Final Project

Applied Linear Systems

Parisha Joshi - parishamaheshj18@vt.edu

Submission Date: 12/7/2018

```
clear all;
close all;
clc;
load sys.mat
```

Problem 1

LQR function to make ful state feedack gain matrix:

```
Q_star=diag([52,32,41]);
R=diag([0.00001,0.00001,0.00001]);
Q= C' * Q_star*C;

[G,S,E]= lqr(sys,Q,R);
A_c=A-B*G;
sys_new= ss(A_c,B,C,D)
```

```
sys_new =

  A =
              x1         x2         x3         x4         x5         x6
   x1          0          1          0          0          0          0
   x2     -0.535    -0.8383     0.1146     0.1085    0.02218    0.03733
   x3          0          0          0          1          0          0
   x4     0.1087     0.1215    -0.4534    -0.8819     0.1803      0.204
   x5          0          0          0          0          0          1
   x6    0.01991    0.04713     0.2118       0.23    -0.3567    -0.7714

  B =
                u1          u2          u3
   x1            0           0           0
   x2    0.0001774           0           0
   x3            0           0           0
   x4            0   0.0001877           0
   x5            0           0           0
   x6            0           0   0.0001993

  C =
       x1  x2  x3  x4  x5  x6
   y1   1   0   0   0   0   0
   y2   1   0  -1   0   0   0
   y3   0   0   1   0  -1   0

  D =
       u1  u2  u3
   y1   0   0   0
   y2   0   0   0
   y3   0   0   0
```

```
Continuous-time state-space model.
```

Problem 2

From the midterm project:

Initial conditions are: [1 0 0.5 0 -1 0]

```
init_Condition = [1,0,0.5,0,-1,0];
[y,t,x]= initial(sys_new,init_Condition);
```

Calculating the settling time :

```
CloseLoop_step = stepinfo(y,t,0,'SettlingTimeThresold', 0.06); %Closed loop step info
[SettleTime_A,SettleTime_B,SettleTime_C] = CloseLoop_step.SettlingTime
```

```
SettleTime_A = 7.3393
SettleTime_B = 6.7815
SettleTime_C = 3.8118
```

Plotting The results gained by initial function:

```
Tsf_In = -G * transpose(x)
```

```
Tsf_In = 3×139
10^3 ×
   -1.8178   -1.5984   -1.3885   -1.1885   -0.9986   -0.8189   -0.6495   -0.4905 ···
   -1.6892   -1.5379   -1.3933   -1.2555   -1.1246   -1.0006   -0.8834   -0.7731
    2.1206    1.9074    1.7045    1.5121    1.3301    1.1586    0.9975    0.8466
```

```
max_torq_A = max(abs(Tsf_In(:,1)))
```

```
max_torq_A = 2.1206e+03
```

```
max_torq_B =max(abs(Tsf_In(:,2)))
```

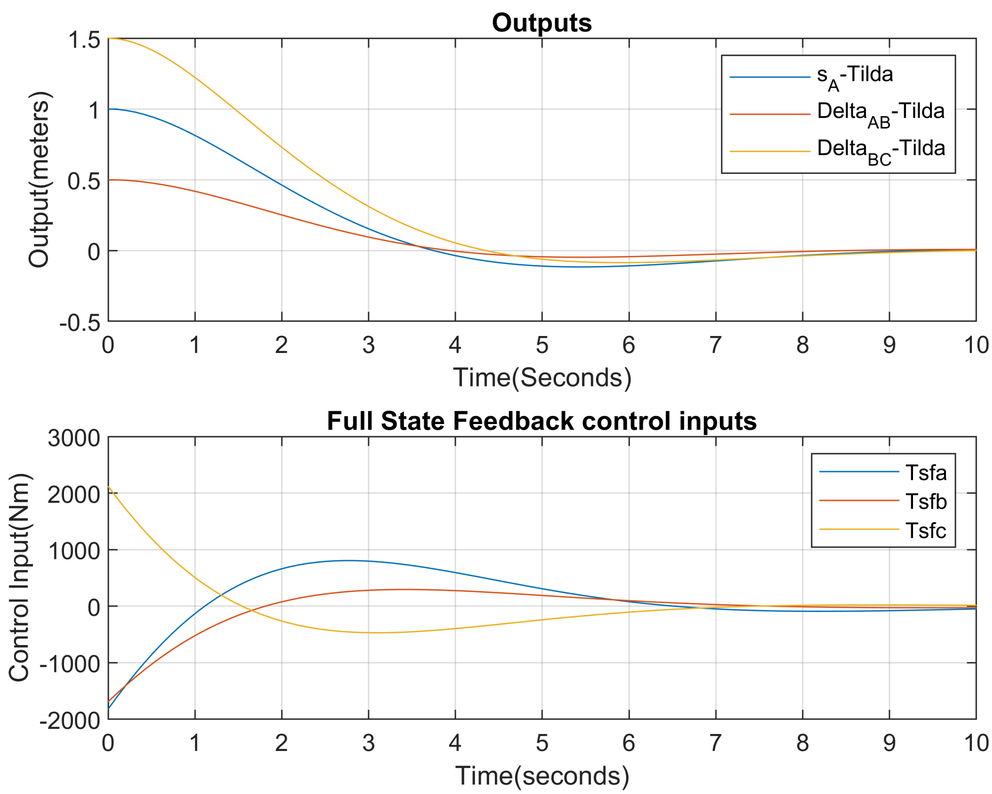
```
max_torq_B = 1.9074e+03
```

```
max_torq_C = max(abs(Tsf_In(:,3)))
```

```
max_torq_C = 1.7045e+03
```

```
figure()
hold on
subplot(2,1,1)
plot(t,y)
xlim([0 10])
xlabel('Time(Seconds)')
ylabel('Output(meters)')
title('Outputs')
legend('s_A-Tilda','Delta_A_B-Tilda','Delta_B_C-Tilda')
```

```
grid on

subplot(2,1,2)
plot(t,Tsf_In)
xlabel('Time(seconds)')
ylabel('Control Input(Nm)')
legend('Tsfa','Tsfb','Tsfc')
title('Full State Feedback control inputs')
xlim([0 10])
grid on
```



Problem 3

Checking observability with different combinations of y1,y2 and y3:

1) Using y1

```
sys_1 = sys(1,:);
Obs_matrix_1 = obsv(sys_1);
rank_y1= rank(Obs_matrix_1)
```

```
rank_y1 = 2
```

System is **not completely observable**, because the rank is 2 and is not equal to number of states.

2) Using y2

```
sys_2 = sys(2,:);
Obs_matrix_2 = obsv(sys_2);
rank_y2= rank(Obs_matrix_2)
```

```
rank_y2 = 4
```

System is **not completely observable**, because the rank is 4 and is not equal to number of states.

3) Using y3

```
sys_3 = sys(3,:);
Obs_matrix_3 = obsv(sys_3);
rank_y3= rank(Obs_matrix_3)
```

```
rank_y3 = 6
```

System is **completely observable**, because the rank is 6 and is equal to number of states.

4) Using y1, y2

```
sys_12 = sys([1:2],:);
Obs_matrix_12 = obsv(sys_12);
rank_y1y2= rank(Obs_matrix_12)
```

```
rank_y1y2 = 4
```

System is **not completely observable**, because the rank is 4 and is not equal to number of states.

5) Using y2, y3

```
sys_23 = sys([2:3],:);
Obs_matrix_23 = obsv(sys_23);
rank_y2y3= rank(Obs_matrix_23)
```

```
rank_y2y3 = 6
```

System is **completely observable**, because the rank is 6 and equal to number of states.

6) Using y1, y3

```
sys_13 = sys([1,3],:);
Obs_matrix_13 = obsv(sys_13);
rank_y1y3= rank(Obs_matrix_13)
```

```
rank_y1y3 = 6
```

System is **completely observable**, because the rank is 6 and equal to number of states.

7) Using y1,y2, y3

```
sys_123 = sys([1:3],:);
Obs_matrix_123 = obsv(sys_123);
```

```
rank_y1y2y3= rank(Obs_matrix_123)
```

```
rank_y1y2y3 = 6
```

System is **completely observable**, because the rank is 6 and equal to number of states.

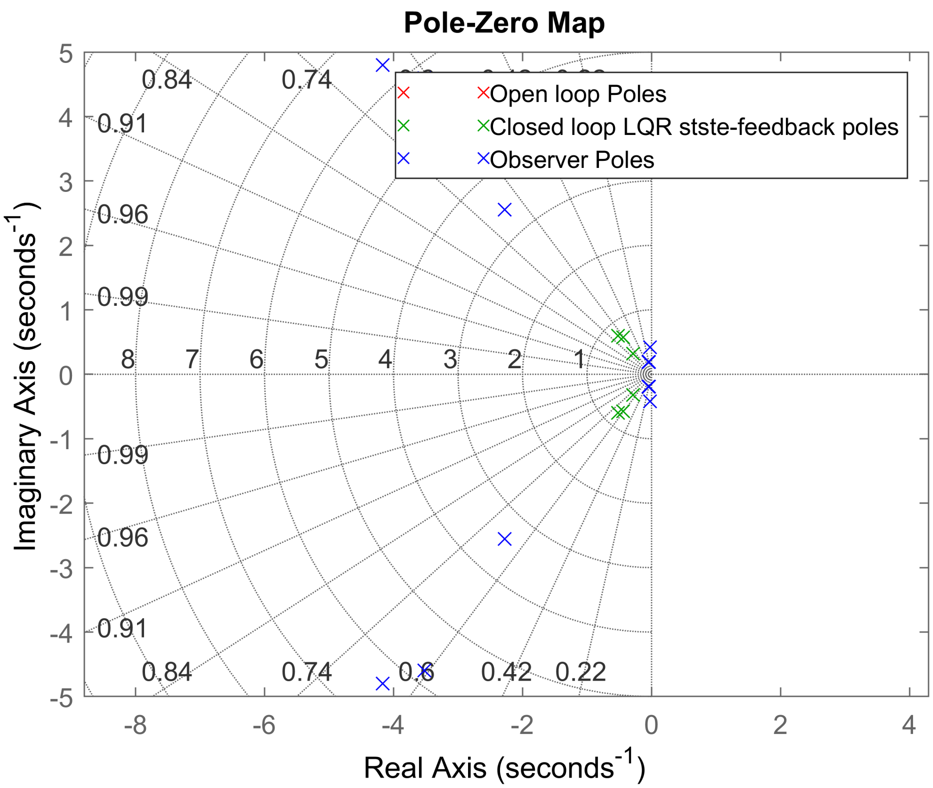
Justification:

The observability of this system depends more on output y3. It is important for y3 which is Delta_BC-Tilda,
To be present in the calculation for system to be completely observable. The subsets of outputs on which the system is completely observble is= **{ [y3], [y1, y3], [y2, y3], [y1, y2, y3] }**

Problem 4

Luenberger Observer design:

```
%Assume that the A-hat,B-hat,C-hat and D-hat is equal to A,B,C and D.
A_ht= A;
B_ht= B;
C_ht= C;
D_ht= D;
%We want to shift the poles more to left where E is the eigenvalues after G is applied.
alfa= 8;
E_new=alfa*E;
%luenberger observer K
K=place(A',C',E_new).';
aug_A = [A, zeros(size(A));K*C, A-K*C];
aug_B = [B;B];
aug_C = [C, zeros(size(C));zeros(size(C)), C];
aug_D = [ D; D];

aug_sys = ss(aug_A, aug_B, aug_C, aug_D);

figure()
pzmap(sys,'rx',sys_new,'gx',aug_sys,'bx')
legend('Open loop Poles', 'Closed loop LQR stste-feedback poles', 'Observer Poles' )
axis equal
sgrid
```

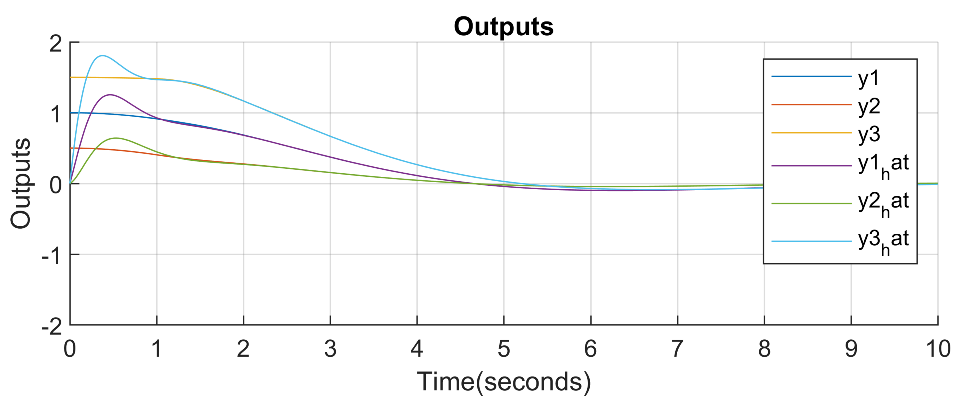


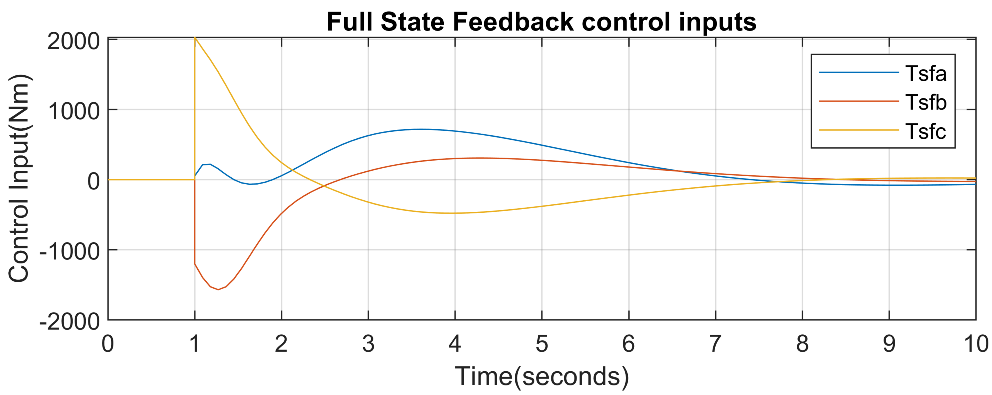
Problem 5

Open Loop system simulation for 1 second and then closing the loop with FSF.

```
init_Cond_new = [1 0 0.5 0 -1 0 0 0 0 0 0 0];
[y_ol,t_ol,x_ol]= initial(aug_sys,init_Cond_new,1);

[m,n]=size(x_ol);
init_cond_new_1=x_ol(m,:);

%introducing G after 1 second the matrix will be:

aug_A_cl = [A, -B*G; K*C, A-K*C-B*G];
aug_C_cl = [C zeros(size(C)); zeros(size(C)) C];
aug_sys_cl = ss(aug_A_cl, [], aug_C_cl, []);
[y_aug_cl, t_aug_cl, x_aug_cl] = initial(aug_sys_cl, init_cond_new_1,9);

%combining these two results:

y_final=[y_ol;y_aug_cl];
t_final=[t_ol;t_aug_cl+1];
x_final=[x_ol,x_aug_cl];

Tsf_In = zeros(3,max(size(x_aug_cl)));
for i = 1:size(x_aug_cl(:,1))
```

```
  Tsf_In(:,i) = -G * transpose(x_aug_cl(i,7:12));  %input u = -G*X
end
Tsf_op = zeros(3,101);
Tsf_total = [Tsf_op, Tsf_In];

figure()
subplot(2,1,1)
hold on
plot(t_final,y_final);
xlim([0 10]);
ylim([-2 2])
xlabel('Time(seconds)')
ylabel('Outputs')
title('Outputs')
legend('y1','y2','y3', 'y1_hat', 'y2_hat', 'y3_hat')
grid on

subplot(2,1,2)
Tsf_In;
plot(t_final,Tsf_total);
xlim([0 10]);
xlabel('Time(seconds)')
ylabel('Control Input(Nm)')
legend('Tsfa','Tsfb','Tsfc')
title('Full State Feedback control inputs')
grid on;
```





```
max_torq_A = max(abs(Tsf_In(:,1)))
```

```
max_torq_A = 2.0300e+03
```

```
max_torq_B =max(abs(Tsf_In(:,2)))
```

```
max_torq_B = 1.8672e+03
```

```
max_torq_C = max(abs(Tsf_In(:,3)))
```

```
max_torq_C = 1.7101e+03
```

Problem 6

Closed loop control system design with LQI function for reference track:

```
%designing Q and R matrices
Q_i_new = diag([50 50 50]);
Q_i = [Q, zeros(6,3); zeros(3,6), Q_i_new];
R_i = 0.00001 * eye(3);
K_new = lqi(sys, Q_i,R_i);

%Separating GI and G0 from K_new
GI = K_new(:,7:9);
G0 = K_new(:,1:6);
```

Problem 7

Defining LTI Objects for the individual blocks and connecting the sys

```
REF_sys = tf(eye(3));  % Static gain identity matrix
SUM_sys_1 = tf(eye(3)); % Static gain identity matrix
SUM_sys_2 = tf(eye(3)); % Static gain identity matrix
G0_sys = tf(-G0);      % Static gain G0 matrix (negated)
GI_sys = tf(-GI);      % Static gain GI matrix (negated)
a=tf([0 1],[1 0]);

INTE_sys = [a 0 0;0 a 0;0 0 a];

A_ob= A_ht - K*C_ht;
B_ob = [B_ht K];
C_ob = [eye(6);C_ht];
OB_sys = ss(A_ob, B_ob, C_ob, []);

%appending all the LTI models

bulk_system = append(sys, OB_sys, REF_sys, SUM_sys_1, INTE_sys, G0_sys, GI_sys, SUM_sys_2);
size(bulk_system)
```

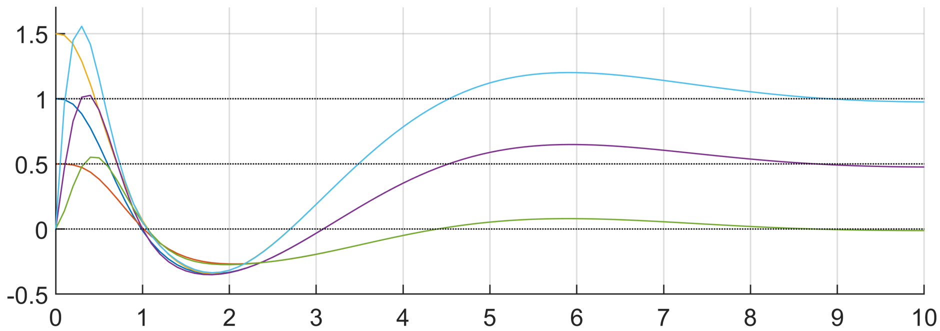
```
State-space model with 30 outputs, 30 inputs, and 15 states.
```

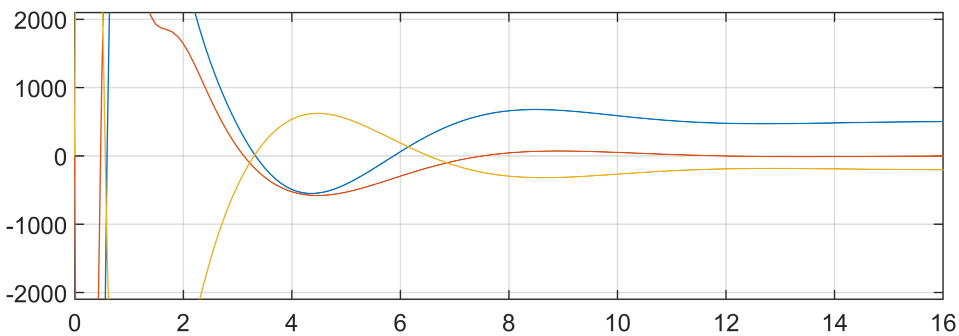
```
%Connecting the inputs and outputs of the bulk sys using the connection function
%opt = connectOptions('Simplify',false);
connection = [1 28 0; 2 29 0; 3 30 0;  4 28 0; 5 29 0; 6 30 0; 7 1 0;8 2 0; 9 3 0;10 0 0; 11 0
fin_system = connect(bulk_system, connection, [10,11,12],[1,2,3,10,11,12,28,29,30]);
size(fin_system)
```

```
State-space model with 9 outputs, 3 inputs, and 15 states.
```

Problem 8

```
new_init_cond = [1 0 0.5 0 -1 0 0 0 0 0 0 0 0 0 0]';
Tfinal = 20; % final sim time (seconds)
Ts = 0.1; % data sample period (seconds)
T = Ts * (0:round(Tfinal/Ts)); % time vector (seconds)

r_c = [0.5 0 1]; %Defining reference input
r = zeros(size(T,2),3);
for i = 1:size(T,2)
    r(i,:) = r_c;
end
figure()
subplot(2,1,1)
[y,t] = lsim(fin_system,r,T, new_init_cond);
hold on
plot(t, y(:,1:6))
plot(t, r,'k:')
ylim([-0.5 1.7])
xlim([0 10])
grid on
subplot(2,1,2)
plot(t, y(:,7:9))
ylim([-2100 2100])
xlim([0 16])
grid on
```


1.5
1
0.5
0
-0.5
0
1
2
3
4
5
6
7
8
9
10


2000
1000
0
-1000
-2000
0
2
4
6
8
10
12
14
16

```
%fin
```