19/20 ZG

Course Name: Control System Design

Course Number and Section: **14:332:417**

Experiment: Final Project

Date Performed:

Date Submitted: 12/21/2020

Submitted by: Matthew Hanna; RUID: 178001207

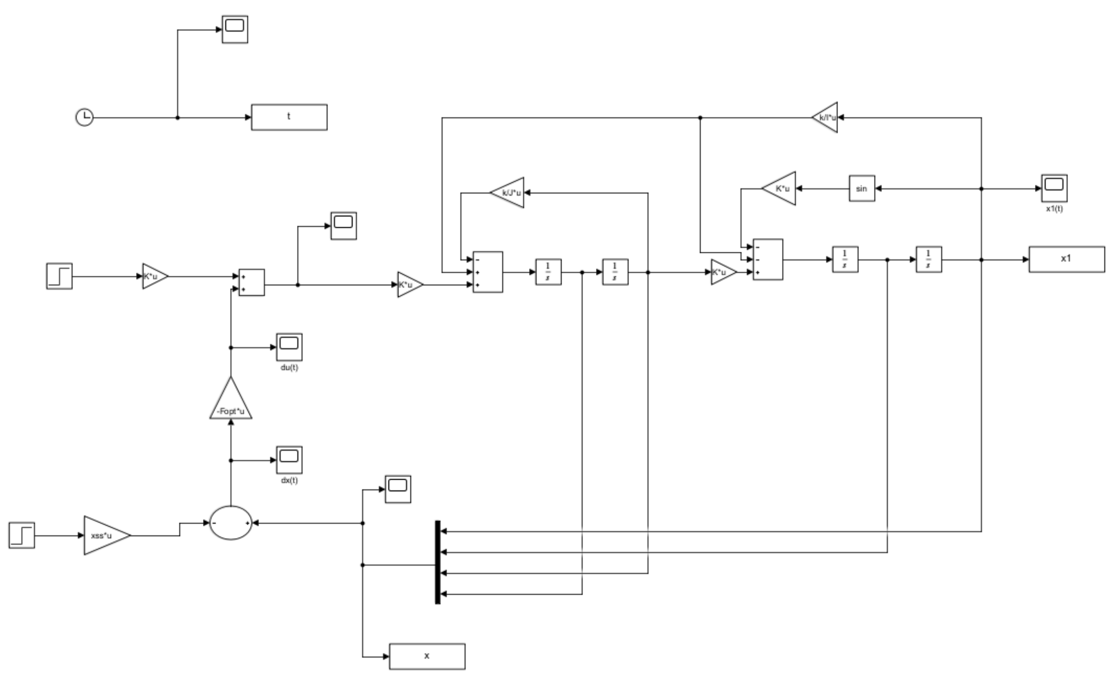
Table of Contents

Project Formulation:

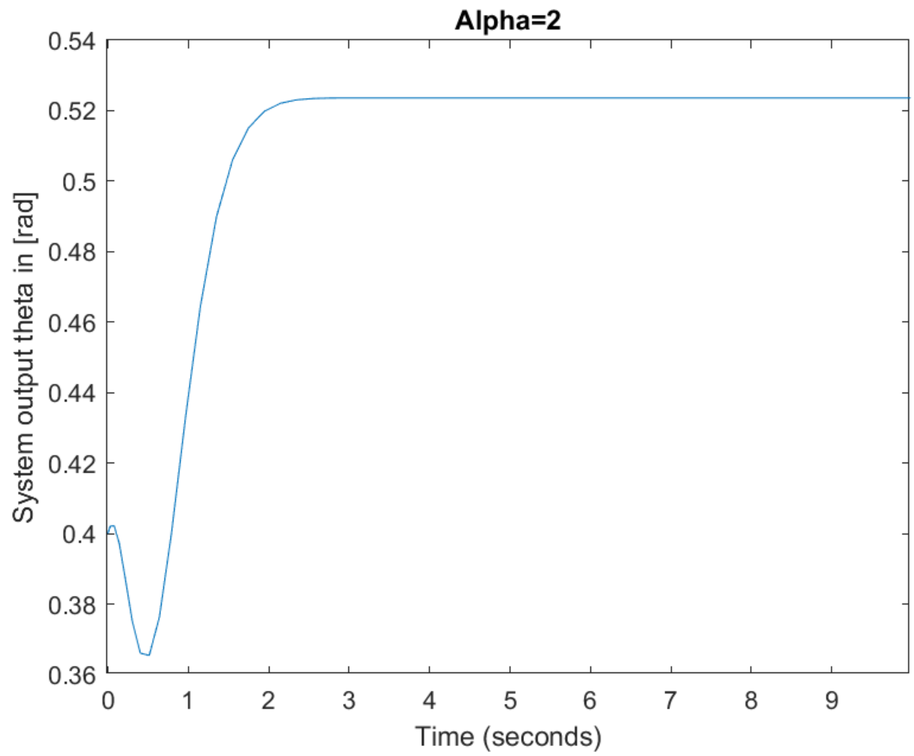
This project is split into two different objectives. The first part deals with utilizing a linearization technique in MATLAB on a robotic manipulator to achieve a steady state angle of $\theta_{1ss} = \pi / 6$. We first considered the effect of a linear-quadratic optimal controller with five stability robustness parameters on the system. The following part repeats the same process, but with an eigenvalue assignment controller. In part 2 of the project, we were tasked with designing an optimal linear-quadratic Gaussian controller driven by a Kalman filter for an aircraft under wind disturbances. After designing the Kalman filter using the given data, we found the optimal performance criterion for the LQG controller given different weighted matrices and analyzed the impact of noise on J_{opt}.

Robotic manipulator

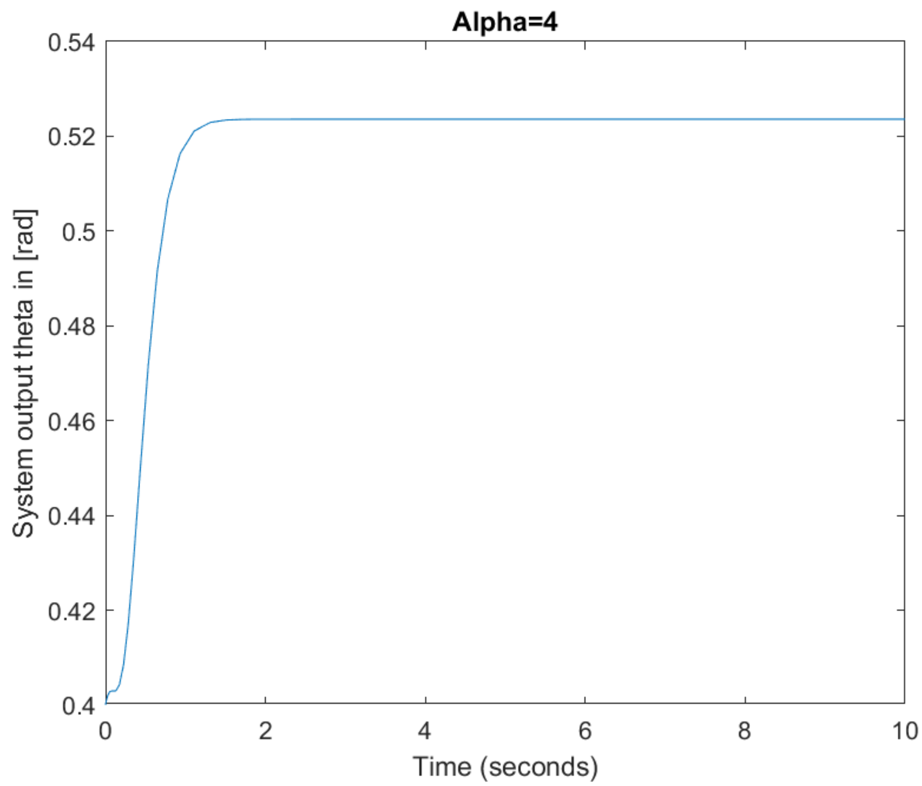
a) Simulink Block Diagram



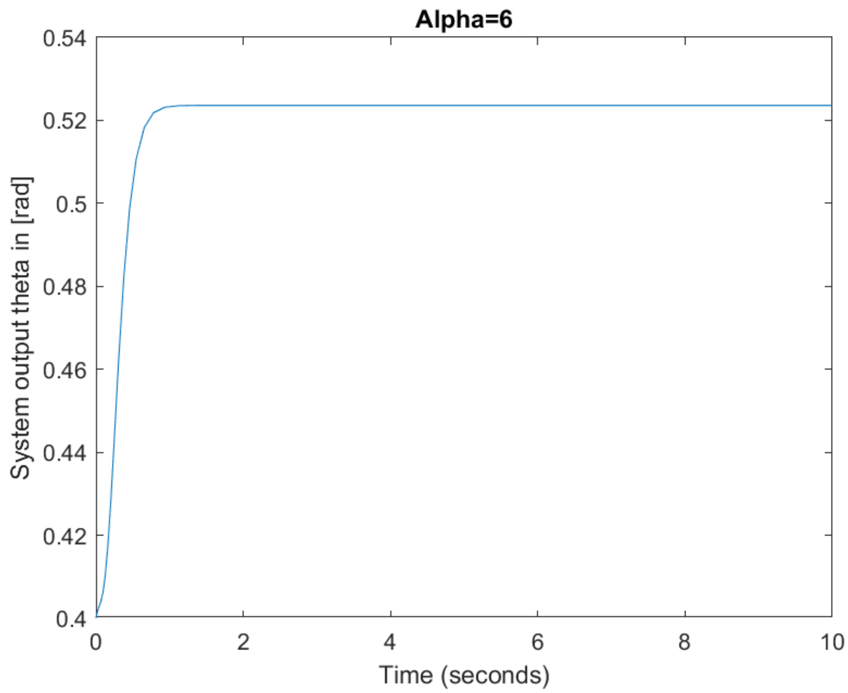
Plots:

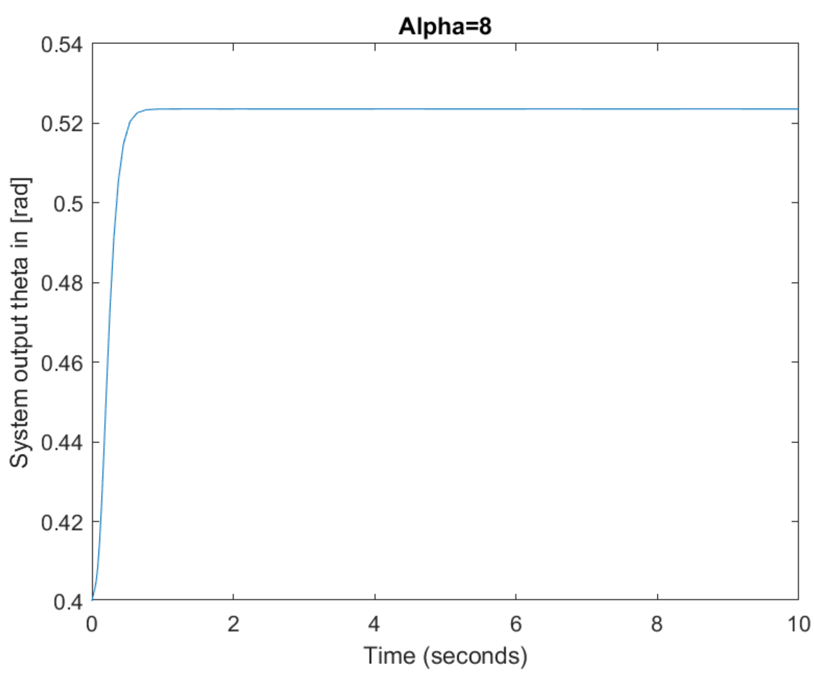


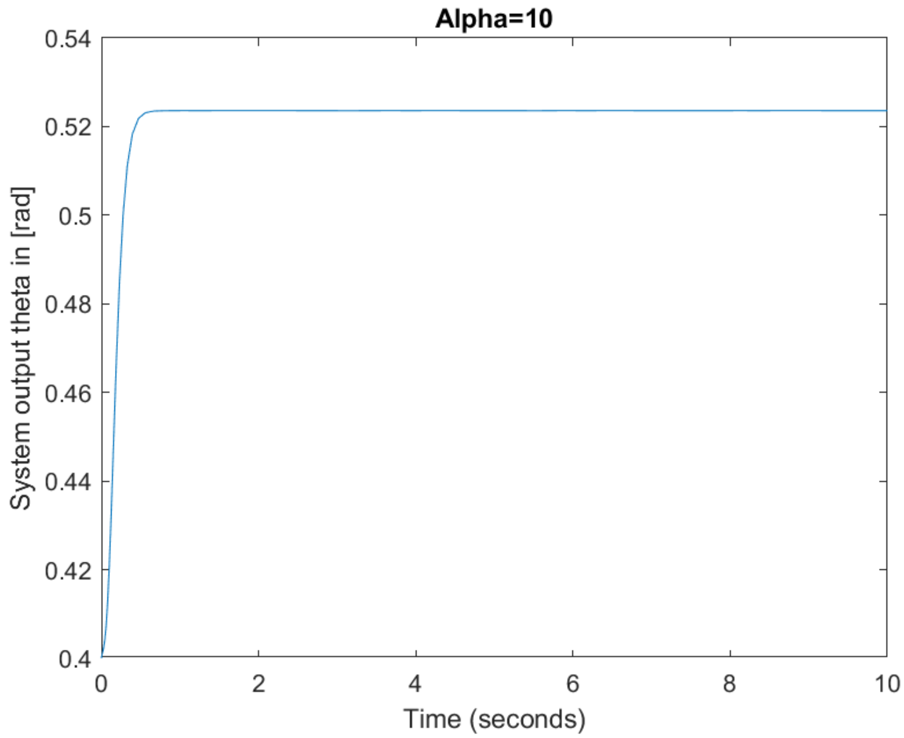
Alpha=2
0.54
0.52
0.5
0.48
0.46
0.44
0.42
0.4
0.38
0.36
System output theta in [rad]
0
1
2
3
4
5
6
7
8
9
Time (seconds)



Alpha=4
0.54
0.52
0.5
0.48
0.46
0.44
0.42
0.4
System output theta in [rad]
0
2
4
6
8
10
Time (seconds)


Alpha=6
0.54
0.52
0.5
0.48
0.46
0.44
0.42
0.4
System output theta in [rad]
0
2
4
6
8
10
Time (seconds)


Alpha=8
0.54
0.52
0.5
0.48
0.46
0.44
0.42
0.4
System output theta in [rad]
0
2
4
6
8
10
Time (seconds)

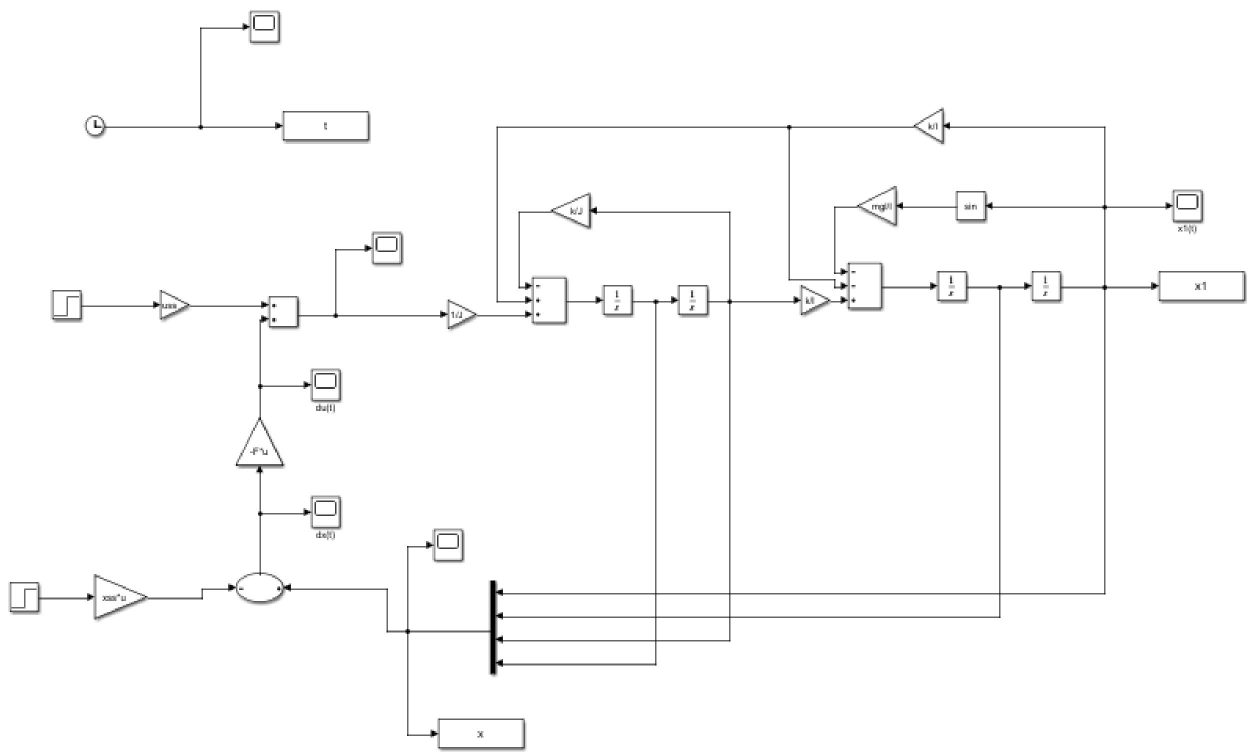


Analysis:

After plotting the five values for the stability robustness parameter, we can observe that at alpha=10, the output response is the most optimal. Compared to the results in the alpha=2 and alpha=4 cases, we see that there is not much of a dip during the first half second of the response. The output response angle at steady state for all the cases is around .523, which is the desired steady state angle, $\theta_{1ss} = \pi / 6$. As we increase the robustness parameter, the output response reaches steady state faster. This can be seen when comparing the alpha=10 and alpha=2 cases. At alpha=10, the output response reaches steady state at around 0.5 seconds. At alpha=2, the steady state occurs at around 2 seconds.

Part 1:

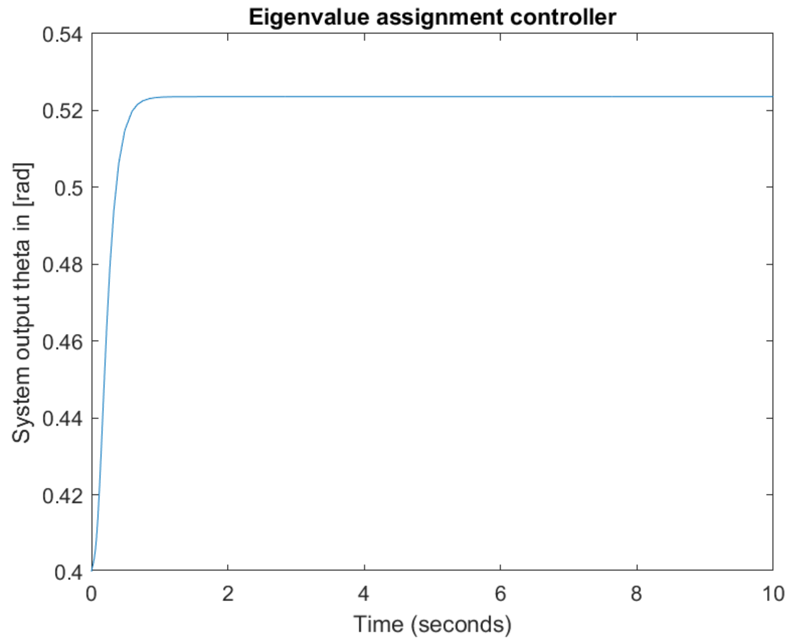
b) Simulink Diagram:



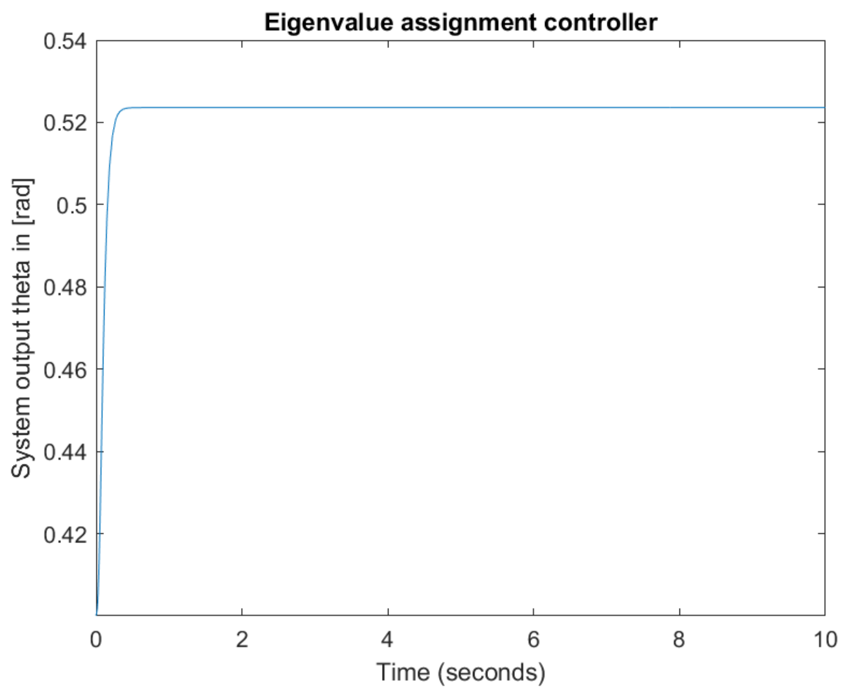
The plots below will examine two different eigenvalue sets. The first is lambda desired equal to the array, [-8 -16 -24 -40]. The second plot used lambda desired equal to the array, [-20 -30 -60 -100]. The first set will give results extremely similar to the linearization technique's best case(alpha=10). The second case as shown below provides even better results than the linearization technique.

Plots:

First case eigenvalue set (lambda= [-8 -16 -24 -40])



Second eigenvalue set (lambda= [-20 -30 -60 -100])

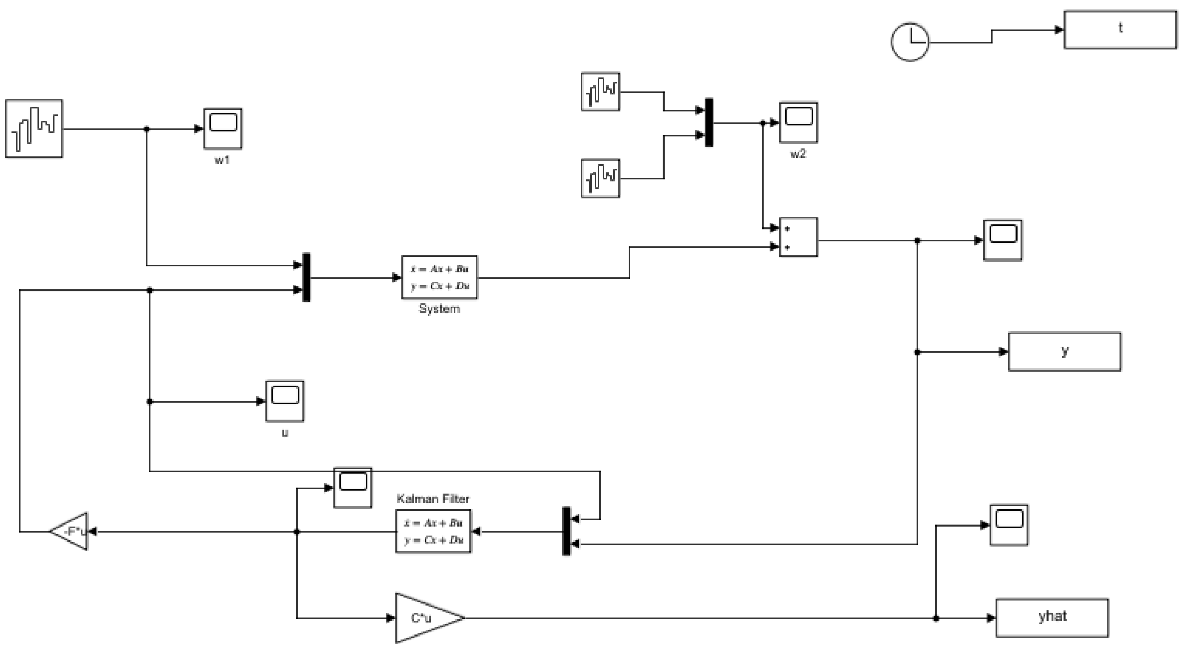


Analysis:

The first eigenvalue set gives us results that are extremely close to that of the linearization technique. Increasing the eigenvalue set in the second case makes the output response reach steady state much faster. Moreover, the second case is generally better, however the eigenvalues are a bit large.

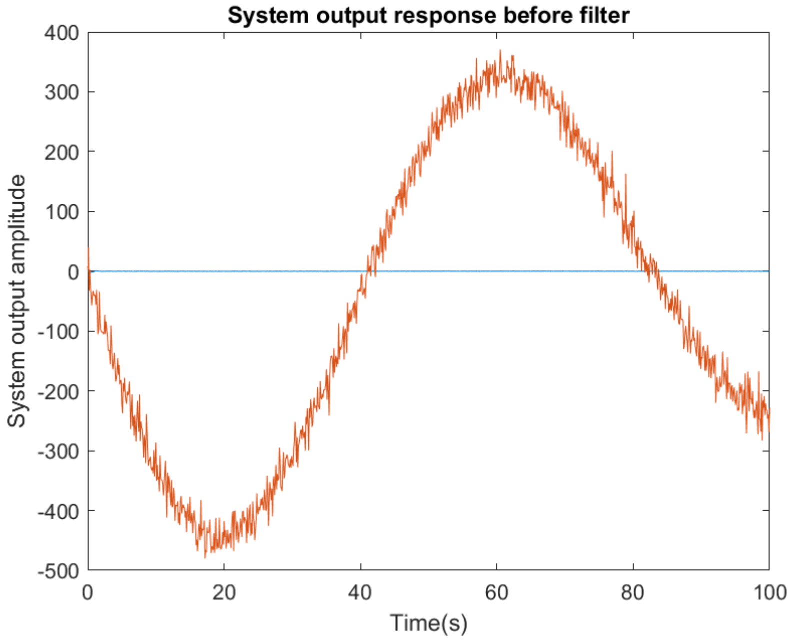
Part 2:

a) Simulink Diagram



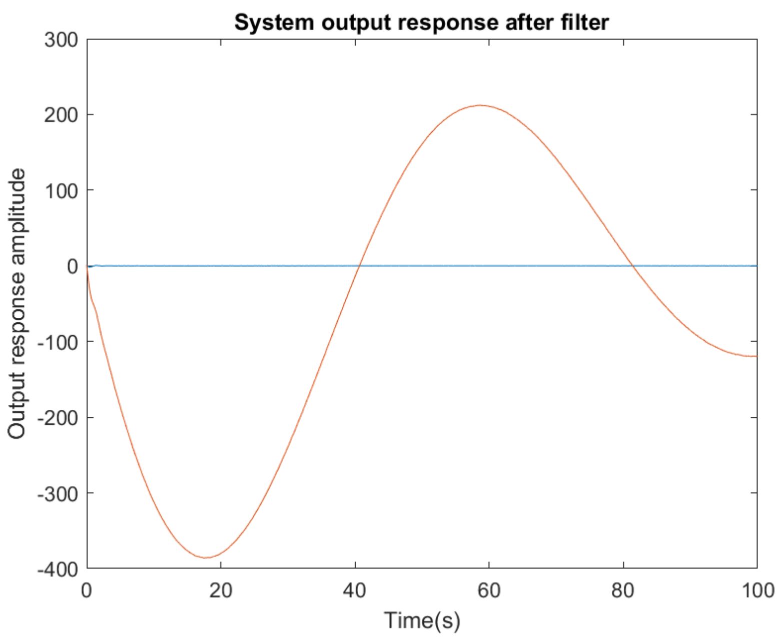
Plots:

System Output response before filter, t=100 seconds:

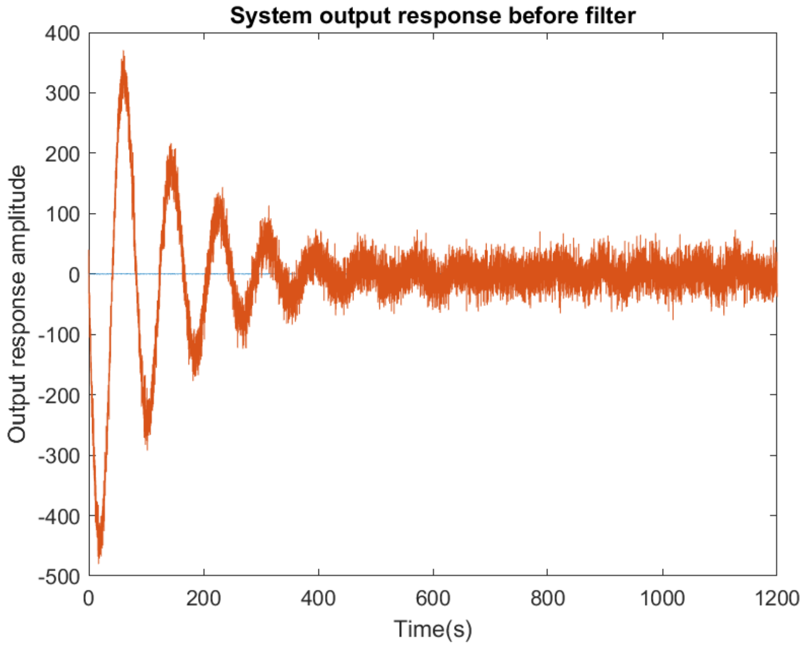


-1 noise power is not properly set

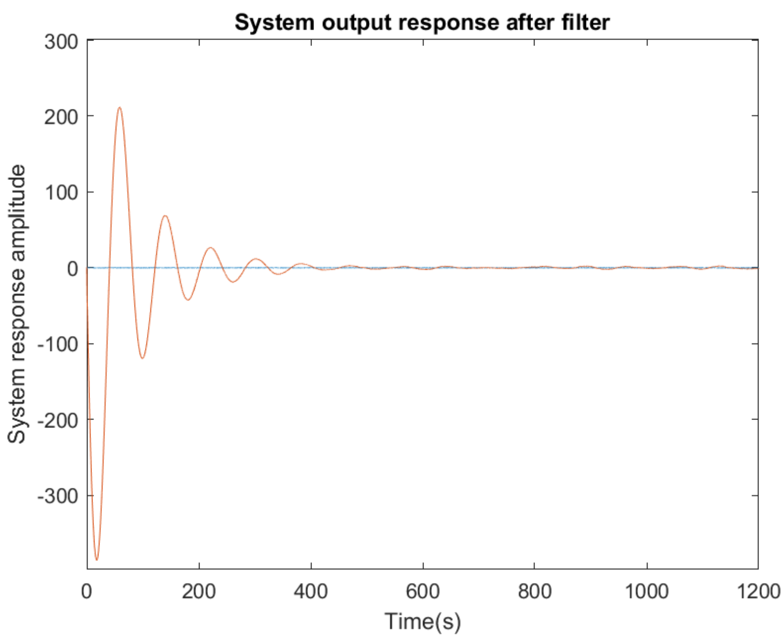
System Output response after filter, t=100 seconds:



System Output response before filter, t=1200 seconds:



System Output response after filter, t=1200 seconds:



Analysis:

The system output response before the filtering for both times is extremely noisy. After putting it through the Kalman filter, the noise is removed, and the output response is optimal. Plotting the output responses at t=1200 shows that the response settles to zero at around 400 seconds with the Kalman filter. However, without the Kalman filter, there is a substantial amount of noise where the output response should be zero. Moreover, the Kalman filter is successful in removing the noise from the system.

Part 2:

b) Optimal performance criterion J_{opt} for the LQG controller

W	*V*	*Jopt*
2	I_2	*1.3342e +03*
2	$0.5 \times I_2$	*1.3337e +03*
0.5	*2I*	*334.475*
1	I_2	*667.4685*
0.5	*0.5*I*	*333.7342*

Analysis:

The optimal performance criterion depends more on W than V. When W is lower, Jopt increases drastically. For example, when W=0.5, Jopt is 334.475. When W is 1, Jopt is almost double that at 667.4685. When W is larger comparatively, Jopt is extremely small at around 1.3342e+03.

Appendix:

Part 1:

a)

Code:

```
n=4; m=1; r=1; % dimensions of the state, input, and output variables
% Set point controller
I=1; J=1; k=0.08; mgl=5;
x1ss=pi/6; x2ss=0; x3ss=x1ss+(mgl/k)*sin(pi/6); x4ss=0
% MATLAB functions sine in cos provide results in radians
yss=x1ss;
xss=[x1ss; x2ss; x3ss; x4ss]
uss=mgl*sin(x1ss)
% linearized dynamics controller
A=[0 1 0 0;-(k+mgl)*cos(x1ss)/I 0 0.08 0; 0 0 0 1; k/J 0 -k/J 0];
B=[0; 0; 0; 1/J];
C=[1 0 0 0]; D=0;
R1=eye(4); R2=1;
 % prescribed stability robustness parameter
for alpha=[2 4 6 8 10]
CM=ctrb(A+alpha*eye(4),B)
rank (CM); % It should be equal to n
OM=obsv(A+alpha*eye(4),R1)
rank(OM); % It should be equal to n

% LQ with the alpha-prescribed degree of stability
[Fopt,P]=lqr(A+alpha*eye(4),B,R1,R2)
% can use also the eigenvalue assignment controller with selected
% desired closed-loop eigenvalues, for example
% lambda_desired=[-5 -6 -7 -8]
% F=place(A,B,lambda_desired)
% Setting up the integrator initial conditions xi0, i=1,2,3,4
% initial conditions should be closed to their steady state values
% assume that the system initial conditions are:
x10=0.4; x20=0.1; x30=32; x40=0.1;
% Comments: Angles x1, x3 are in radians. Since the angles are2*pi %
periodic, they can be transformed to the [0 2*pi] rad range using
x3ss_rad=mod(x3ss,2*pi)
% conversion from radians to degrees: angle [deg]=(180/pi)*angle[rad]
x3ss_deg=mod(x3ss*180/pi,360)
x30=mod(x30,2*pi)
sim('simfinalparta')
figure(alpha-1);
plot(x1)
end
```

b)

Code:

```
n=4; m=1; r=1; % dimensions of the state, input, and output variables
% Set point controller
I=1; J=1; k=0.08; mgl=5;
x1ss=pi/6; x2ss=0; x3ss=x1ss+(mgl/k)*sin(pi/6); x4ss=0
% MATLAB functions sine in cos provide results in radians
yss=x1ss;
xss=[x1ss; x2ss; x3ss; x4ss]
uss=mgl*sin(x1ss)
% linearized dynamics controller
A=[0 1 0 0;-(k+mgl)*cos(x1ss)/I 0 0.08 0; 0 0 0 1; k/J 0 -k/J 0];
B=[0; 0; 0; 1/J];
C=[1 0 0 0]; D=0;
R1=eye(4); R2=1;
 % prescribed stability robustness parameter
for alpha=[2 4 6 8 10]
CM=ctrb(A+alpha*eye(4),B)
rank (CM); % It should be equal to n
OM=obsv(A+alpha*eye(4),R1)
rank(OM); % It should be equal to n

% LQ with the alpha-prescribed degree of stability
%[Fopt,P]=lqr(A+alpha*eye(4),B,R1,R2)
% can use also the eigenvalue assignment controller with selected
% desired closed-loop eigenvalues, for example

lambda_desired=[-8 -16 -24 -40]
%lambda_desired=[-20 -30 -60 -100]
F=place(A,B,lambda_desired)
% Setting up the integrator initial conditions xi0, i=1,2,3,4
% initial conditions should be closed to their steady state values
% assume that the system initial conditions are:
x10=0.4; x20=0.1; x30=32; x40=0.1;
% Comments: Angles x1, x3 are in radians. Since the angles are2*pi %
periodic, they can be transformed to the [0 2*pi] rad range using
x3ss_rad=mod(x3ss,2*pi)
% conversion from radians to degrees: angle [deg]=(180/pi)*angle[rad]
x3ss_deg=mod(x3ss*180/pi,360)
x30=mod(x30,2*pi)
sim('simfinalpartb')
plot(x1)
end
```

Part 2:

a)

```
n=4; m1=1; m2=1; r=2  % dimensions of the state, control, system noise,
output and  measurement noise
A=[-0.01357  -32.2  -46.3  0; 0.00012  0  1.214  0; -0.0001212  0  -1.214  1;
0.00057  0  -9.01  -0.6696]
B=[-0.433; 0.1394; -0.1394; -0.1577]
G=[-46.3; 1.214; -1.214; -9.01]
W=[0.000315]; V=[0.000686 0; 0 40] % system and measurements white noise
intensities
C=[0 0 0 1;1 0 0 0]; D=zeros(2,1)
x0bar=[1 1 1 1]'; % assumed system initial condition mean value
R1=diag([0.001,0, 3260, 3260]); R2=3260; %weighted matrices for the quadratic
performance criterion
% Existence conditions for the regulator algebraic Riccati equation
CM_lqr=ctrb(A,B); rank(CM_lqr)
OM_lqr=obsv(A,R1); rank(OM_lqr)
% Existence conditions for the filter algebraic Riccati equation
CM_lqe=ctrb(A,G); rank(CM_lqe)
OM_lqe=obsv(A,C); rank(OM_lqe)
% Optimal control gain and the optimal (minimal) value of the performance
criterion
[F,P]=lqr(A,B,R1,R2);
[K,Q]=lqe(A,G,C,W,V)
Jopt=trace(P*G*W*G'+Q*F'*R2*F)

figure(1)
plot(t,y)
figure(2)
plot(t,yhat)
```

b)

```
n=4; m1=1; m2=1; r=2  % dimensions of the state, control, system noise,
output and  measurement noise
A=[-0.01357  -32.2  -46.3  0; 0.00012  0  1.214  0; -0.0001212  0  -1.214  1;
0.00057  0  -9.01  -0.6696]
B=[-0.433; 0.1394; -0.1394; -0.1577]
G=[-46.3; 1.214; -1.214; -9.01]
W=0.5; V=0.5*eye(2) % system and measurements white noise intensities
C=[0 0 0 1;1 0 0 0]; D=zeros(2,1)
x0bar=[1 1 1 1]'; % assumed system initial condition mean value
R1=eye(4); R2=1; %weighted matrices for the quadratic performance criterion
% Existence conditions for the regulator algebraic Riccati equation
CM_lqr=ctrb(A,B); rank(CM_lqr)
OM_lqr=obsv(A,R1); rank(OM_lqr)
% Existence conditions for the filter algebraic Riccati equation
CM_lqe=ctrb(A,G); rank(CM_lqe)
OM_lqe=obsv(A,C); rank(OM_lqe)
% Optimal control gain and the optimal (minimal) value of the performance
criterion
[F,P]=lqr(A,B,R1,R2);
[K,Q]=lqe(A,G,C,W,V)
Jopt=trace(P*G*W*G'+Q*F'*R2*F)
```