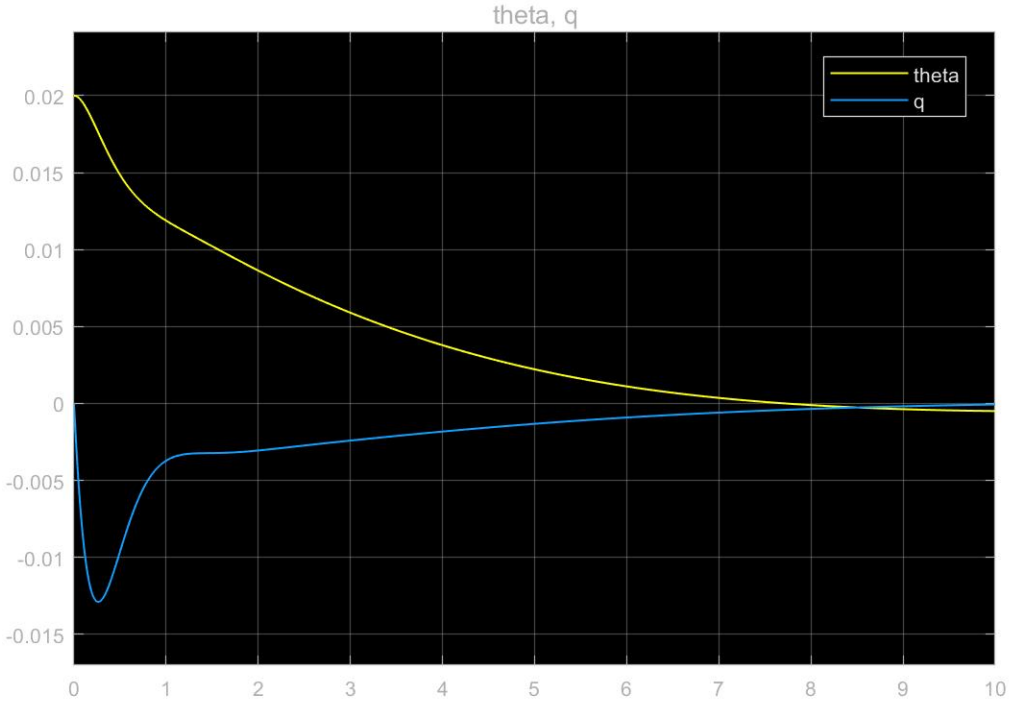

Fig1.7 Stable theta and q

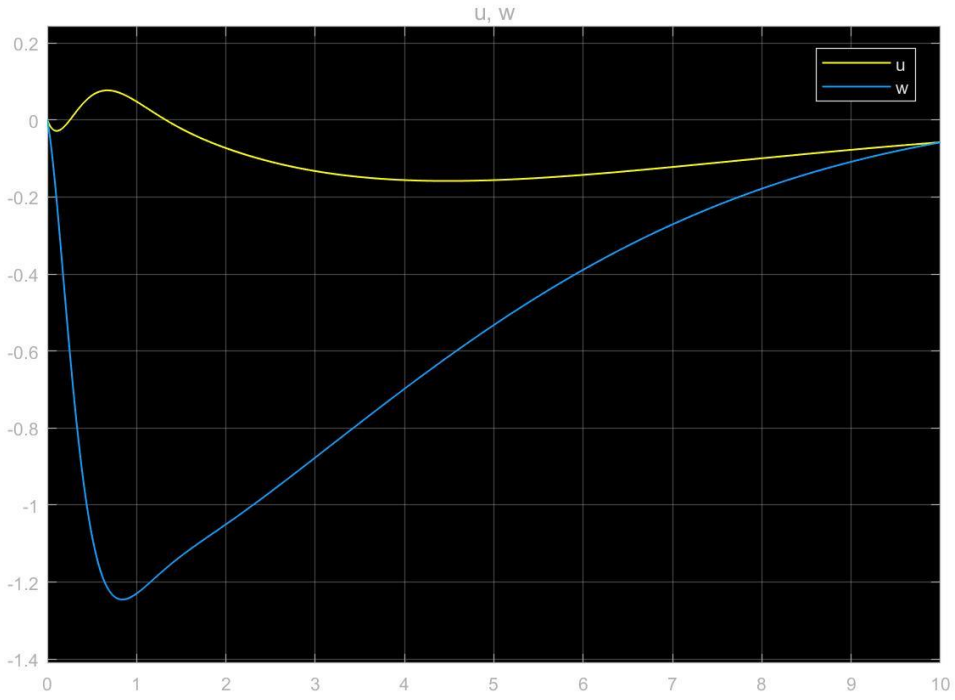

Fig1.8 Stable u and w

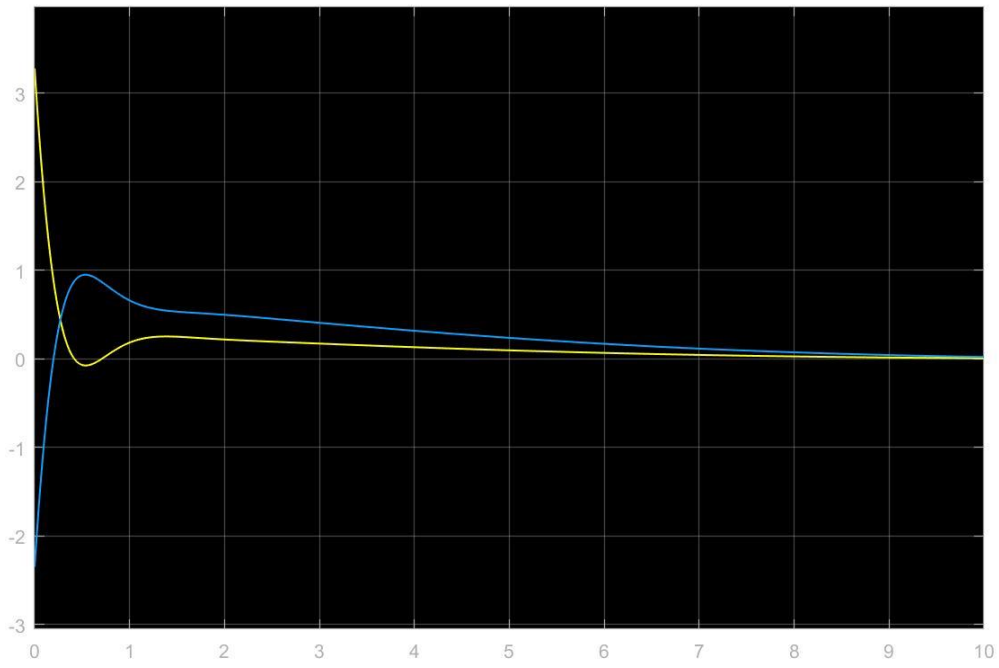

Fig1.9 Control time history

Problem 2:

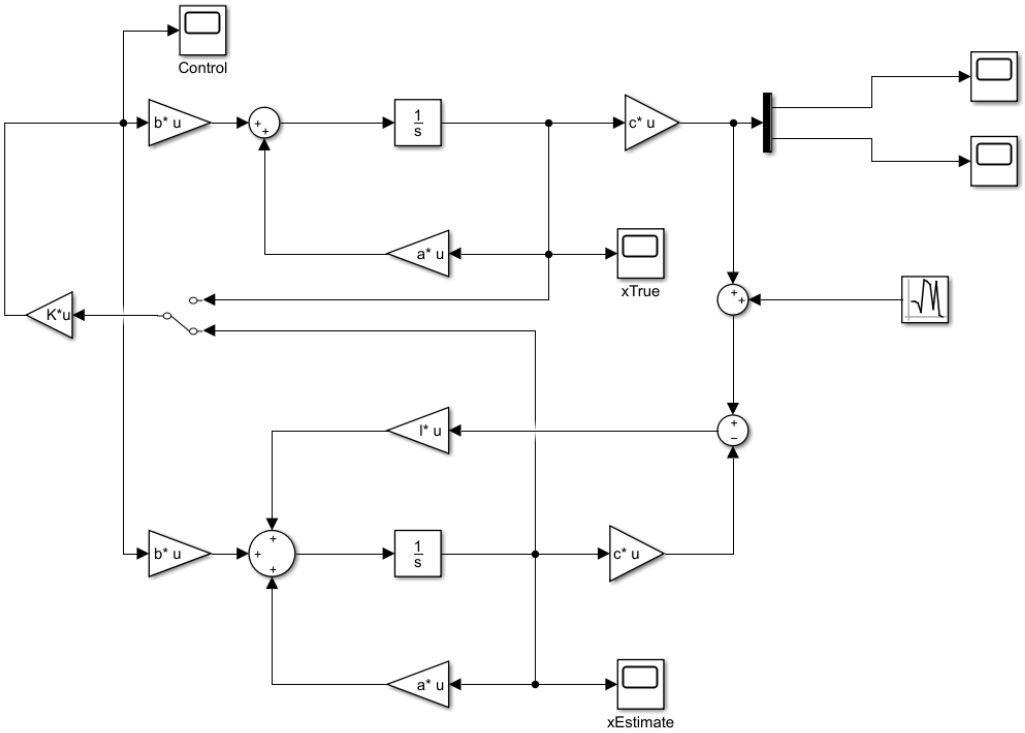

Fig2.1 Simulink diagram for observer

MATLAB Code:

```
%% The system matrices
a = [ -0.0000 1.0000 -0.0000 -0.0000;
    -0.0000 -0.5004 -0.0022 0.0064;
    -30.8405 -66.2409 0.0089 0.0199;
    -8.8533 224.6563 -0.0360 -0.8097 ];

b = [ -0.0000 0.0000;
    -0.0226 0.0212;
    0.0010 0.0035;
    -0.1601 0.2148 ];

c = [ 1.0000 -0.0050 0.0000 0.0000;
    0.0000 1.0025 0.0000 -0.0000;
    0.1604 0.0084 0.9612 0.2769;
    0.0000 -0.0049 -0.0011 0.0040 ];

d = zeros(4,2);
%% The State space system
system = ss(a,b,c,d);
%% Implementing state feedback
Q = 10 * eye(4);
R = eye(2);
[K,S,e] = lqr(a,b,Q,R);
controlledSystem = ss(a-b*K,b,c,d);
newpoles = eig(controlledSystem);
```

```
figure
pzplot(controlledSystem);

%% for observer
q = 0.1 * eye(4);
r = 5 * eye(4);
[l,s,e1] = lqr(a',c',q,r)
```

Results:

```
l =

    0.0164    0.0003   -0.0564   -0.3293
    0.0005    0.0320   -1.3533    3.9452
   -0.1438   -0.2075   13.1779  -28.9002
   -0.0012    0.0171   -0.8529    2.4947


s =

   1.0e+03 *

    0.0001    0.0000   -0.0003   -0.0015
    0.0000    0.0002   -0.0067    0.0197
   -0.0003   -0.0067    0.3504   -0.9781
   -0.0015    0.0197   -0.9781    2.8736


e1 =

  -1.5482 + 1.8485i
  -1.5482 - 1.8485i
  -0.4625 + 0.0000i
  -2.4410 + 0.0000i
```

a) Comparing estimated states with the true states when using true state as the feedback.

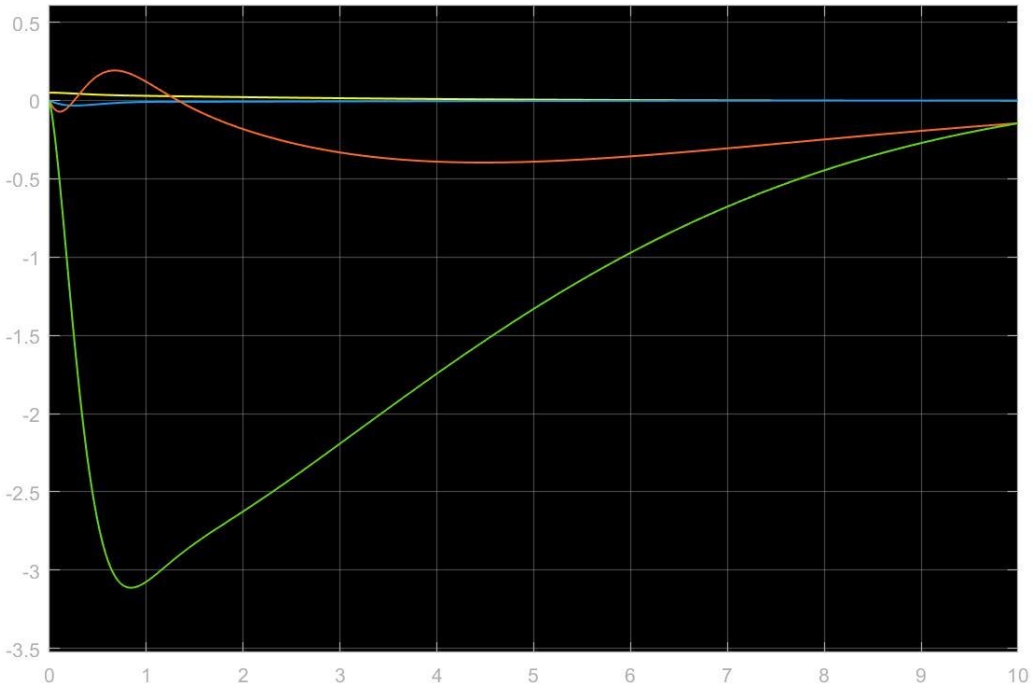

Fig2.2 True states

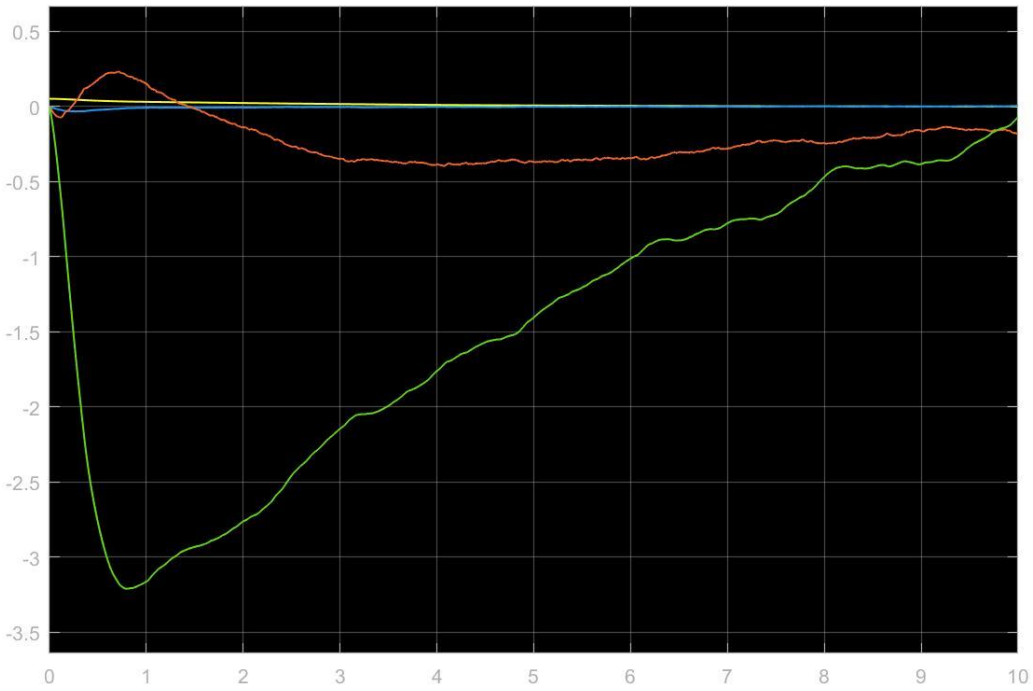

Fig2.3 Estimated states

State time history and control time history when the estimated states are fed back to the main system.

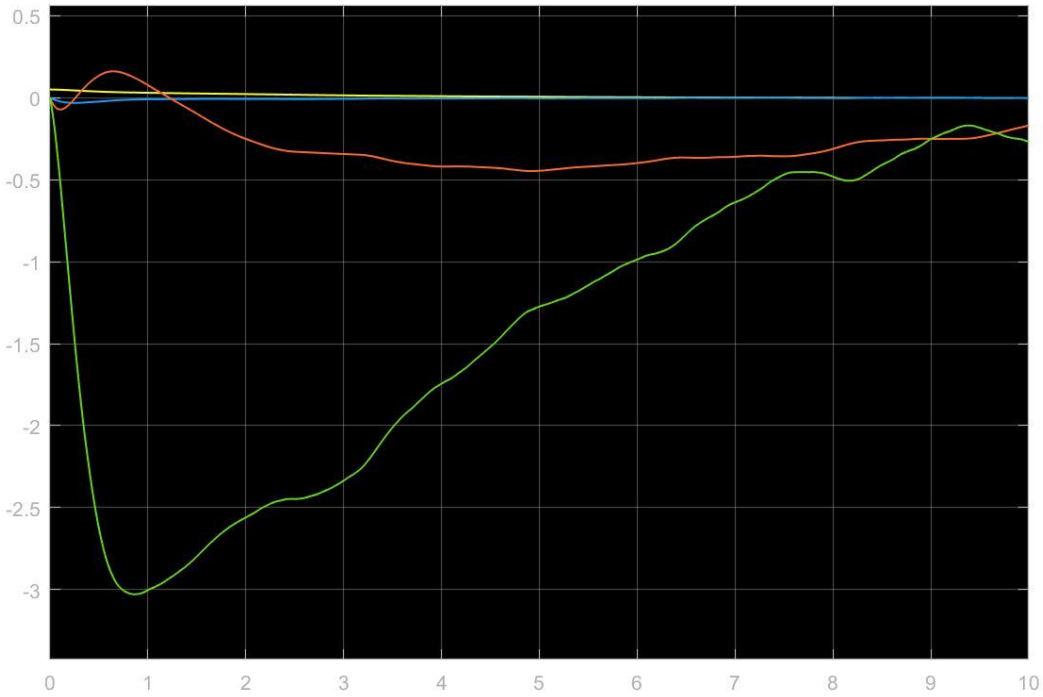

Fig2.4 State time history with estimated state feedback

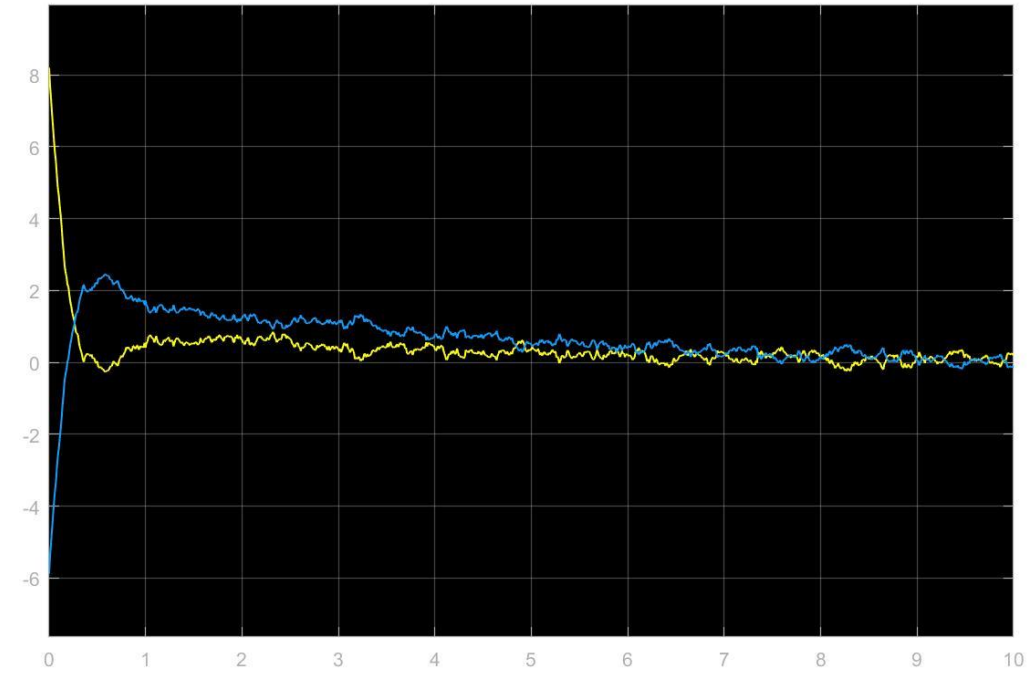

Fig2.5 Control time history with estimated state feedback

b) Adjusting the values of Q and R to make the eigen values at least 5 times those of the closed loop eigen values.
MATLAB Code:

```
%% shifting the eigen values
q1 = 15*eye(4);
r1 = 0.1*eye(4);
[l1,s1,e2] = lqr(a',c',q1,r1)
```

Results:

Eigen values of a-b*K

```
e =

  -3.5897 + 3.2429i
  -3.5897 - 3.2429i
  -0.2680 + 0.2052i
  -0.2680 - 0.2052i
```

Eigen values after changing q and r

```
e2 =

 -16.8696 +11.6389i
 -16.8696 -11.6389i
  -1.8051 + 0.0000i
 -12.5416 + 0.0000i
```

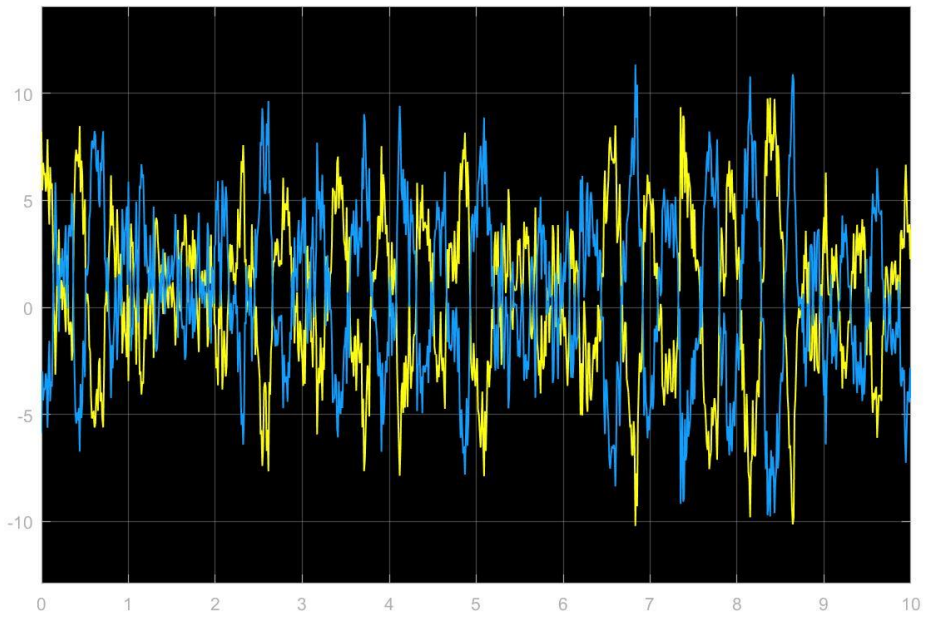

Fig2.6 Control time history when eigen values shifted

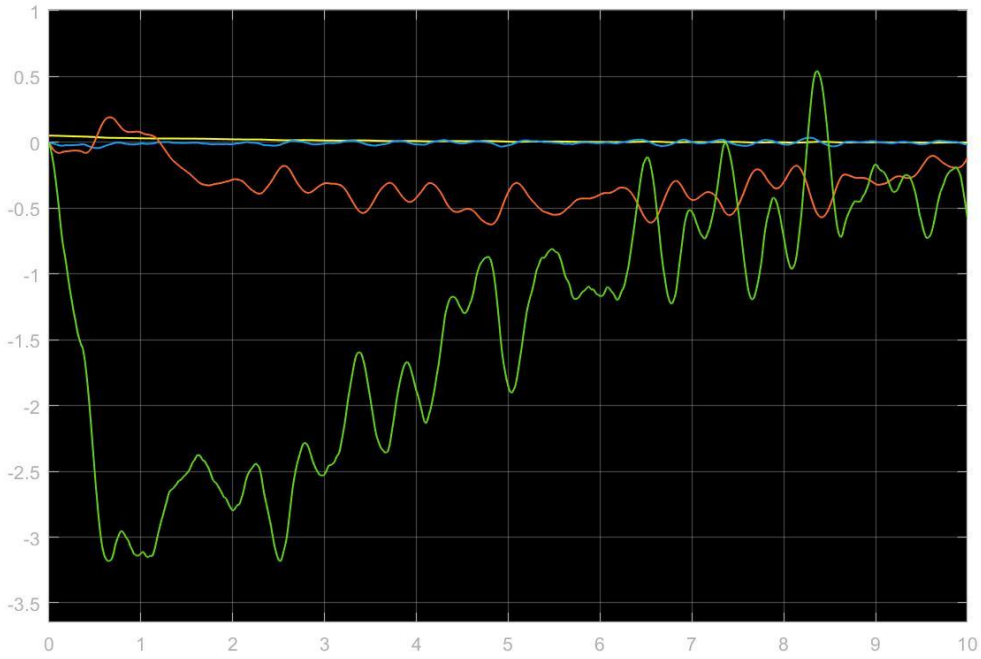

Fig2.7 State time history when eigen values shifted

Problem 3:

a) For the first part, the MATLAB code is the same except a few changes as follows:

```
%% for Kalman filter
P0 = 1000 * eye(4);
```

The Kalman filter consists of a predictor and a corrector block which is implemented using the following scripts in the MATLAB function block:

Predictor:

```
%% Predictor algorithm
function [xpred, ppred]= predictor(u,x,P)
a = [ -0.0000 1.0000 -0.0000 -0.0000;
    -0.0000 -0.5004 -0.0022 0.0064;
    -30.8405 -66.2409 0.0089 0.0199;
    -8.8533 224.6563 -0.0360 -0.8097 ];

b = [ -0.0000 0.0000;
    -0.0226 0.0212;
    0.0010 0.0035;
    -0.1601 0.2148 ];
Q=10*eye(4);
R=eye(2);

ad=expm(a*0.01);
```

```
bd=inv(a)*(ad-eye(4))*b;

F=[-a Q;zeros(4,4) a'];
G=expm(F*0.01);
Glr=G(5:8,5:8);
Gur=G(1:4,5:8);
Qd=Glr'*Gur;

xpred=ad*x+bd*u;
ppred=ad*P+ad'*Qd;
end
```

Corrector:

```
%% Corrector algorithm
function [xnew,pnew]= corrector(z, xpred, ppred)
R=eye(4);
C = [ 1.0000 -0.0050 0.0000 0.0000;
    0.0000 1.0025 0.0000 -0.0000;
    0.1604 0.0084 0.9612 0.2769;
    0.0000 -0.0049 -0.0011 0.0040 ];
H=C;
k=ppred*H'*inv((H*ppred*H')+R);
pnew=((eye(4)-k*H)*ppred*transpose(eye(4)-k*H))+k*R*k';
xnew=xpred+k*(z-H*xpred);
end
```

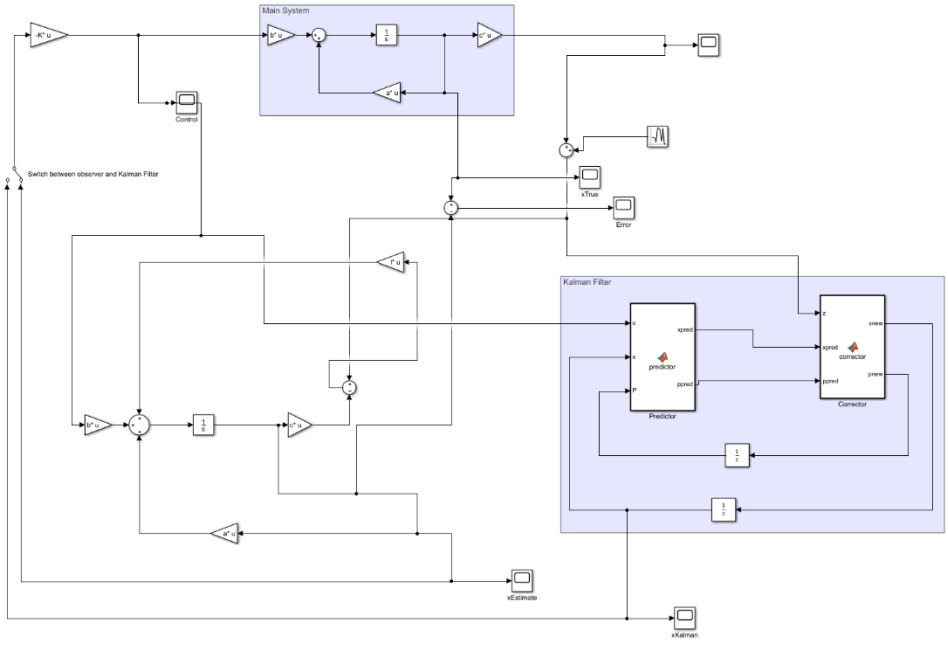

Fig3.1 Simulink diagram when observer states used as feedback

Results:

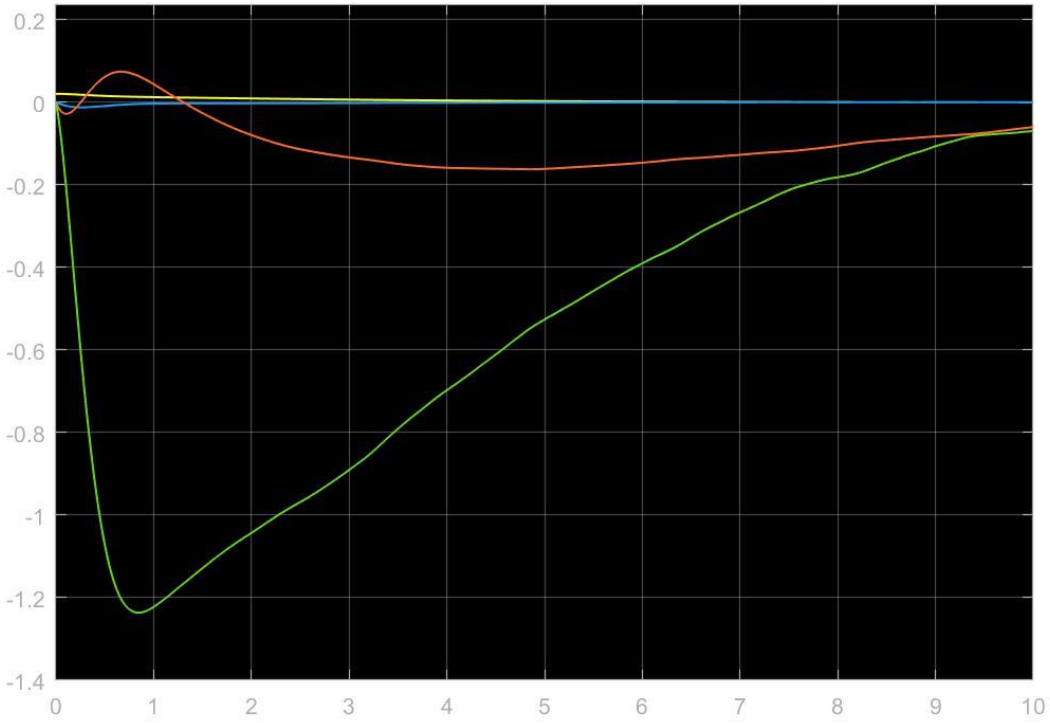

Fig3.2 True states time history

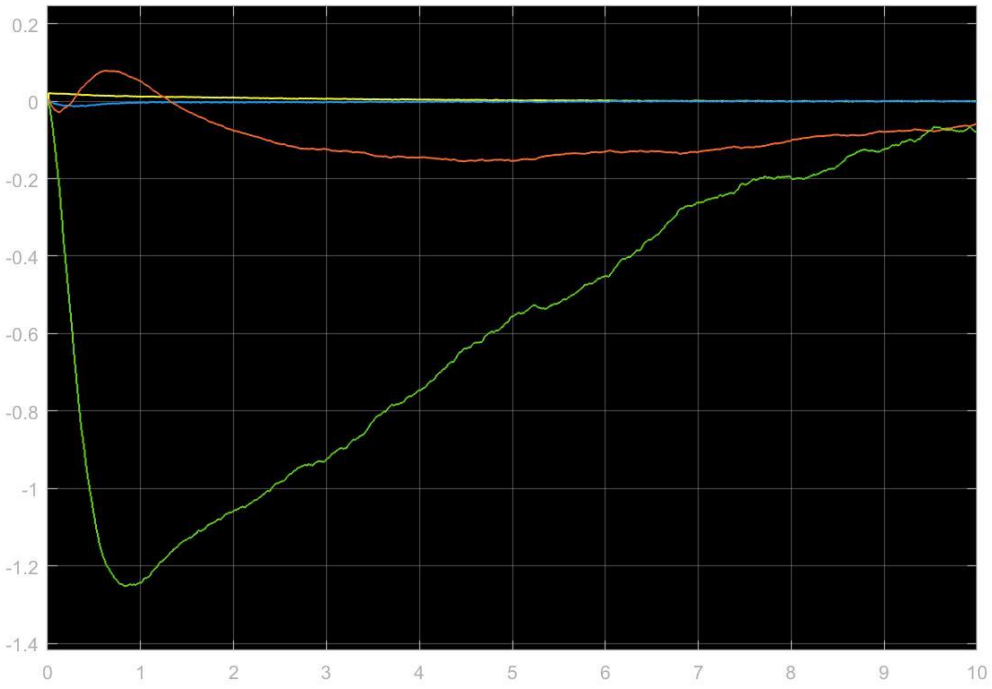

Fig3.3 Kalman filter state estimates

We can observe that the Kalman filter state estimates are very close to the true states.

b) When R = 0.0001*eye(4)

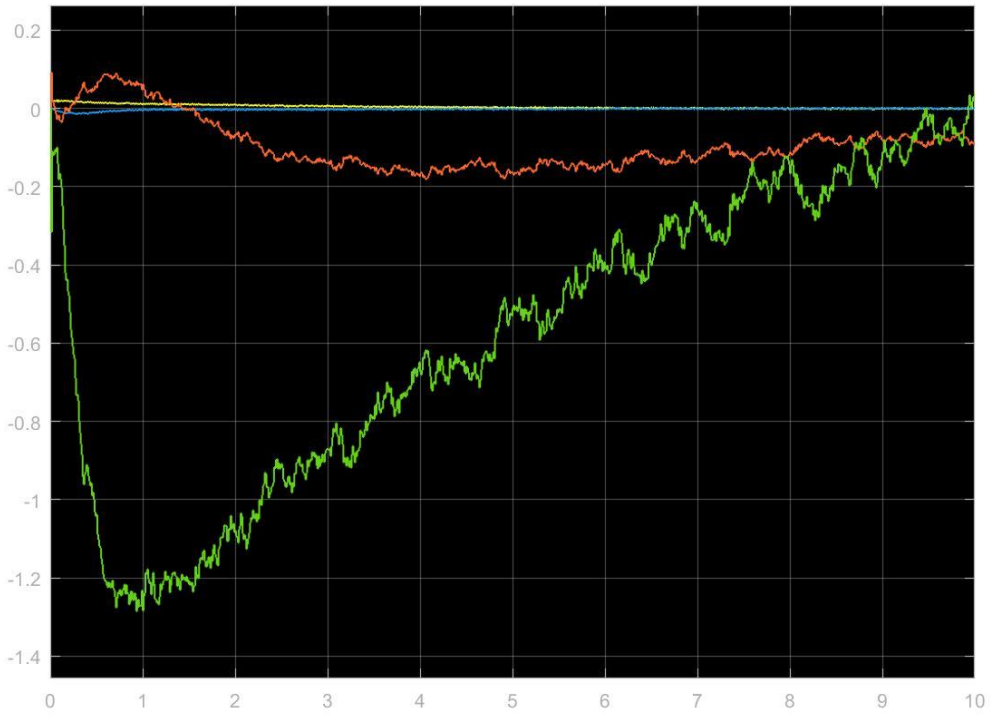

Fig3.4 The Kalman filter estimates when the R changes

We can observe that as the value of R is changed, the estimates get worse more susceptible to noise.

c) Using the Kalman states as the feedback, the following results are obtained when R = eye(4)

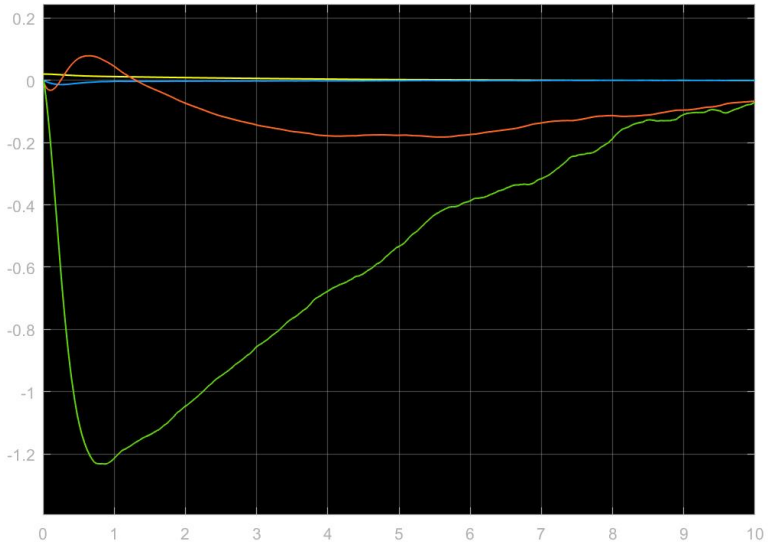

Fig3.5 The true states time history when Kalman states are used as feedback

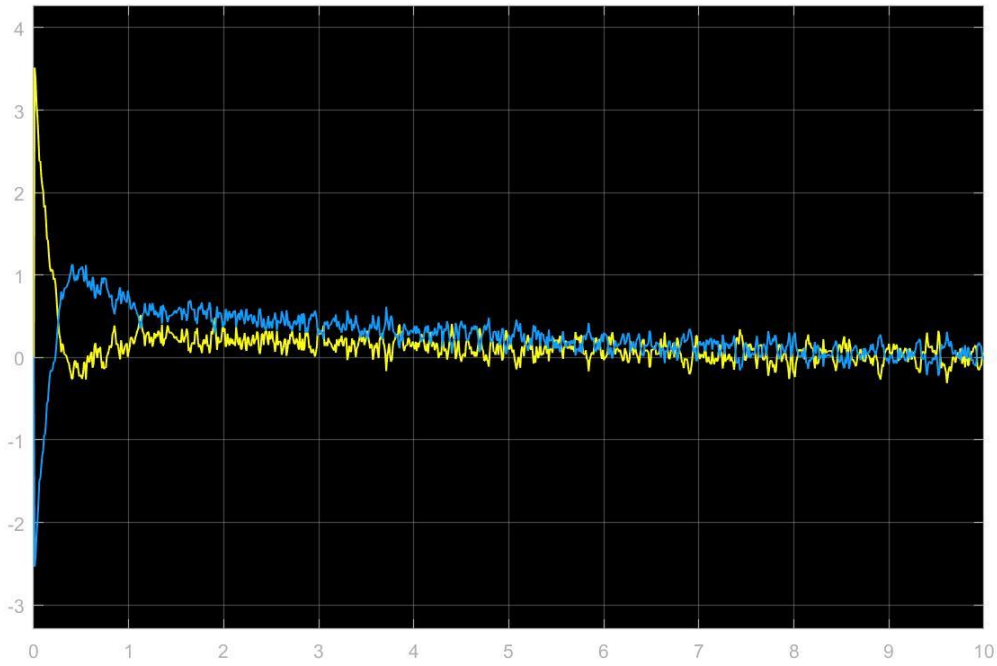

Fig3.6 The control input when Kalman filter states used in feedback

When R = 0.0001*eye(4)

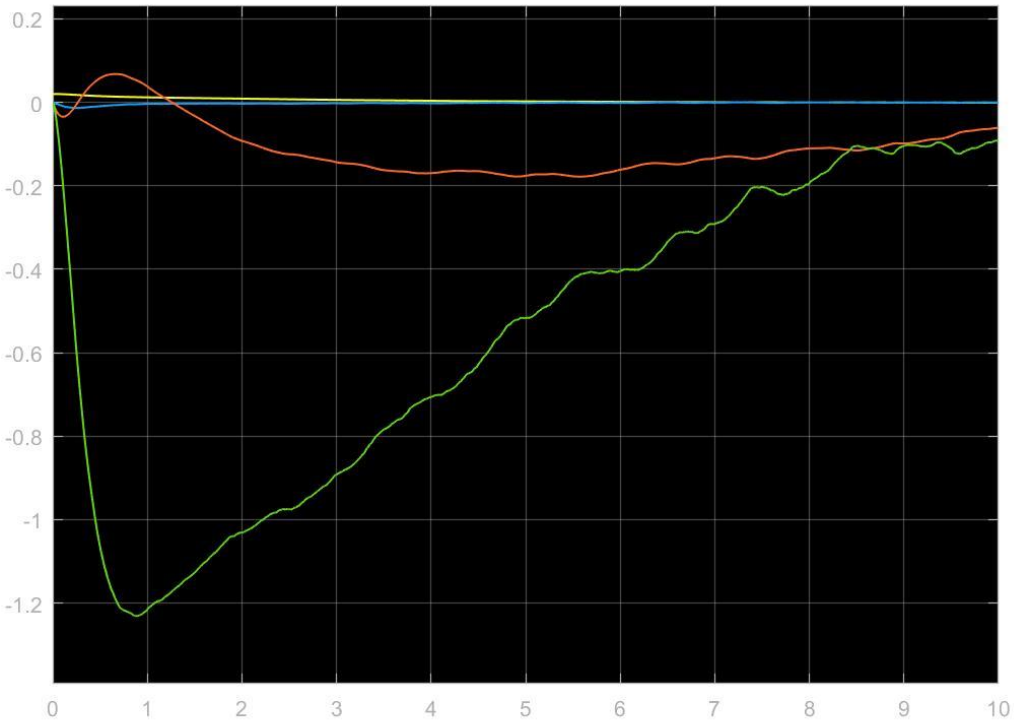

Fig3.7 The true states time history when Kalman states are used as feedback

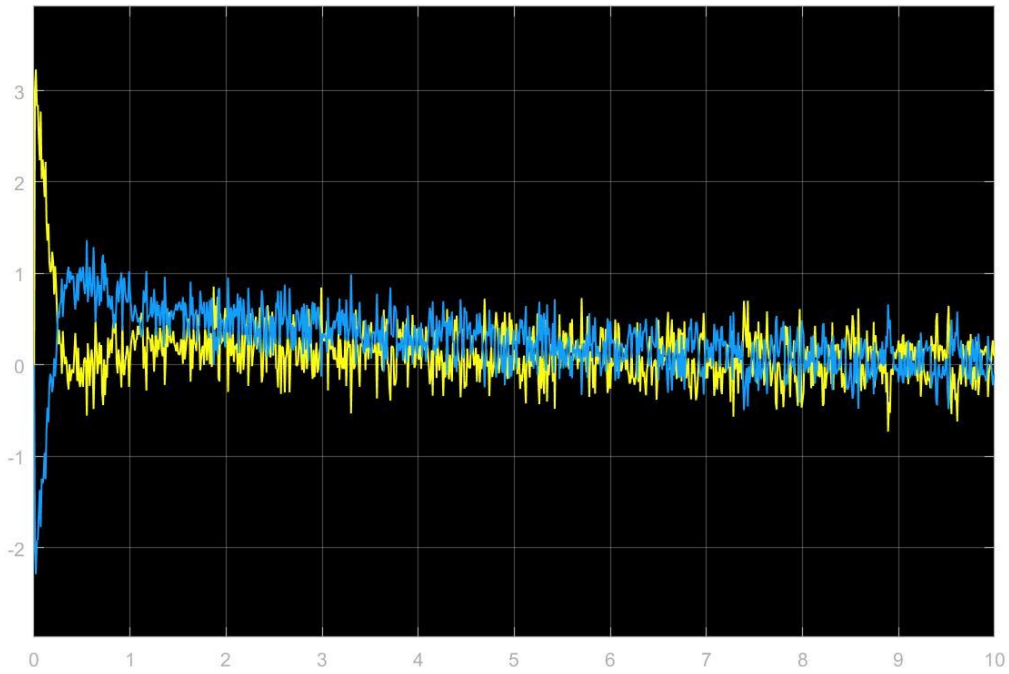

Fig3.8 The control input when Kalman filter states used in feedback

We can observe that when the observer states are used in feedback, the states that are reconstructed are smoother as compared to using the Kalman filter state estimates.

The same thing can be said for the control inputs as well. The control inputs are more susceptible to noise when using Kalman filter state estimates and smoother when using the observer states as feedback.