Control Theory Home Work 5

Utkarsh Kalra BS18-03 u.kalra@innopolis.university

Varient g

1 Git repo

https://github.com/kalraUtkarsh/Control-Theory-Utkarsh-Kalra

2 Observers

We have already already defined the dynamics and other things in the previous homework as the system is the same as in the previous one.
The dynamics:
$(M+m)x'' - mlcos(\theta)\theta'' + mlsin(\theta)\theta^2$
$\theta'' - cos(\theta)x'' - gsin(\theta) = 0$

The system in the state space form:
$z' = f(z) + g(z)u$
$y = h(z) = \begin{bmatrix} x & (\theta)^2 \end{bmatrix}^T$
$wherez = \begin{bmatrix} x & (\theta) & x' & (\theta)' \end{bmatrix}^T$
And as done on the previous homework in task 2(B)

$$\begin{bmatrix} x' \\ \theta' \\ x'' \\ \theta'' \end{bmatrix} = \begin{bmatrix} x' \\ \theta' \\ \frac{-mlsin(\theta)\theta'^2 + mgsin(\theta)cos(\theta)}{(M+m) - mcos^2(\theta)} \\ \frac{-(M+m)gsin(\theta) + mlcos(\theta)sin(\theta)\theta'^2}{mlcos^2(\theta) - (M+m)l} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{1}{(M+m) - mcos^2(\theta)} \\ \frac{cos(\theta)}{(M+m) - mcos^2(\theta)} \end{bmatrix}$$

And as done in the 2(C) of the last homework A =

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{2m\cos(\theta)\sin(\theta)(ml\dot{\theta}^2\sin(\theta) - mg\cos(\theta)\sin(\theta))}{(M+m-m\cos^2(\theta))^2} - \frac{mg\sin^2(\theta) - mgcos^2(\theta) + 2ml\dot{\theta}^2\cos(\theta)}{M+m-m\cos^2(\theta)} & 0 & -\frac{2ml\dot{\theta}\sin(\theta)}{M+m-m\cos^2(\theta)} \\ 0 & \frac{(M+m)g\cos(\theta) - 2ml\dot{\theta}\cos^2(\theta) + 2ml\dot{\theta}\sin^2(\theta)}{l(M+m-m\cos^2(\theta))} - \frac{2m\cos(\theta)\sin(\theta)(-ml\dot{\theta}^2\cos(\theta)\sin(\theta) + (M+m)g\sin(\theta))}{l(M+m-m\cos^2(\theta))^2} & 0 & -\frac{2m\dot{\theta}\cos(\theta)\sin(\theta)}{M+m-m\cos^2(\theta)} \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{M+m-m\cos^2(\theta)} \\ \frac{\cos(\theta)}{l(M+m-m\cos^2(\theta))} \end{bmatrix}$$

B =
My varient **g**, M=11.6, m=2.7, l=0.57

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 2.2810 & -0.0862 & 0 \\ 0 & -207.7090 & -0.15124 & 0 \end{bmatrix} B = \begin{bmatrix} 0 \\ 0 \\ 0.0862 \\ 0.15124 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

2.1 2(A)

for this we have to check the rank of the observability matrix, and check the number of unobservable states. the matrix has full rank and the unobservable states comes to be 0 in matlab, so the system is **OBSERVABLE**.

2.2 2(B)

For this we have to calculate the eigenvalues of A, and if all the values are negative then the system is stable and eigenvalues of A in my case comes out to be:

```
 0.0000 + 0.0000i
 0.0008 +14.4121i
 0.0008 -14.4121i
-0.0879 + 0.0000i
```

not all values are negative so the system is **not stable**

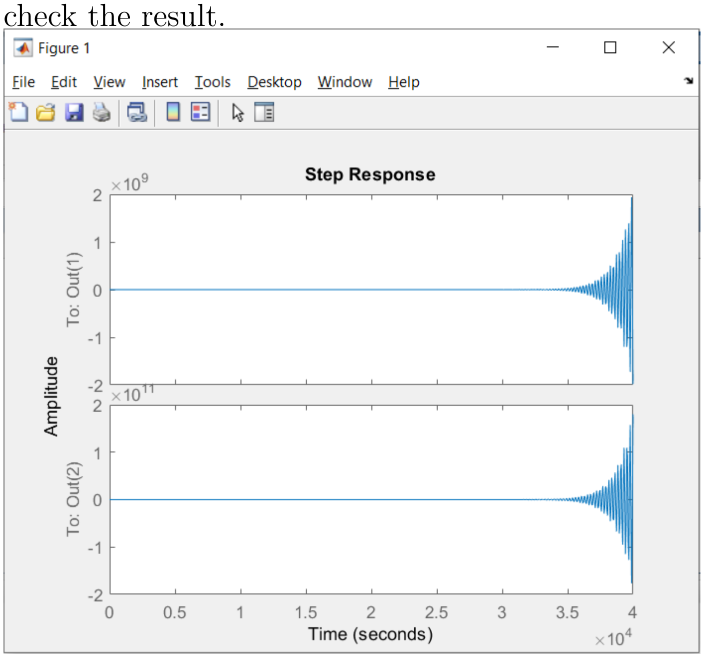
2.3 2(C)LUENBERGER OBSERVER

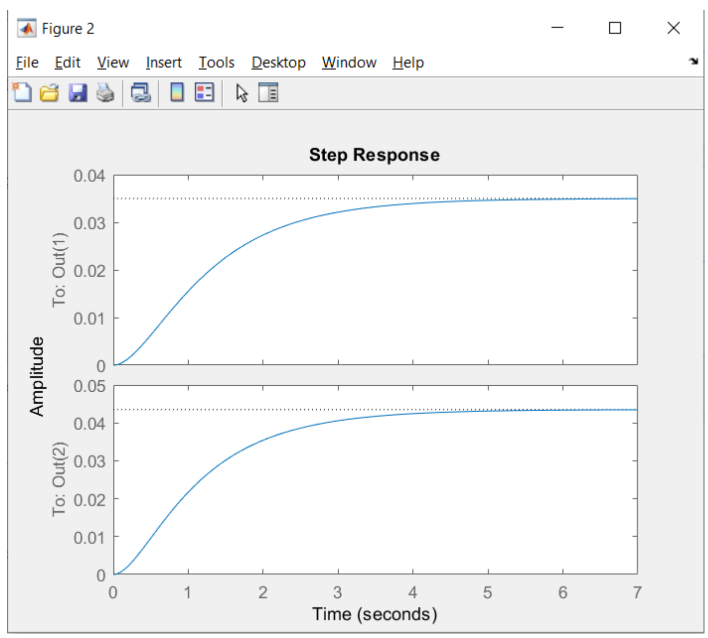
Pole placement: there will be the estimation state $\hat{z}$, error by e = z -$\hat{z}$

$$\dot{\hat{z}} = A\hat{z} + Bu + L(y - \hat{y})$$
$$\hat{y} = C\hat{z}$$

$$\dot{z} - \dot{\hat{z}} = A(z - \hat{z}) + LC(z - \hat{z})$$
$$\dot{e} = (A - LC)e$$

the error is given by the poles A-LC, As A is 4x4 matrix we have to have 4 poles, get the L matrix and check the result.



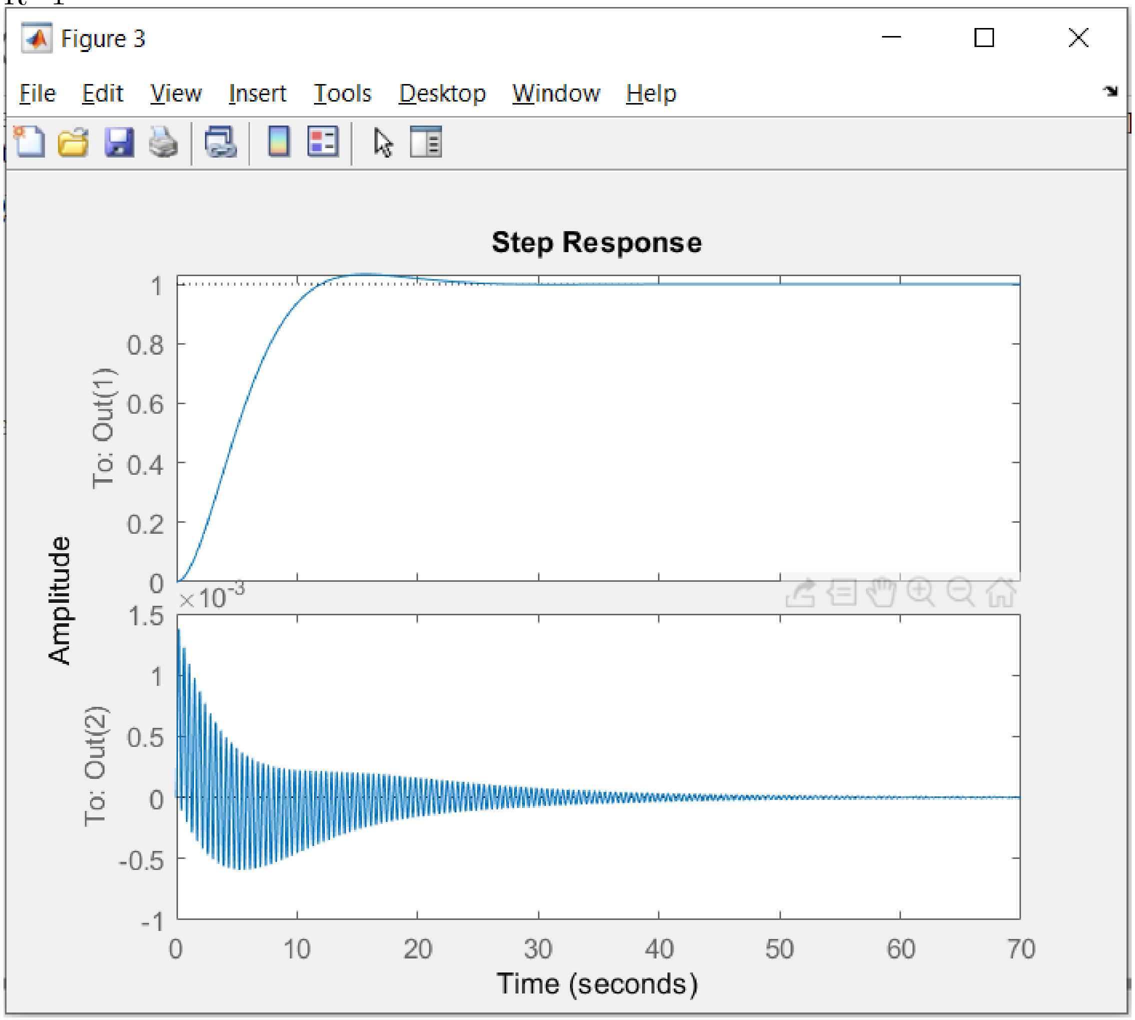


The first figure shows the unstable system system and the second figure shows the stable after adding the observer still its not staedy yet.

now doing the LQR

for this we need the Q and R matrices and they both will be 1 $Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

R=1



this figure shows that the system does become stable after some time but still fluctuates.

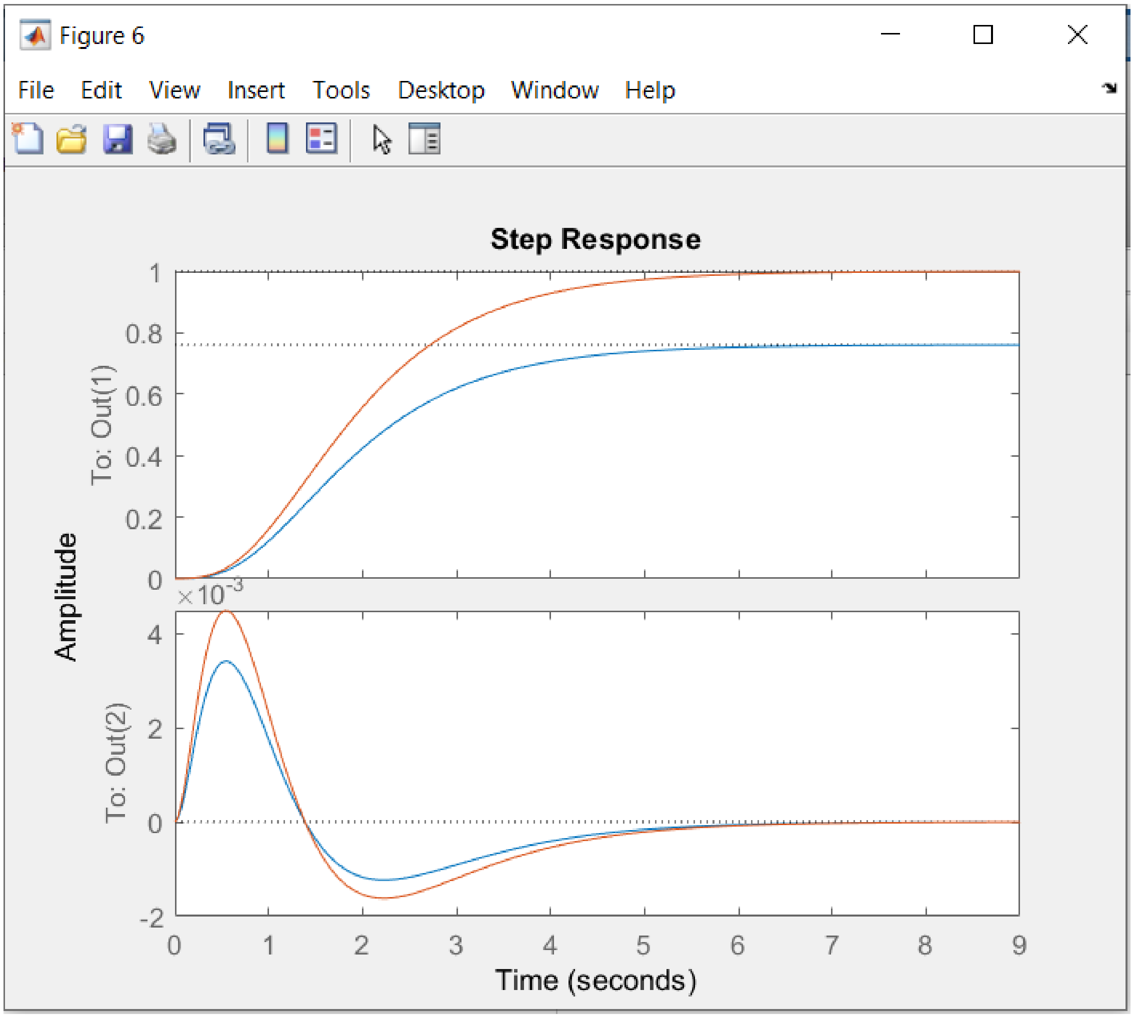
2.4 2(D)State Feedback Controller

it is denoted by: u= r$k_r - k_z$
so the system is
$\dot{z} = (A - BK)z + Brk_r = A_[d]z + Brk_R$
$y = Cz$
$u = rk_r - Kz$
using pole placement method for designing the state feedback controller

```
p1=-1;
p2=-2;
p3=-3;
p4=-4;
A=[0 0 1 0; 0 0 0 1; 0 2.2810 -0.0862 0; 0 -207.7090 -0.15124 0];
B=[0;0;0.0862;0.15124];
C=[1 0 0 0;0 1 0 0];
D=[0;0];
K = place(A, B,[p1 p2 p3 p4]);
Acl = A-B*K;

unscaled_system=ss(Acl,B,C,D);
figure(6);
step(unscaled_system);

Kdc = dcgain(us);
Kr = 1/Kdc(1, 1);
scaled_sys=ss(Acl,B*Kr,C,D);
step(scaled_sys);
figure(5);
legend('Unscaled','Scaled');
disp(Kdc);
disp(Kr);
```

mand Window

```
  0.7604
  0.0000

  1.3151
```



Here both the scaled and unscaled systems follow kind of a similar pattern, yet for scaled the gain kr helps reach the steady state of the system.

2.5 2(E)Luenberger and State Feedback Controller

Here we are supposed to have both observer and controller in the system:
$\hat{z} = A\hat{z} + Bu + L(y - \hat{y})$
$\hat{y} = C\hat{z}$
$u = rk_r - K\hat{z}$
So
$\dot{\hat{z}} = A\hat{z} + B(rk_{[}r] - K\hat{z}) + L(z - \hat{z})\dot{e} = (A - BK - LC)e$

```
p1=-1;
p2=-2;
p3=-3;
p4=-4;
A=[0 0 1 0; 0 0 0 1; 0 2.2810 -0.0862 0; 0 -207.7090 -0.15124 0];
B=[0;0;0.0862;0.15124];
C=[1 0 0 0;0 1 0 0];
D=[0;0];
L = place(A',C',[p1 p2 p3 p4])';

K = place(A,B,[p1 p2 p3 p4]);

unscaled_ob_con_sys = ss(A-B*K-L*C,B,C,D);
figure(7);
step(unscaled_ob_con_sys);
hold on
Kdc = dcgain(unscaled_ob_con_sys);
Kr = 1/Kdc(1, 1);
scaled_os_con_sys=ss(A-B*K-L*C,B*Kr,C,D);
step(scaled_sys);

legend('Unscaled','Scaled');
disp(Kdc);
disp(Kr);
```

Command Window

```
>> hw_5_2
   -0.0009
   -0.0012

 -1.0580e+03
```

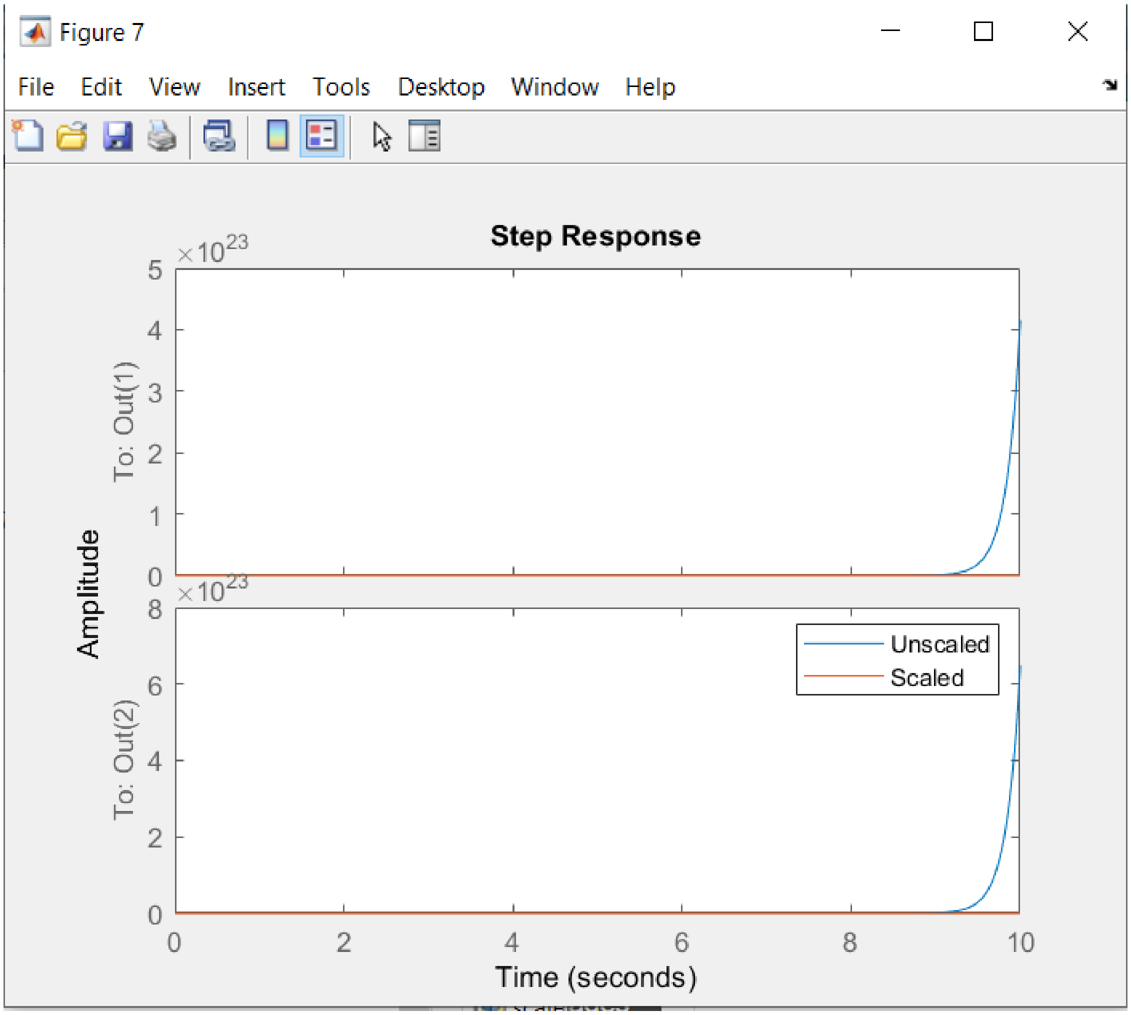
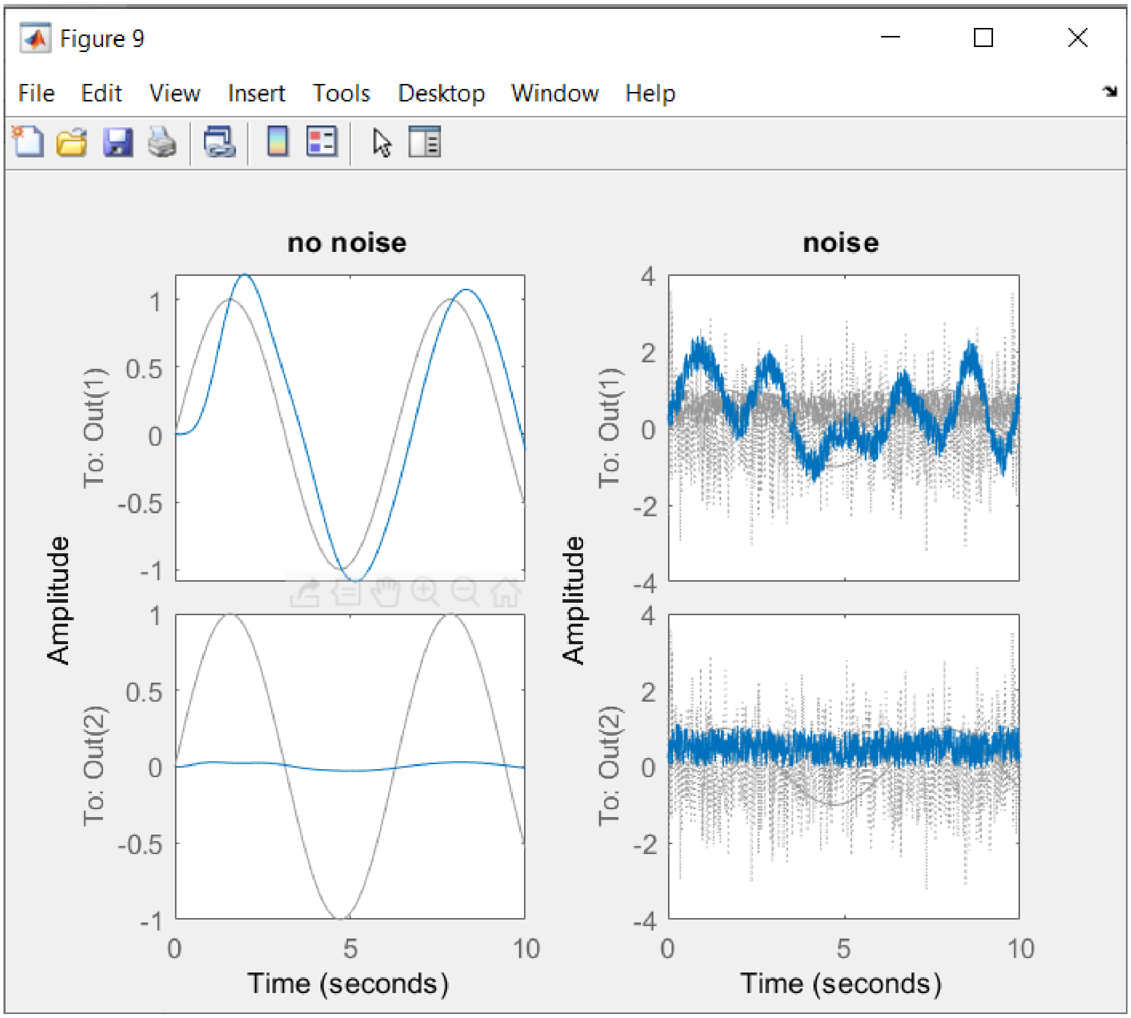


Figure 7
File Edit View Insert Tools Desktop Window Help
Step Response
$\times 10^{23}$
To: Out(1)
Amplitude
To: Out(2)
Unscaled
Scaled
Time (seconds)

2.6 2(F)Gaussian Noise

Gaussian noise is s statistical noise having a probability density function equal to that of the normal distribution, which is also known as the Gaussian distribution.
And now we need to add that to the output, so the system becomes:
$\dot{z} = Az + bu$
$y = Cz + v$
We have to consider the noise as a new input so naturally B and D matrices have to be changed.
$u_a = \begin{bmatrix} u & v \end{bmatrix}^T$
$\dot{z} = Az + \begin{bmatrix} B & 0 \end{bmatrix} \begin{bmatrix} u & v \end{bmatrix}^T = Az + Bu$
$y = Cz + \begin{bmatrix} D & 1 \end{bmatrix} \begin{bmatrix} u & v \end{bmatrix}^T = Cz + v$
Now let the noise v be some random vector.

```

K = place(A,B,[p1 p2 p3 p4]);

unscaled_ob_con_sys = ss(A-B*K-L*C,B,C,D);

Kdc = dcgain(unscaled_ob_con_sys);
Kr = 1/Kdc(1, 1);
scaled_os_con_sys=ss(A-B*K-L*C,B*Kr,C,D);

Acl = A-B*K-L*C;
Bcl=B*Kr;
Ba = [Bcl [0;0;0;0]];
rng default;
Da = [D [1;1]];
t = 0 : 0.01 : 10;
u = sin(t);
v = rand(1, length(t));
ua=[u; v];
sys = ss(Acl,Ba,C,Da);
figure(8);
subplot(1,2,1);
lsim(scaled_os_con_sys, u, t);
title('no noise');
subplot(1,2,2);
lsim(sys, ua, t);
title('noise');
```



Now the graph is super noisy

2.7 2(G)

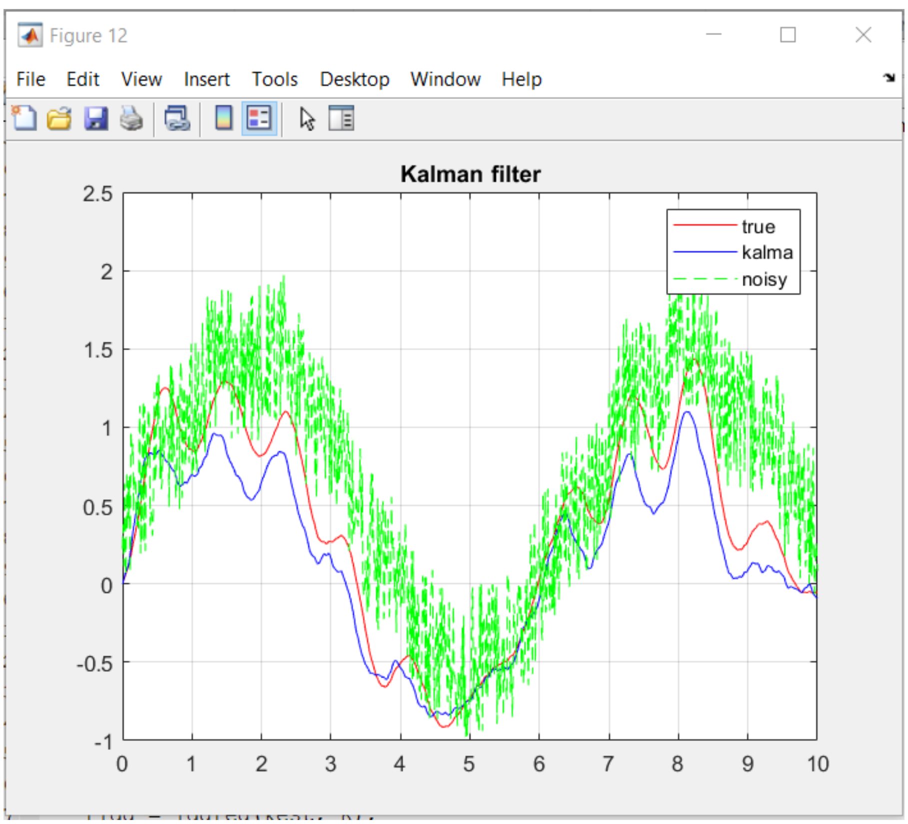
$\dot{z} = Az + bu + w$
$y = Cz + v$
similar steps as in the previous part.
$u_a = \begin{bmatrix} u & v & w \end{bmatrix}^T$
$\dot{z} = Az + \begin{bmatrix} B & 0 & 1 \end{bmatrix} \begin{bmatrix} u & v & w \end{bmatrix}^T = Az + Bu + w$
$y = Cz + \begin{bmatrix} D & 1 & 0 \end{bmatrix} \begin{bmatrix} u & v & w \end{bmatrix}^T = Cz + v$

```
Kdc = dcgain(unscaled_ob_con_sys);
Kr = 1/Kdc(1, 1);
scaled_os_con_sys=ss(A-B*K-L*C,B*Kr,C,D);

Acl = A-B*K-L*C;
Bcl=B*Kr;
Ba = [Bcl [0;0;0;0] [1;1;1;1]];
rng default;
Da = [D [1;1] [0;0]];
t = 0 : 0.01 : 10;
w=randn(1,length(t));
u = sin(t);
v = rand(1, length(t));
ua=[u; v; w];
sys = ss(Acl,Ba,C,Da);
figure(9);
subplot(1,2,1);
lsim(scaled_os_con_sys, u, t);
title('no noise');
subplot(1,2,2);
lsim(sys, ua, t);
title('noise');
```

2.8 Kalman filter

The whole explaination of how to implement the kalman filter with matlab and what does it do is given here.

```
Q=1;
R=1;
[kest, L, P] = kalman(sys, Q, R)
kest = kest(1, :);

s = parallel(sys, kest, 1, 1, [], []);
SimModel = feedback(s, 1, 4, 2, 1);
SimModel = SimModel([1 3],[1 2 3]);
[out, x] = lsim(SimModel, ua, t);

y=out(:, 1);
yf = out(:, 2);

figure(12);
plot(t, y, 'r', t, yf, 'b');
hold on
[yn, x] = lsim(sys, ua, t);
plot(t, yn(:, 1), '--g');
title('Kalman filter');
legend('true','kalma','noisy');
grid
```



In this kalman is blue, the real system is red and green is the noisy system therefore kalman is pretty close to the original system.

2.9 2(J)LGQ

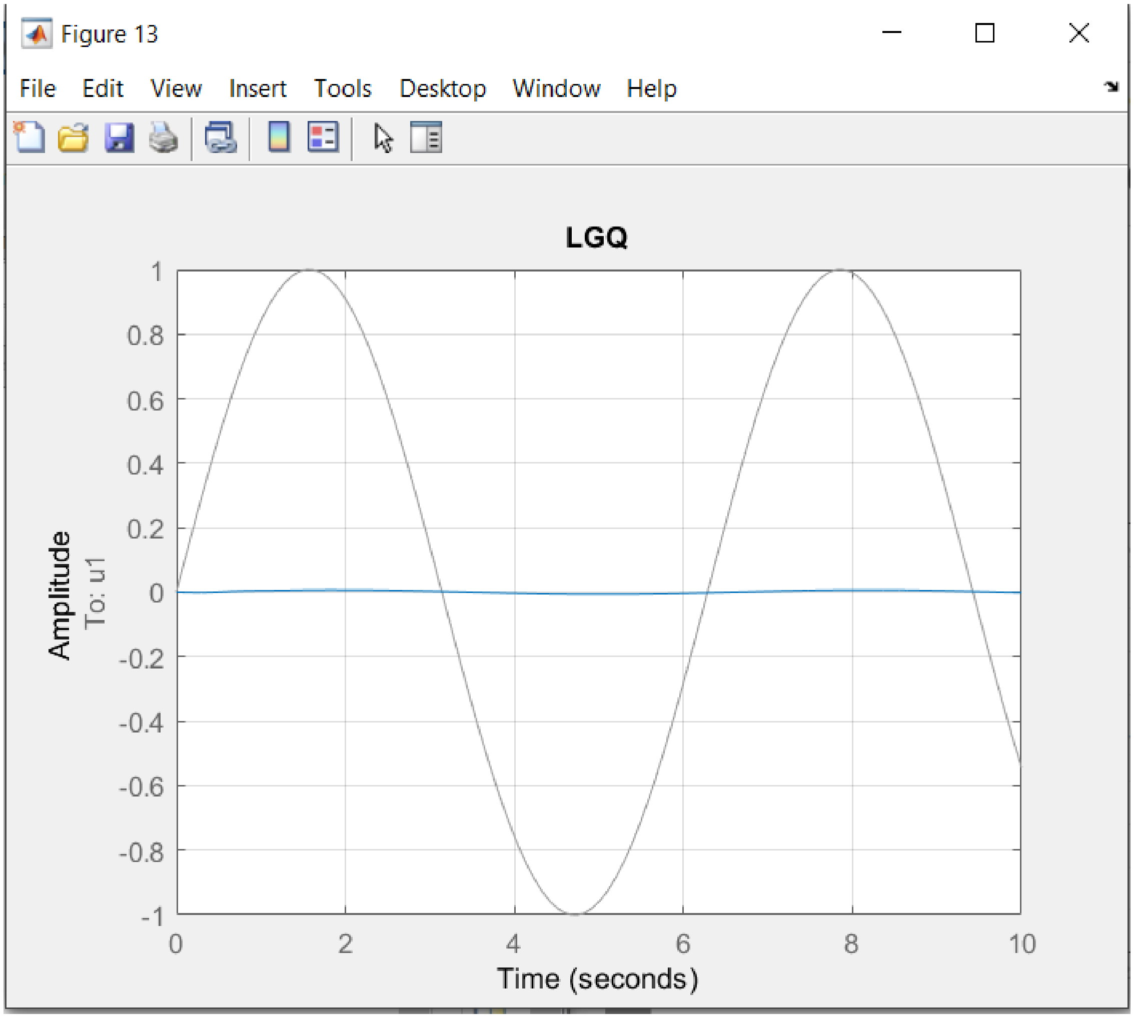


Figure 13
File Edit View Insert Tools Desktop Window Help
LGQ
Amplitude
To: u1
Time (seconds)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
2
4
6
8
10