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

$$\begin{aligned} \text{rank} \begin{pmatrix} \lambda I - A \\ -\sqrt{Q} \end{pmatrix} = n \; \forall \quad \lambda \in \mathbb{C}^+ \\ \text{rank of} \begin{pmatrix} \sqrt{Q} \\ \sqrt{Q}A \end{pmatrix} = 2 \end{aligned} \tag{1}$$

   So it is observable which implies the dynamics is detectable.

   Simulink Diagram

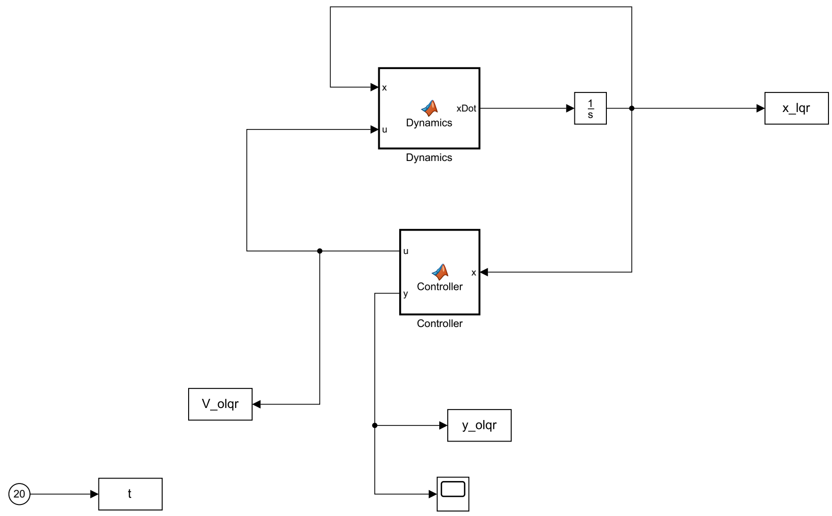

Figure 1: Closed Loop Model with the Dynamics and Controller for simulation

Results

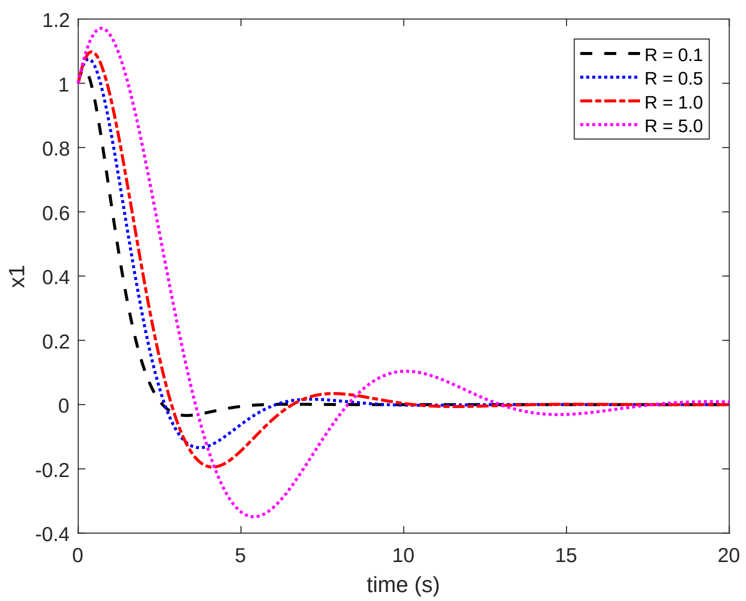

Figure 2: Trajectories of state x_1 during the simulation. For all values of R, the controller regulates the state from 1.0 to zero.

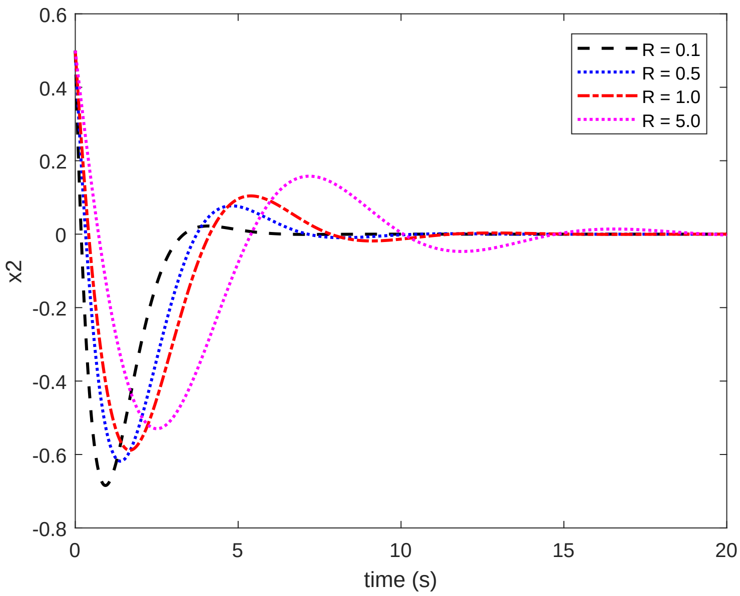

Figure 3: Trajectories of state x_2 during the simulation. For all values of R, the controller regulates the state from 0.5 to zero.

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 $\rho = 1.0$,

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

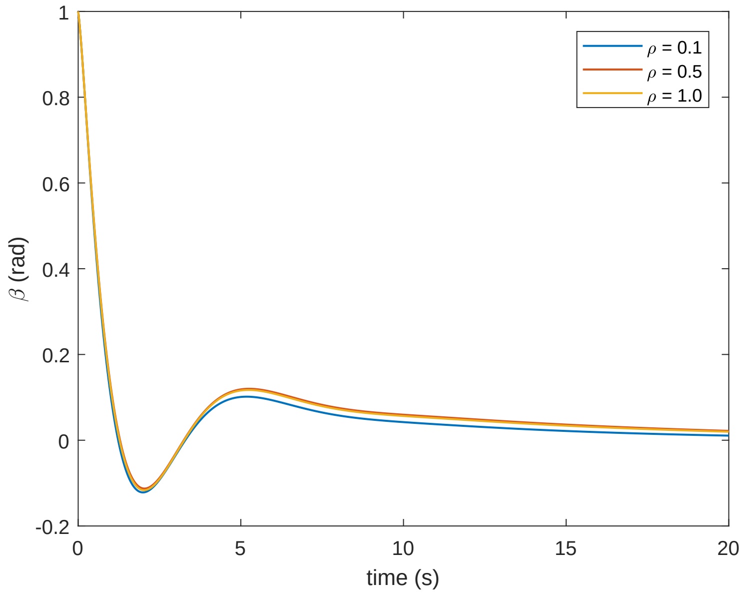

Figure 4: Trajectories of state β during the simulation. For all values of ρ, the controller regulates the state from 1.0 to zero.

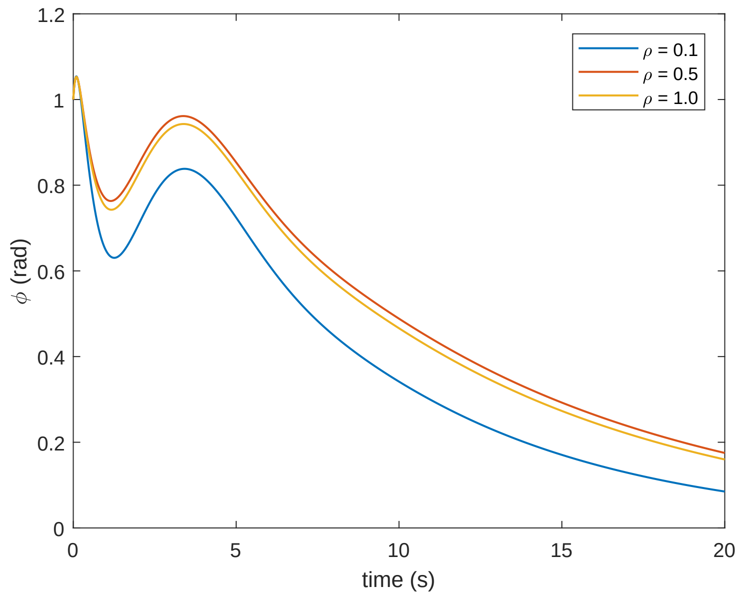

Figure 5: Trajectories of state ϕ during the simulation. For all values of ρ, the controller regulates the state from 1.0 to zero.

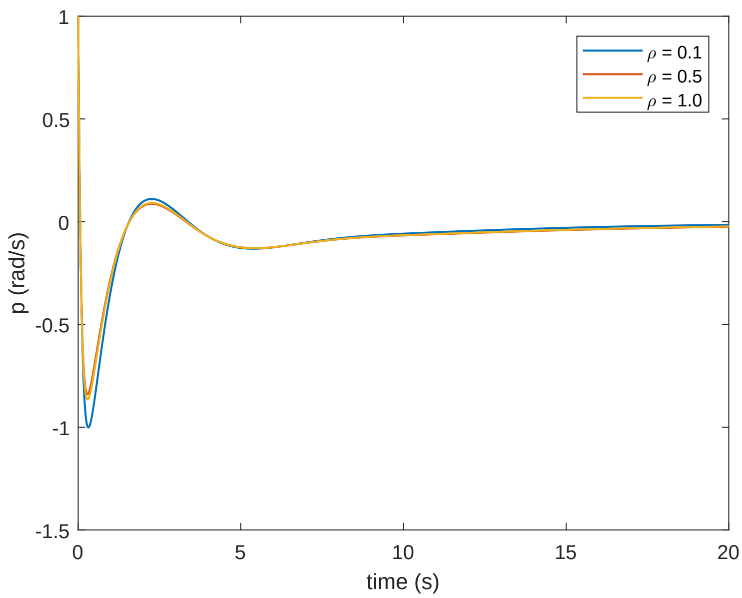

Figure 6: Trajectories of state p during the simulation. For all values of ρ, the controller regulates the state from 1.0 to zero.

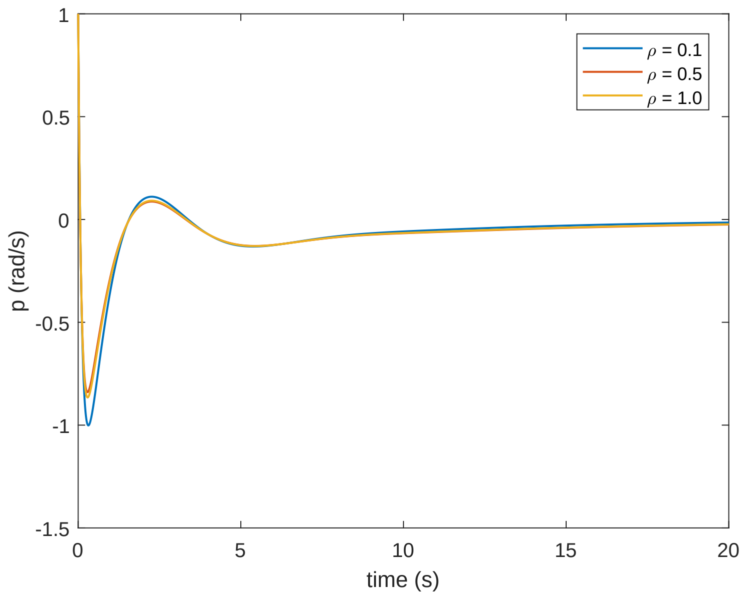

Figure 7: Trajectories of state r during the simulation. For all values of ρ, the controller regulates the state from 1.0 to zero.

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 $\sigma(S_0G)$ to be small.

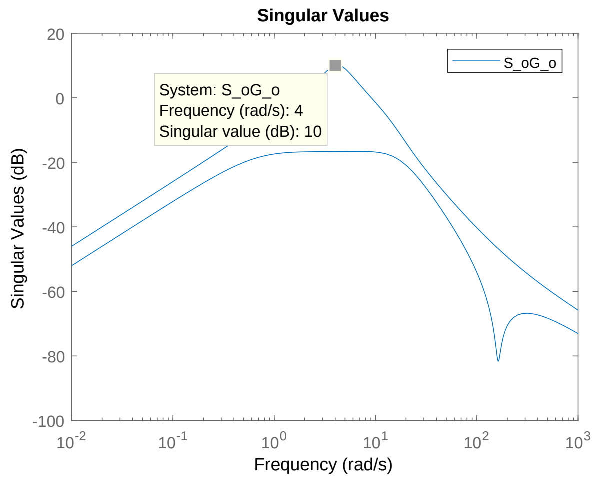

Figure 8: Singular values of S_oG. We want $\omega < 10 rad/s$

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

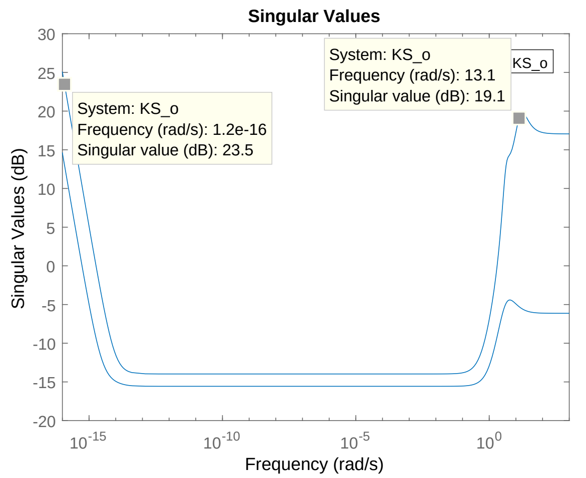

Figure 9: Singular values of KS_o

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

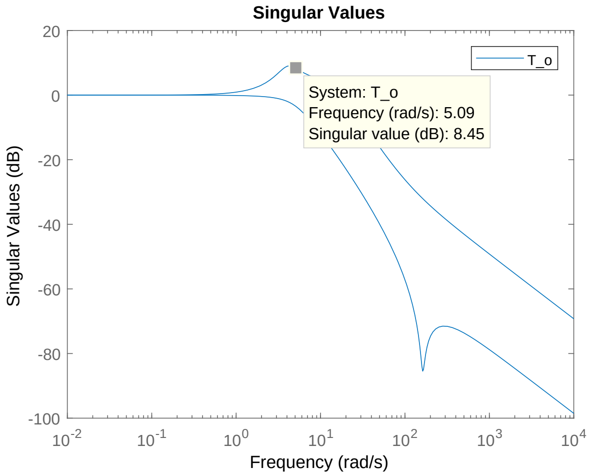

Figure 10: Singular values of T_o

(b) Plot of two weighting functions

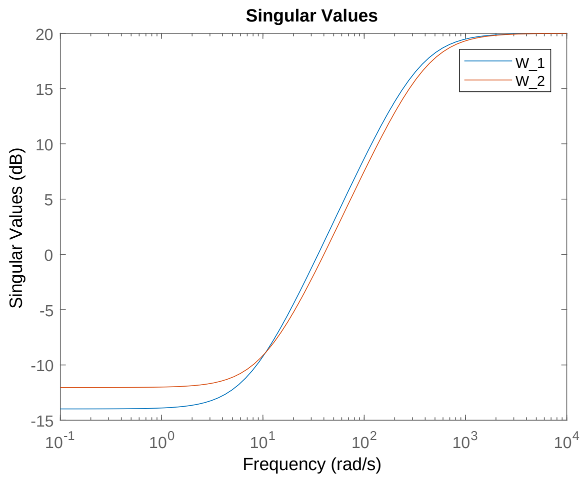

Figure 11: Plot of weighting functions.

(c) Robustness

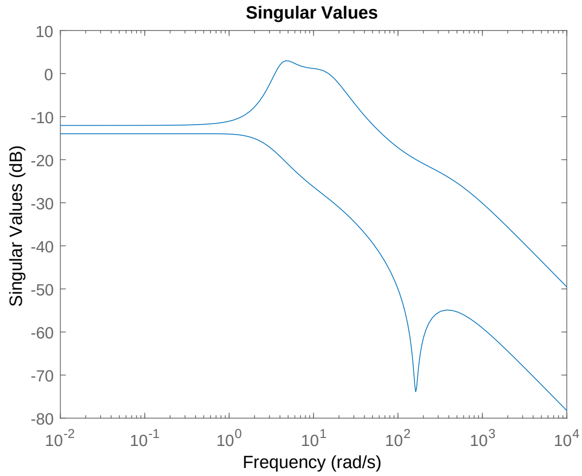

Figure 12: Plot of weighting functions.

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

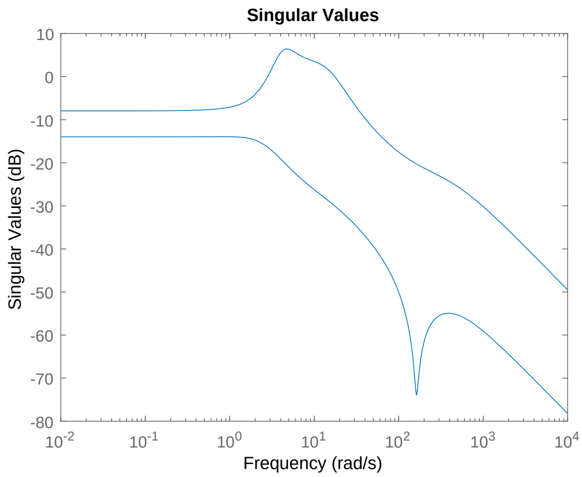

Figure 13: Plot of weighting functions when second channel is increased to 40 %

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

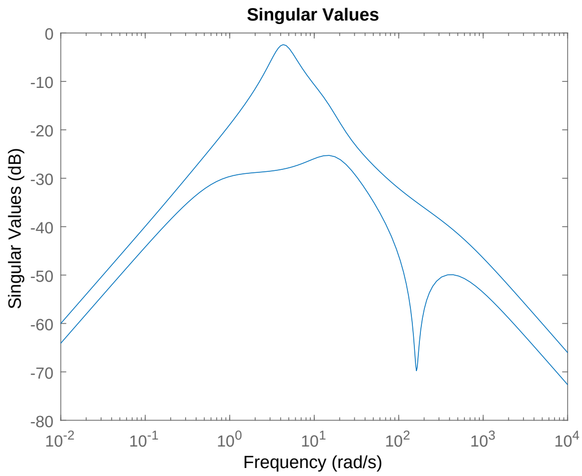

Figure 14: Plot of weighting functions when second channel is increased to 40 %

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

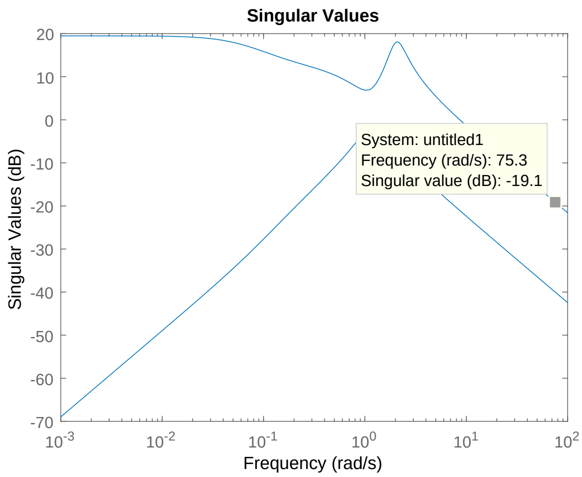

Figure 15: Singular value plot

   - For K2

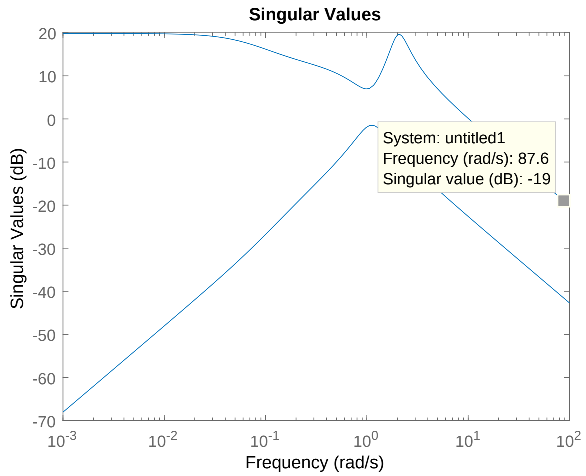

Figure 16: Singular value plot

- For K3

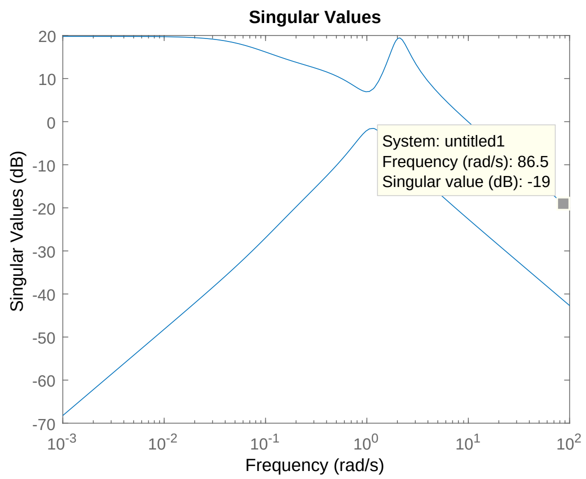

Figure 17: Singular value plot

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

(b) Output multiplicative uncertainty

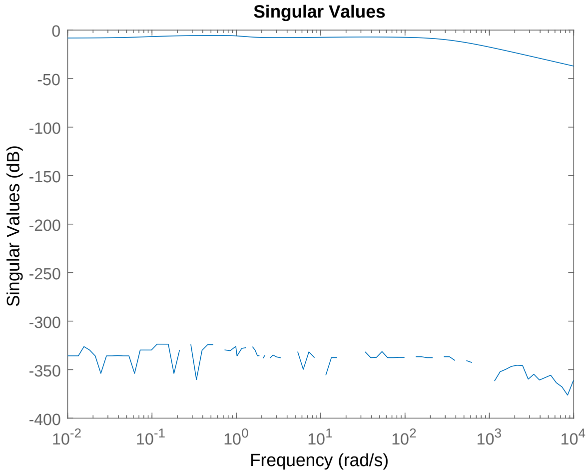

Figure 18: Singular value plot

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**

```
%% MAE 5010- Atmospheric Flight Control HW3
% LQ Regulator Prob 2
% Sandesh Thapa

 clear all;
 close all
 clc;
%%
A = [0 1;0 0];
B = [0 1]';
C = [1 1];
Q = eye(2);
x0 = [1 0.5]';
X = x0*x0';
R = 5.0;

 [K,J,P] = OutputLQRProb2(A,B,C,Q,R,x0);
%%
sim('Sim_OutputLQR')
%%
```

```
figure
plot(t,x_lqrA(:,1),'--k',t,x_lqrB(:,1),':b',t,x_lqrC(:,1),'-.r',t,x_lqrD(:,1),':
    m','Linewidth',1.5);
legend('R = 0.1','R = 0.5','R = 1.0','R = 5.0')
xlabel('time (s)')
ylabel('x1')
hold on

figure
plot(t,x_lqrA(:,2),'--k',t,x_lqrB(:,2),':b',t,x_lqrC(:,2),'-.r',t,x_lqrD(:,2),':
    m','Linewidth',1.5);
legend('R = 0.1','R = 0.5','R = 1.0','R = 5.0')
xlabel('time (s)')
ylabel('x2')
hold on

%%
figure
plot(t,x_lqrA(:,1));
legend('R = 0.1')
xlabel('time (s)')
ylabel('x1')
hold on

figure
plot(t,x_lqrA(:,2));
legend('R = 0.1')
xlabel('time (s)')
ylabel('x2')
hold on
```

2. **Output LQR Function Problem 2**

```
function [K_k,J_k,P_k] = OutputLQRProb2(A,B,C,Q,R,x0)
% function [K_k,J_k,P_k] = OutputLQRProb2(A,B,C,Q,R,x0) --> Output LQR
% solver based on Moerder and Calise (1985) for Prob 2
% x0 = [1 0]';
X = x0*x0';
K_k = 5;

% K_k = ones(4,2);

A_k = A - B*K_k*C;
eig(A_k);
```

```
Jt = zeros(1000,1);
JtNew = zeros(1000,1);
Kt = zeros(1000,1);
alpha = 0.5;
for k = 1:1000
    %     K_k = K_0;
    A_k = A - B*K_k*C;
    P_k = lyap(A_k,Q + C'*K_k'*R*K_k*C);
    S_k = lyap(A_k,X);
    J_k = 1/2*trace(P_k*X);
    Jt(k) = J_k;
    DeltaK = inv(R)*B'*P_k*S_k*C'*inv(C*S_k*C') - K_k;

    for i = 1:1000
        K_kNew = K_k + alpha*DeltaK;
        A_k = A - B*K_kNew*C;
        P_kNew = lyap(A_k,Q + C'*K_kNew'*R*K_kNew*C);
        S_k = lyap(A_k,X);
        EigAk = eig(A_k);

        J_kNew = 1/2*trace(P_kNew*X);
        JtNew(i) = J_kNew;
        Delta_J = J_k - J_kNew;

        if max(real(EigAk)) < 0 && J_kNew < J_k
            K_k = K_kNew;
            Kt(k) = K_k;
            J_k = J_kNew;
            Jt(i) = J_k;
            break
        else
            alpha = alpha/10;
            K_kNew = K_k + alpha*DeltaK;
            K_k = K_kNew;
            Kt(k) = K_k;
            A_k = A - B*K_k*C;
            EigAk = eig(A_k);
            P_kNew = lyap(A_k,Q + C'*K_kNew'*R*K_kNew*C);
            S_k = lyap(A_k,X);
            J_kNew = 1/2*trace(P_kNew*X);
            J_k = J_kNew;
            Jt(i) = J_k;
            DeltaK = inv(R)*B'*P_k*S_k*C'*inv(C*S_k*C') - K_k;
```

```
            end
        end

        if Delta_J < 0.0000001
            K_k = K_kNew;
            Kt(k) = K_k;
            J_k = J_kNew;
            break
        end

end
```

3. **Problem 3 & 5**

```
clear all;
close all
clc;
%%
A = [-0.131150   0.14858    0.32434       -0.93964;
          0.0        0.0        1.0        0.33976;
      -10.614          0.0     -1.1793        1.023;
      0.99655            0.0    -0.001874   -0.25855];

B = [0.00012       0.00032897;
     0.0           0.0;
     -0.1031578    0.020987;
     -0.0021330    -0.010715];

C = [0 0 57.29578 0;
     0 0 0 57.29578];

 D = zeros(2,2);

qdr = 50;
qr = 100;
% v = [qdr,qr,qr,qdr,0,0,0];
Q = diag([qdr,qr,qr,qdr]);
x0 = [1.0 1.0 1.0 1.0]';
X = x0*x0';
rho = 1.0;
n = 3;
% for j = 1:n
R = rho*eye(2);

```

```
[K,J,P] = OutputLQRProb3(A,B,C,Q,R,x0);

%%
% K = K_k;
sim('Sim_OutputLQR')
%%
% end
% for j = 1:n
figure(1)
plot(t,x_lqrA(:,1),t,x_lqrB(:,1),t,x_lqrC(:,1),'Linewidth',1.0);
legend('\rho = 0.1','\rho = 0.5','\rho = 1.0')
xlabel('time (s)')
ylabel('\beta (rad)')
hold on

figure(2)
plot(t,x_lqrA(:,2),t,x_lqrB(:,2),t,x_lqrC(:,2),'Linewidth',1.0);
legend('\rho = 0.1','\rho = 0.5','\rho = 1.0')
xlabel('time (s)')
ylabel('\phi (rad)')
hold on

figure(3)
plot(t,x_lqrA(:,3),t,x_lqrB(:,3),t,x_lqrC(:,3),'Linewidth',1.0);
legend('\rho = 0.1','\rho = 0.5','\rho = 1.0')
xlabel('time (s)')
ylabel('p (rad/s)')
hold on

figure(4)
plot(t,x_lqrA(:,4),t,x_lqrB(:,4),t,x_lqrC(:,4),'Linewidth',1.0);
legend('\rho = 0.1','\rho = 0.5','\rho = 1.0')
xlabel('time (s)')
ylabel('r (rad/s)')
hold on
```

4. **Output LQR Function Problem 3**

```
function [K_k,J_k,P_k] = OutputLQRProb3(A,B,C,Q,R,x0)
K_k = [-0.5 -1.0;0.5 -1.0];
X = x0*x0';
A_k = A - B*K_k*C;
eig(A_k)

```

```
Jt = zeros(1000,1);
JtNew = zeros(1000,1);
Kt = zeros(1000,1);
alpha = 0.5;
for k = 1:1000
    %     K_k = K_0;
    A_k = A - B*K_k*C;
    P_k = lyap(A_k,Q + C'*K_k'*R*K_k*C);
    S_k = lyap(A_k,X);
    J_k = 1/2*trace(P_k*X);
    Jt(k) = J_k;
    DeltaK = inv(R)*B'*P_k*S_k*C'*inv(C*S_k*C') - K_k;

    for i = 1:1000
        K_kNew = K_k + alpha*DeltaK;
        A_k = A - B*K_kNew*C;
        P_kNew = lyap(A_k,Q + C'*K_kNew'*R*K_kNew*C);
        S_k = lyap(A_k,X);
        EigAk = eig(A_k);

        J_kNew = 1/2*trace(P_kNew*X);
        JtNew(i) = J_kNew;
        %         if  i > 1
        %             Delta_J(i) = Jt(i) - Jt(i+1);
        Delta_J = J_k - J_kNew;

        if max(real(EigAk)) < 0 && J_kNew < J_k
            K_k = K_kNew;
%             Kt(k) = K_k;
            J_k = J_kNew;
            Jt(i) = J_k;
            break
        else
            alpha = alpha/10;
            K_kNew = K_k + alpha*DeltaK;
            K_k = K_kNew;
%             Kt(k) = K_k;
            A_k = A - B*K_k*C;
            EigAk = eig(A_k);
            P_kNew = lyap(A_k,Q + C'*K_kNew'*R*K_kNew*C);
            S_k = lyap(A_k,X);
            J_kNew = 1/2*trace(P_kNew*X);
            J_k = J_kNew;
```

```
                Jt(i) = J_k;
                DeltaK = inv(R)*B'*P_k*S_k*C'*inv(C*S_k*C') - K_k;
            end
        end

        if Delta_J < 0.0000001

            K_k = K_kNew;
%             Kt(k) = K_k;
            J_k = J_kNew;
            break
        end

end
```

5. **Problem 4**

```
%% Problem 4
%% (a)

clear all;
close all;
clc;

sys11 = tf([ 0 0 6],[0.09 1 1]);
sys12 = tf([0 -0.05],[0.1 1]);
sys21 = tf([0 0.07],[0.3 1]);
sys22 = tf([0 0 5],[0.108 1.74 -1]);

G0 = [sys11, sys12;
        sys21, sys22];

K11 = tf([2 2],[1 0]);
K12 = tf([-1 0],[3 1]);
K21 = tf([-5 -5],[0.8 1]);
K22 = tf([4*0.7 4],[1 0]);

K = [K11,K12;
      K21 K22];

loops = loopsens(G0,K);

% 4.1.i input di to y
figure
```

```
sigma(loops.So*G0)
legend('S_oG_o')
hold on

% u --> r
figure
sigma(K*loops.So)
legend('KS_o')
hold on

% n --> y small max(svd(T_o)
figure
sigma(loops.To)
legend('T_o')
hold on
%% 4(b)
W1 = makeweight(0.20,35,10);
W2 = makeweight(0.25,40,10);

figure
sigma(W1)
hold on
sigma(W2)
legend('W_1', 'W_2')
hold on

%% 4(c)
WI = [W1, 0; 0,W2];
M = -WI*loops.Ti;

figure
sigma(M)

HinfNorm1 = hinfnorm(M)
if HinfNorm1 > 1.0
    fprintf('Controller is not robust')
else
    fprintf('System is robutst')
end


%% 4(c) Part b
W2New = makeweight(0.40,40,10);
```

```
WId = [WI, 0; 0,W2New];
M = -WId*loops.Ti;

figure
sigma(M)
HinfNorm2 = hinfnorm(M)

if HinfNorm2 > 1.0
    fprintf('Controller is not robust')
else
    fprintf('System is robutst')
end


%% 4(d)
M_d = loops.PSi*WI;

figure
sigma(M_d)
HinfNormM_d = hinfnorm(M_d)

if HinfNormM_d > 1.0
    fprintf('Controller is not robust')
else
    fprintf('System is robutst')
end
```

6. **Problem 5**

```
%% Problem 5(a)

A = [-0.131150   0.14858    0.32434      -0.93964;
            0.0         0.0         1.0       0.33976;
        -10.614           0.0    -1.1793         1.023;
        0.99655             0.0    -0.001874   -0.25855];

B = [0.00012       0.00032897;
      0.0            0.0;
      -0.1031578  0.020987;
      -0.0021330   -0.010715];

C = [0  0  57.29578  0;
      0  0  0  57.29578];

```

```
D = zeros(2,2);
K1 = [   -1.1914    -0.7028;
          0.6179    -1.0284];

K2 = [ -1.4810    -0.6327;
      0.6555    -1.0136];

K3 =    [-1.4414    -0.6528;
      0.6468    -1.0087];

G = tf(ss(A,B,C,D));
% G0 = ss(A,B,C,D,2);
% Lp = loopsens(G0,K1)
% figure
% sigma(Lp.To)
% hold on

figure
sigma(G*K1)
hold on

figure
sigma(G*K2);
hold on

figure
sigma(G*K3);
hold on

%% 5(b)
Lp = loopsens(G,K2);

Wii = makeweight(0.30,30,10);
WII = [Wii,Wii;
       Wii,Wii];

M = -WII*Lp.To;
figure
sigma(M)
hold on

HinfNorm2 = hinfnorm(M)

```

```
if HinfNorm2 > 1.0
    fprintf('Controller is not robust')
else
    fprintf('System is robutst')
end

%%
Wii = makeweight(0.56,30,10);
WII = [Wii,Wii;
       Wii,Wii];

M = -WII*Lp.To;

HinfNorm2 = hinfnorm(M)

if HinfNorm2 > 1.0
    fprintf('Controller is not robust')
else
    fprintf('System is robutst')
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.