
HOMEWORK 3

April 26, 2018

Sandesh Thapa
MAE 5010
Atmospheric Flight Control

Spring 2018
Dr. Imraan Faruque
School of Mechanical and Aerospace Engineering
Oklahoma State University

1. Attached
2. LQR-Prob2

Dynamics:

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 1 \end{bmatrix}$$

Simulation parameters:

$$x_0 = \begin{bmatrix} 1.0 & 0.5 \end{bmatrix}^T, \quad R = \begin{cases} 0.1 \\ 0.5 \\ 1.0 \\ 5.0 \end{cases}, \quad Q = I$$

Initial stabilizing gain: $K_0 = 5.0$

Conditions check [1]

- (a) Finding a stabilize gain K such that $A_c = A - BKC$ is stable which implies that the system is output stabilizable. We were able to find a stabilize gain K (for e.g. , $k = 5.0$ works) such that $A_c = A - BKC$ is stable which implies that the system is output stabilizable.
- (b) C matrix has full row rank in this case. (rank of $C = 1$).
- (c) $R > 0$ and $Q \geq 0$ is true.
- (d) (A, \sqrt{Q}) is detectable.

Rank of

$$\text{rank} \begin{pmatrix} \lambda I - A \\ -\sqrt{Q} \end{pmatrix} = n \quad \forall \quad \lambda \in \mathbb{C}^+ \quad (1)$$

$$\text{rank of} \begin{pmatrix} \sqrt{Q} \\ \sqrt{Q}A \end{pmatrix} = 2$$

So it is observable which implies the dynamics is detectable.

Simulink Diagram

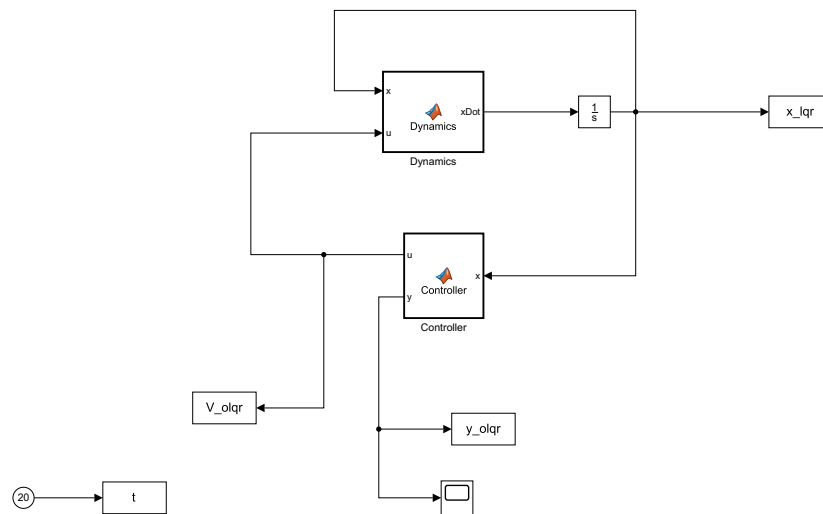


Figure 1: Closed Loop Model with the Dynamics and Controller for simulation

Results

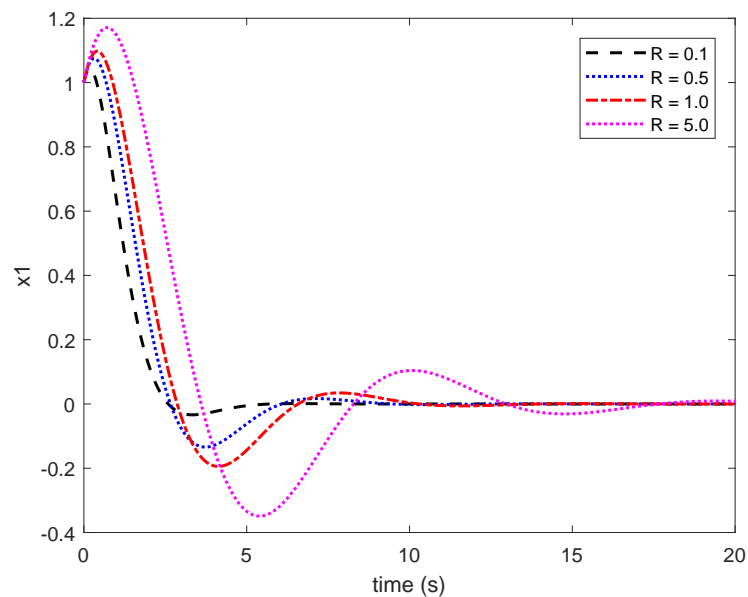


Figure 2: Trajectories of state x_1 during the simulation. For all values of R , the controller regulates the state from 1.0 to zero.

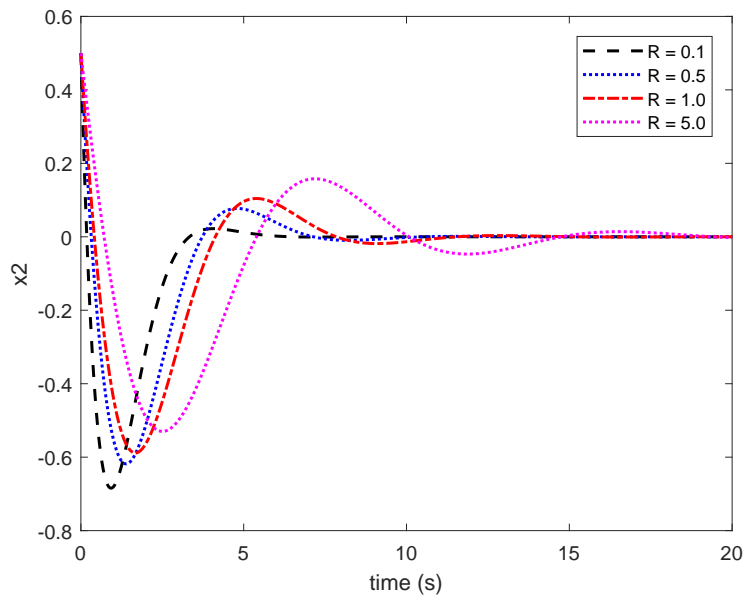


Figure 3: Trajectories of state x_2 during the simulation. For all values of R , the controller regulates the state from 0.5 to zero.

It can be observed from Fig 2, and 3 that as R is increased the settling time also increases for both states. From state x_1 , overshoot(first peak) increases significantly as we increase R . But for state x_2 , overshoot (first peak) decreases for as R is decreased.

3. Problem 4

Simulation parameters:

$$x_0 = \begin{bmatrix} 1.0 & 1.0 & 1.0 & 1.0 \end{bmatrix}^T, \quad \rho = \begin{cases} 0.1 \\ 0.5 \\ 1.0 \end{cases},$$

Initial stabilizing gain:

$$K_0 = \begin{bmatrix} 0.5 & -1.0 \\ 0.5 & -1.0 \end{bmatrix}$$

Conditions check [1]

- (a) Finding a stabilize gain K such that $A_c = A - BKC$ is stable which implies that the system is output stabilizable.

We were able to find a stabilize gain K (for e.g. , $k = K_0$ works) such that $A_c = A - BKC$ is stable which implies that the system is output stabilizable.

- (b) C matrix has full row rank in this case. (rank of $C = 2$).

(c) $R > 0$ and $Q \geq 0$ is true.

(d) (A, \sqrt{Q}) is detectable.

So it is observable which implies the dynamics is detectable.

Results

Set of optimal gains

For $\rho = 0.1$,

$$K_1 = \begin{bmatrix} -1.1914 & -0.7028 \\ 0.6179 & -1.0284 \end{bmatrix}$$

For $\rho = 0.5$,

$$K_1 = \begin{bmatrix} -1.4810 & -0.6327 \\ 0.6555 & -1.0136 \end{bmatrix}$$

For $\rho = 1.0$,

$$K_3 = \begin{bmatrix} -1.4414 & -0.6528 \\ 0.6468 & -1.0087 \end{bmatrix}$$

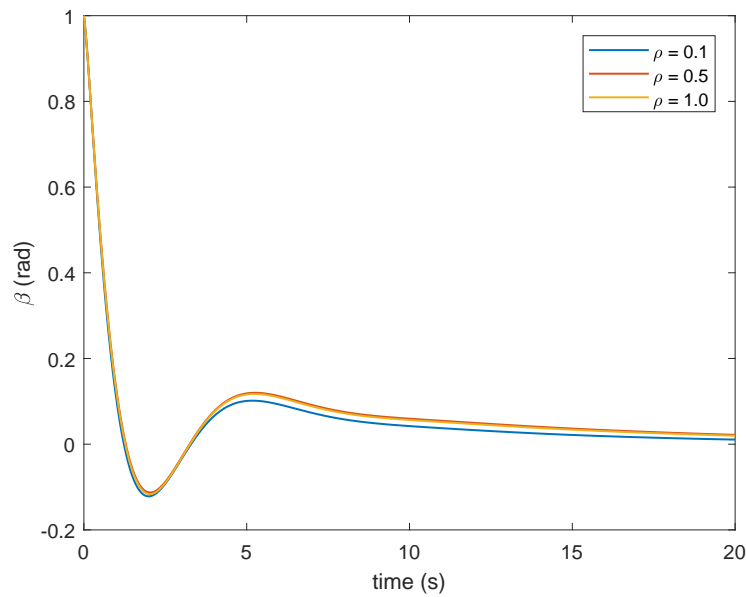


Figure 4: Trajectories of state β during the simulation. For all values of ρ , the controller regulates the state from 1.0 to zero.

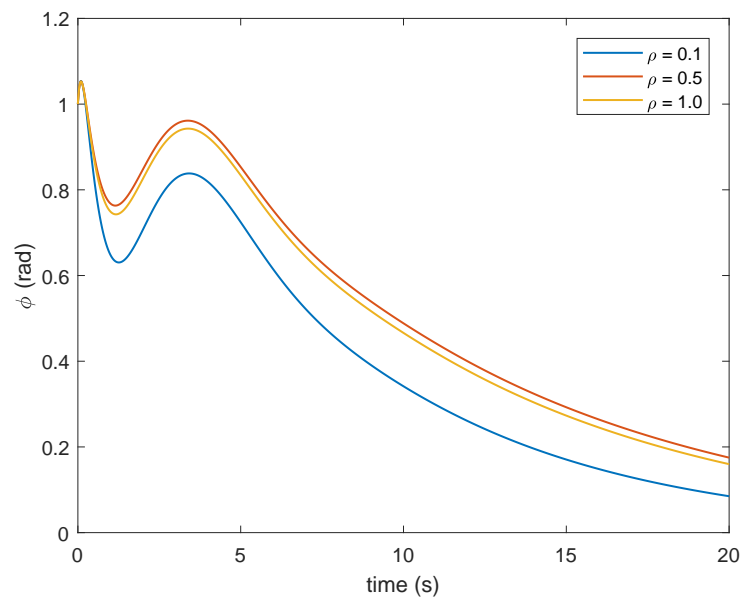


Figure 5: Trajectories of state ϕ during the simulation. For all values of ρ , the controller regulates the state from 1.0 to zero.

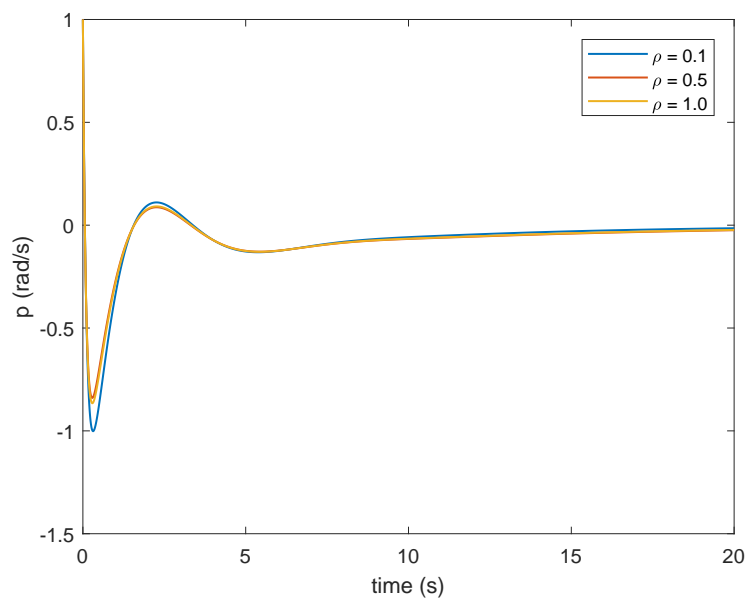


Figure 6: Trajectories of state p during the simulation. For all values of ρ , the controller regulates the state from 1.0 to zero.

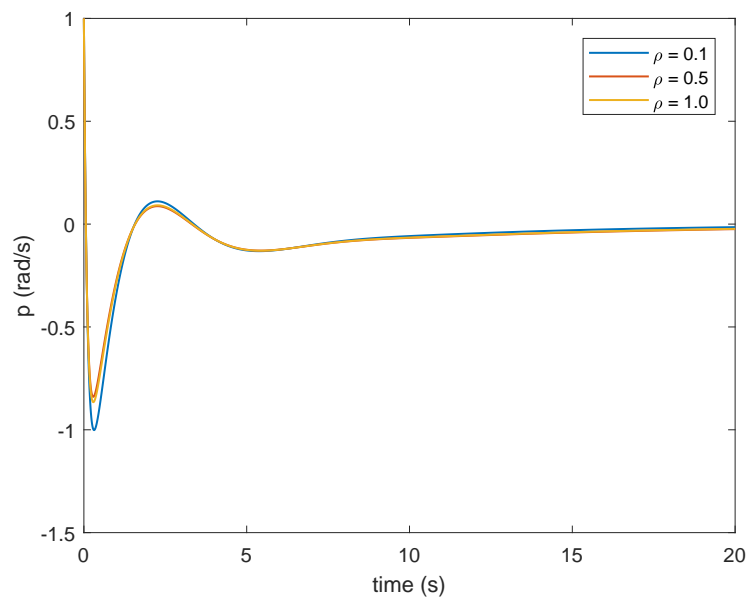


Figure 7: Trajectories of state r during the simulation. For all values of ρ , the controller regulates the state from 1.0 to zero.

There is not significant change in control authority, as we change ρ , however, as seen in the above plot, we can observe that as ρ is increased it decreases the overshoot, the settling times remains almost the same for all three cases.

4. Problem 4

(a) Performance Specs

- Attenuation of input disturbance signals d_I at the plant output y . We want $\sigma(S_0 G)$ to be small.

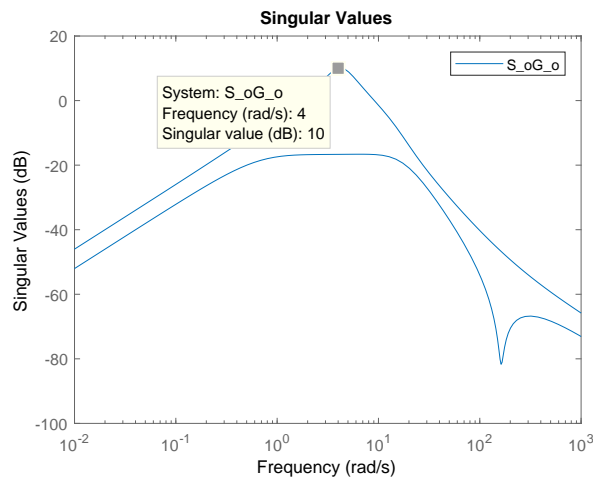


Figure 8: Singular values of $S_o G$. We want $\omega < 10 \text{ rad/s}$

- Avoidance of large control signals u due to reference demands r . We want $\sigma(KS_o)$ to be small.

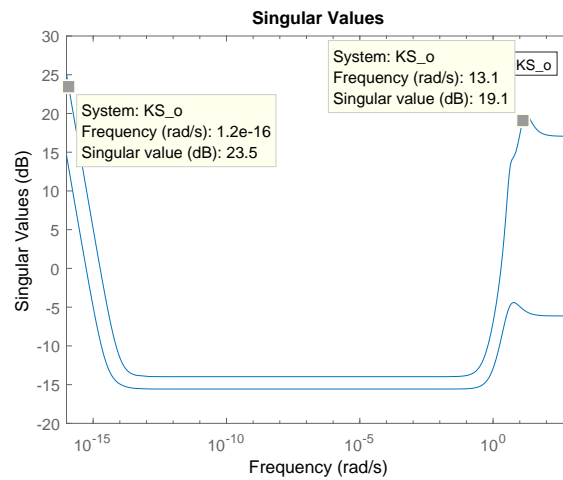


Figure 9: Singular values of $K S_o$

- Attenuation of measurement noise signals n at the plant input y . We want $\sigma(T_o)$ to be small.

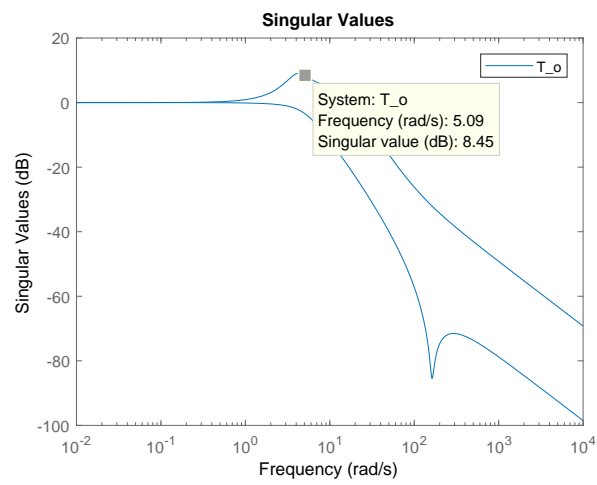


Figure 10: Singular values of T_o

(b) Plot of two weighting functions

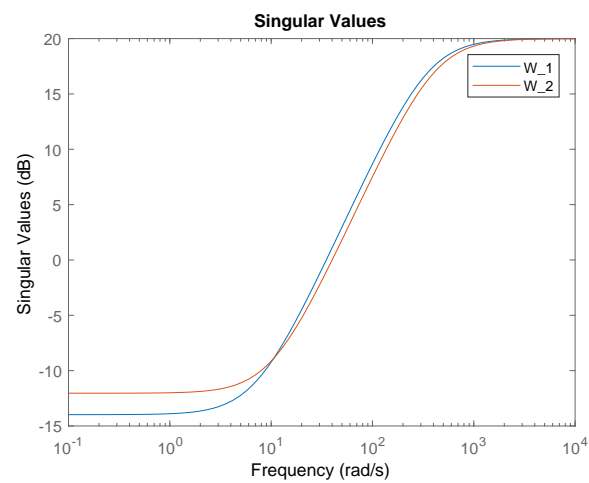


Figure 11: Plot of weighting functions.

(c) Robustness

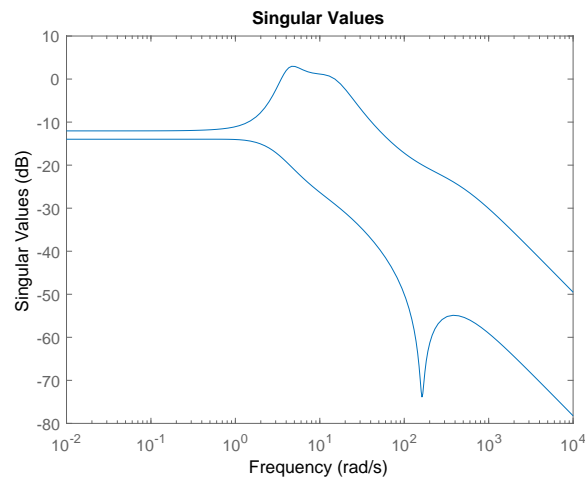


Figure 12: Plot of weighting functions.

$\| -W_1 T_I \|_\infty = 1.4104 > 1$. The controller is not robust.

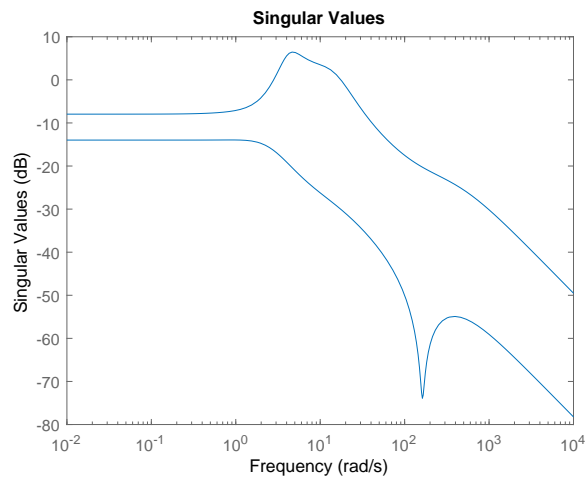


Figure 13: Plot of weighting functions when second channel is increased to 40 %

When uncertainty in second channel is increased to 40 %, We get $\| -W_1 T_I \|_\infty = 2.1031 > 1$.
The controller is not robust.

(d) Inverse additive uncertainty

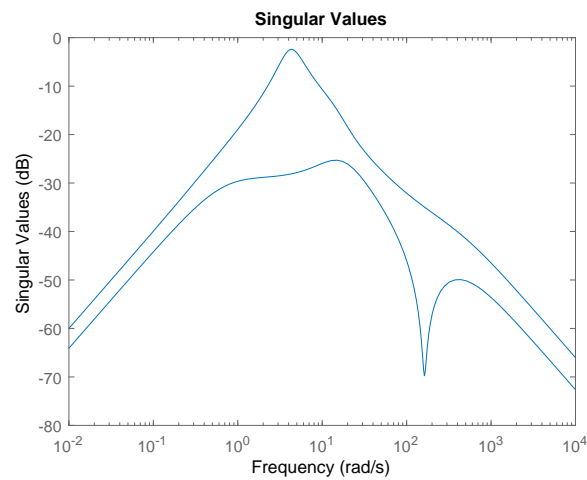


Figure 14: Plot of weighting functions when second channel is increased to 40 %

When uncertainty in second channel is increased to 40 %, We get $\|M\|_{\infty} = 0.7579 < 1$. The controller is robust.

5. Problem 5

(a) 5 a

- For K1

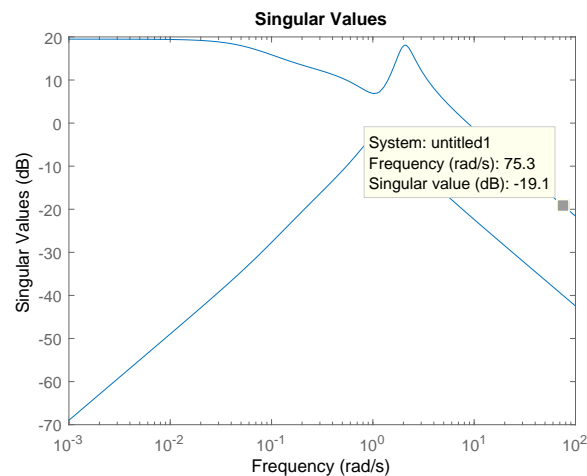


Figure 15: Singular value plot

- For K2

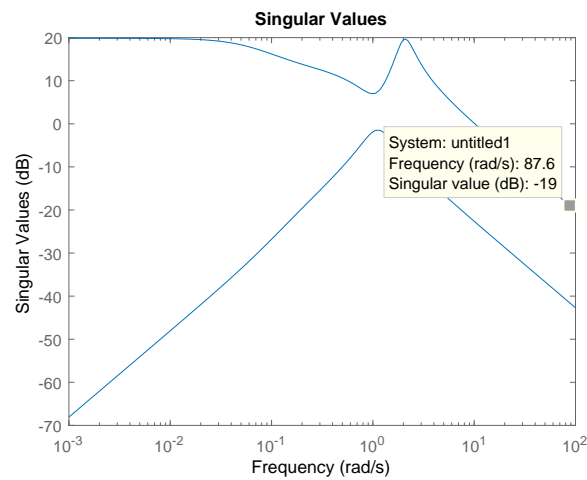


Figure 16: Singular value plot

- For K3

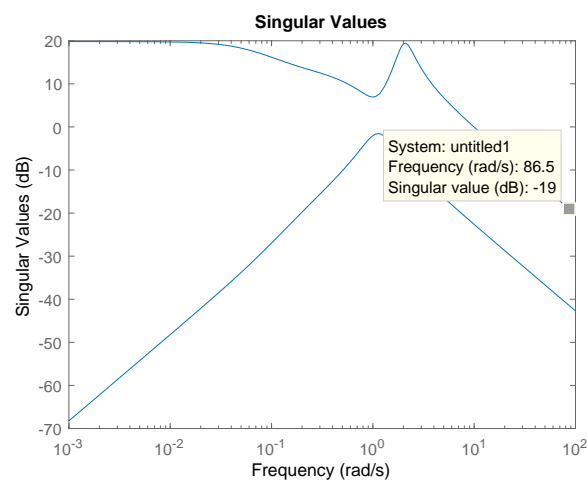


Figure 17: Singular value plot

The minimum frequency for disturbance rejection is 75.3 rad/s.

- (b) Output multiplicative uncertainty

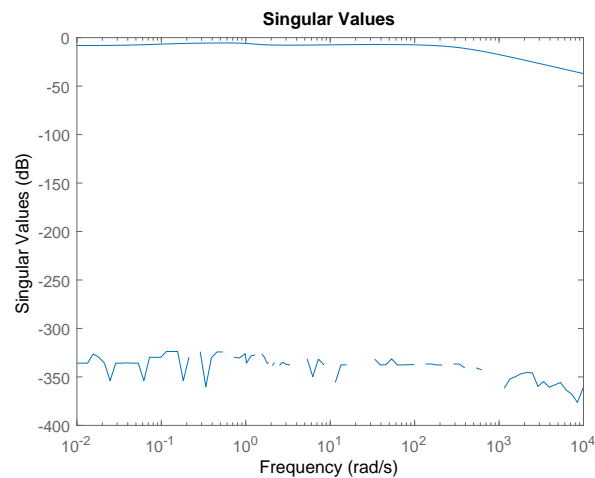


Figure 18: Singular value plot

H-infinity norm = 0.5289, the system is robust.

For second case, H-infinity norm = 0.9852, the system is robust. We can increase the percentage upto 56 % and still have robust stability.

A Matlab Code

1. Problem 2

```

1 %% MAE 5010– Atmospheric Flight Control HW3
2 % LQ Regulator Prob 2
3 % Sandesh Thapa
4
5 clear all;
6 close all
7 clc;
8 %%
9 A = [0 1;0 0];
10 B = [0 1]';
11 C = [1 1];
12 Q = eye(2);
13 x0 = [1 0.5]';
14 X = x0*x0';
15 R = 5.0;
16
17 [K, J, P] = OutputLQRProb2(A, B, C, Q, R, x0);
18 %%
19 sim('Sim_OutputLQR')
20 %%

```

```

21 figure
22 plot(t,x_lqrA(:,1),'--k',t,x_lqrB(:,1),' :b',t,x_lqrC(:,1),'-.r',t,x_lqrD(:,1),' :
    m','Linewidth',1.5);
23 legend('R = 0.1','R = 0.5','R = 1.0','R = 5.0')
24 xlabel('time (s)')
25 ylabel('x1')
26 hold on
27
28 figure
29 plot(t,x_lqrA(:,2),'--k',t,x_lqrB(:,2),' :b',t,x_lqrC(:,2),'-.r',t,x_lqrD(:,2),' :
    m','Linewidth',1.5);
30 legend('R = 0.1','R = 0.5','R = 1.0','R = 5.0')
31 xlabel('time (s)')
32 ylabel('x2')
33 hold on
34
35 %%
36 figure
37 plot(t,x_lqrA(:,1));
38 legend('R = 0.1')
39 xlabel('time (s)')
40 ylabel('x1')
41 hold on
42
43 figure
44 plot(t,x_lqrA(:,2));
45 legend('R = 0.1')
46 xlabel('time (s)')
47 ylabel('x2')
48 hold on

```

2. Output LQR Function Problem 2

```

1 function [K_k,J_k,P_k] = OutputLQRProb2(A,B,C,Q,R,x0)
2 % function [K_k,J_k,P_k] = OutputLQRProb2(A,B,C,Q,R,x0) --> Output LQR
3 % solver based on Moerder and Calise (1985) for Prob 2
4 % x0 = [1 0]';
5 X = x0*x0';
6 K_k = 5;
7
8 % K_k = ones(4,2);
9
10 A_k = A - B*K_k*C;
11 eig(A_k);

```

```

12 Jt = zeros(1000,1);
13 JtNew = zeros(1000,1);
14 Kt = zeros(1000,1);
15 alpha = 0.5;
16 for k = 1:1000
17     % K_k = K_0;
18     A_k = A - B*K_k*C;
19     P_k = lyap(A_k,Q + C'*K_k'*R*K_k*C);
20     S_k = lyap(A_k,X);
21     J_k = 1/2*trace(P_k*X);
22     Jt(k) = J_k;
23     DeltaK = inv(R)*B'*P_k*S_k*C'*inv(C*S_k*C') - K_k;
24
25     for i = 1:1000
26         K_kNew = K_k + alpha*DeltaK;
27         A_k = A - B*K_kNew*C;
28         P_kNew = lyap(A_k,Q + C'*K_kNew'*R*K_kNew*C);
29         S_k = lyap(A_k,X);
30         EigAk = eig(A_k);
31
32         J_kNew = 1/2*trace(P_kNew*X);
33         JtNew(i) = J_kNew;
34         Delta_J = J_k - J_kNew;
35
36         if max(real(EigAk)) < 0 && J_kNew < J_k
37             K_k = K_kNew;
38             Kt(k) = K_k;
39             J_k = J_kNew;
40             Jt(i) = J_k;
41             break
42         else
43             alpha = alpha/10;
44             K_kNew = K_k + alpha*DeltaK;
45             K_k = K_kNew;
46             Kt(k) = K_k;
47             A_k = A - B*K_k*C;
48             EigAk = eig(A_k);
49             P_kNew = lyap(A_k,Q + C'*K_kNew'*R*K_kNew*C);
50             S_k = lyap(A_k,X);
51             J_kNew = 1/2*trace(P_kNew*X);
52             J_k = J_kNew;
53             Jt(i) = J_k;
54             DeltaK = inv(R)*B'*P_k*S_k*C'*inv(C*S_k*C') - K_k;

```

```

55         end
56     end
57
58     if Delta_J < 0.0000001
59         K_k = K_kNew;
60         Kt(k) = K_k;
61         J_k = J_kNew;
62         break
63     end
64
65 end

```

3. Problem 3 & 5

```

1  clear all;
2  close all
3  clc;
4  %%
5  A = [-0.131150  0.14858  0.32434  -0.93964;
6        0.0      0.0      1.0      0.33976;
7        -10.614   0.0     -1.1793   1.023;
8        0.99655   0.0     -0.001874 -0.25855];
9
10 B = [0.00012  0.00032897;
11       0.0      0.0;
12       -0.1031578 0.020987;
13       -0.0021330 -0.010715];
14
15 C = [0 0 57.29578 0;
16       0 0 0 57.29578];
17
18 D = zeros(2,2);
19
20 qdr = 50;
21 qr = 100;
22 % v = [qdr,qr,qr,qdr,0,0,0];
23 Q = diag([qdr,qr,qr,qdr]);
24 x0 = [1.0 1.0 1.0 1.0]';
25 X = x0*x0';
26 rho = 1.0;
27 n = 3;
28 % for j = 1:n
29 R = rho*eye(2);
30

```



```

31 [K,J,P] = OutputLQRProb3(A,B,C,Q,R,x0);
32
33 %%
34 % K = K_k;
35 sim('Sim_OutputLQR')
36 %%
37 % end
38 % for j = 1:n
39 figure(1)
40 plot(t,x_lqrA(:,1),t,x_lqrB(:,1),t,x_lqrC(:,1),'Linewidth',1.0);
41 legend('\rho = 0.1','\rho = 0.5','\rho = 1.0')
42 xlabel('time (s)')
43 ylabel('\beta (rad)')
44 hold on
45
46 figure(2)
47 plot(t,x_lqrA(:,2),t,x_lqrB(:,2),t,x_lqrC(:,2),'Linewidth',1.0);
48 legend('\rho = 0.1','\rho = 0.5','\rho = 1.0')
49 xlabel('time (s)')
50 ylabel('\phi (rad)')
51 hold on
52
53 figure(3)
54 plot(t,x_lqrA(:,3),t,x_lqrB(:,3),t,x_lqrC(:,3),'Linewidth',1.0);
55 legend('\rho = 0.1','\rho = 0.5','\rho = 1.0')
56 xlabel('time (s)')
57 ylabel('p (rad/s)')
58 hold on
59
60 figure(4)
61 plot(t,x_lqrA(:,4),t,x_lqrB(:,4),t,x_lqrC(:,4),'Linewidth',1.0);
62 legend('\rho = 0.1','\rho = 0.5','\rho = 1.0')
63 xlabel('time (s)')
64 ylabel('r (rad/s)')
65 hold on

```

4. Output LQR Function Problem 3

```

1 function [K_k,J_k,P_k] = OutputLQRProb3(A,B,C,Q,R,x0)
2 K_k = [-0.5 -1.0;0.5 -1.0];
3 X = x0*x0';
4 A_k = A - B*K_k*C;
5 eig(A_k)
6

```

```

7  Jt = zeros(1000,1);
8  JtNew = zeros(1000,1);
9  Kt = zeros(1000,1);
10 alpha = 0.5;
11 for k = 1:1000
12     %     K_k = K_0;
13     A_k = A - B*K_k*C;
14     P_k = lyap(A_k,Q + C'*K_k'*R*K_k*C);
15     S_k = lyap(A_k,X);
16     J_k = 1/2*trace(P_k*X);
17     Jt(k) = J_k;
18     DeltaK = inv(R)*B'*P_k*S_k*C'*inv(C*S_k*C') - K_k;
19
20     for i = 1:1000
21         K_kNew = K_k + alpha*DeltaK;
22         A_k = A - B*K_kNew*C;
23         P_kNew = lyap(A_k,Q + C'*K_kNew'*R*K_kNew*C);
24         S_k = lyap(A_k,X);
25         EigAk = eig(A_k);
26
27         J_kNew = 1/2*trace(P_kNew*X);
28         JtNew(i) = J_kNew;
29         %         if i > 1
30         %             Delta_J(i) = Jt(i) - Jt(i+1);
31         Delta_J = J_k - J_kNew;
32
33         if max(real(EigAk)) < 0 && J_kNew < J_k
34             K_k = K_kNew;
35             %             Kt(k) = K_k;
36             J_k = J_kNew;
37             Jt(i) = J_k;
38             break
39         else
40             alpha = alpha/10;
41             K_kNew = K_k + alpha*DeltaK;
42             K_k = K_kNew;
43             %             Kt(k) = K_k;
44             A_k = A - B*K_k*C;
45             EigAk = eig(A_k);
46             P_kNew = lyap(A_k,Q + C'*K_kNew'*R*K_kNew*C);
47             S_k = lyap(A_k,X);
48             J_kNew = 1/2*trace(P_kNew*X);
49             J_k = J_kNew;

```

```

50         Jt(i) = J_k;
51         DeltaK = inv(R)*B'*P_k*S_k*C'*inv(C*S_k*C') - K_k;
52     end
53 end
54
55     if Delta_J < 0.0000001
56
57         K_k = K_kNew;
58         %         Kt(k) = K_k;
59         J_k = J_kNew;
60         break
61     end
62
63 end

```

5. Problem 4

```

1 %% Problem 4
2 %% (a)
3
4 clear all;
5 close all;
6 clc;
7
8 sys11 = tf([0 0 6],[0.09 1 1]);
9 sys12 = tf([0 -0.05],[0.1 1]);
10 sys21 = tf([0 0.07],[0.3 1]);
11 sys22 = tf([0 0 5],[0.108 1.74 -1]);
12
13 G0 = [sys11, sys12;
14       sys21, sys22];
15
16 K11 = tf([2 2],[1 0]);
17 K12 = tf([-1 0],[3 1]);
18 K21 = tf([-5 -5],[0.8 1]);
19 K22 = tf([4*0.7 4],[1 0]);
20
21 K = [K11,K12;
22      K21 K22];
23
24 loops = loopsens(G0,K);
25
26 % 4.1.i input di to y
27 figure

```

```
28 sigma(loops.So*G0)
29 legend('S_oG_o')
30 hold on
31
32 % u --> r
33 figure
34 sigma(K*loops.So)
35 legend('KS_o')
36 hold on
37
38 % n --> y small max(svd(T_o))
39 figure
40 sigma(loops.To)
41 legend('T_o')
42 hold on
43 %% 4(b)
44 W1 = makeweight(0.20,35,10);
45 W2 = makeweight(0.25,40,10);
46
47 figure
48 sigma(W1)
49 hold on
50 sigma(W2)
51 legend('W_1', 'W_2')
52 hold on
53
54 %% 4(c)
55 Wl = [W1, 0; 0,W2];
56 M = -Wl*loops.Ti;
57
58 figure
59 sigma(M)
60
61 HinfNorm1 = hinfnorm(M)
62 if HinfNorm1 > 1.0
63     fprintf('Controller is not robust')
64 else
65     fprintf('System is robust')
66 end
67
68
69 %% 4(c) Part b
70 W2New = makeweight(0.40,40,10);
```

```

71 WId = [W1, 0; 0,W2New];
72 M = -WId*loops.Ti;
73
74 figure
75 sigma(M)
76 HinfNorm2 = hinfnorm(M)
77
78 if HinfNorm2 > 1.0
79     fprintf('Controller is not robust')
80 else
81     fprintf('System is robustst')
82 end
83
84
85 %% 4(d)
86 M_d = loops.PSi*WI;
87
88 figure
89 sigma(M_d)
90 HinfNormM_d = hinfnorm(M_d)
91
92 if HinfNormM_d > 1.0
93     fprintf('Controller is not robust')
94 else
95     fprintf('System is robustst')
96 end

```

6. Problem 5

```

1  %% Problem 5(a)
2
3  A = [-0.131150  0.14858  0.32434  -0.93964;
4        0.0      0.0      1.0      0.33976;
5        -10.614   0.0     -1.1793   1.023;
6        0.99655   0.0     -0.001874 -0.25855];
7
8  B = [0.00012  0.00032897;
9        0.0      0.0;
10       -0.1031578 0.020987;
11       -0.0021330 -0.010715];
12
13 C = [0 0 57.29578 0;
14       0 0 0 57.29578];
15

```

```
16 D = zeros(2,2);
17 K1 = [ -1.1914    -0.7028;
18         0.6179    -1.0284];
19
20 K2 = [ -1.4810    -0.6327;
21         0.6555    -1.0136];
22
23 K3 = [ -1.4414    -0.6528;
24         0.6468    -1.0087];
25
26 G = tf(ss(A,B,C,D));
27 % G0 = ss(A,B,C,D,2);
28 % Lp = loopsens(G0,K1)
29 % figure
30 % sigma(Lp.To)
31 % hold on
32
33 figure
34 sigma(G*K1)
35 hold on
36
37 figure
38 sigma(G*K2);
39 hold on
40
41 figure
42 sigma(G*K3);
43 hold on
44
45 %% 5(b)
46 Lp = loopsens(G,K2);
47
48 Wii = makeweight(0.30,30,10);
49 WII = [Wii,Wii;
50        Wii,Wii];
51
52 M = -WII*Lp.To;
53 figure
54 sigma(M)
55 hold on
56
57 HinfNorm2 = hinfnorm(M)
58
```

```
59 if HinfNorm2 > 1.0
60     fprintf('Controller is not robust')
61 else
62     fprintf('System is robust')
63 end
64
65 %%
66 Wii = makeweight(0.56,30,10);
67 WII = [Wii,Wii;
68        Wii,Wii];
69
70 M = -WII*Lp.To;
71
72 HinfNorm2 = hinfnorm(M)
73
74 if HinfNorm2 > 1.0
75     fprintf('Controller is not robust')
76 else
77     fprintf('System is robust')
78 end
```

References

- [1] Brian L Stevens, Frank L Lewis, and Eric N Johnson. *Aircraft control and simulation: dynamics, controls design, and autonomous systems*. John Wiley & Sons, 2015.