# Hard Disk Design

## Table of Contents



# Task 1

The differential equation of input voltage and head position is given, so it is pretty easy to get the open loop transfer function by taking the lapalce transform of both side and can easily get

$$\frac{Y(s)}{U(s)} = \frac{1}{Js^2 + bs}$$

# Task 2

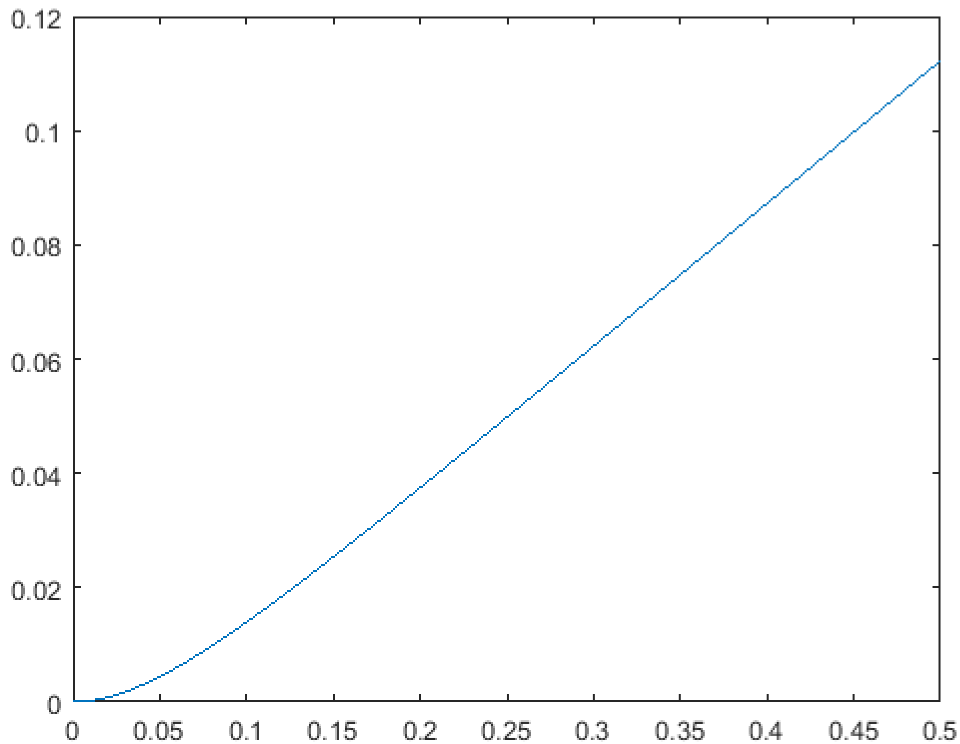
open loop transfer function is given by multiplying both G1 and G2

$$G1(s)G2(s) = \frac{Km}{(Ls + R) * (Js^2 + bs)} = \frac{Km}{LJs^3 + Lbs^2 + RJs^2 + Rbs}$$

```
s = tf('s');
J = 1;
b = 20;
R = 1;
L = 0.001;
Km = 5;
G1 = Km/(L*s+R)
G2 = 1/(J*s^2+b*s)
openTF = G1*G2
t=[0:0.005:0.5];
y = step(openTF,t);
plot(t,y);
```

```
G1 =

        5
  -----------
```

```
  0.001 s + 1

Continuous-time transfer function.


G2 =

        1
  ----------
  s^2 + 20 s

Continuous-time transfer function.


openTF =

                 5
  ---------------------------
  0.001 s^3 + 1.02 s^2 + 20 s

Continuous-time transfer function.
```




You can see that if a constant voltage is applied, then the read head will moves in a constant speed. And at the begining when the voltage is applied, there is a curve, which indicates that the read head is accelerating under applied voltage.

# Task 3a

The proportional compensator is applied, so that the open loop transfer function is given by

$$Ka * G1(s) * G2(s)$$

and closed loop transfer function is given by

$$\frac{Ka * G1(s) * G2(s)}{1 + Ka * G1(s) * G2(s)}$$

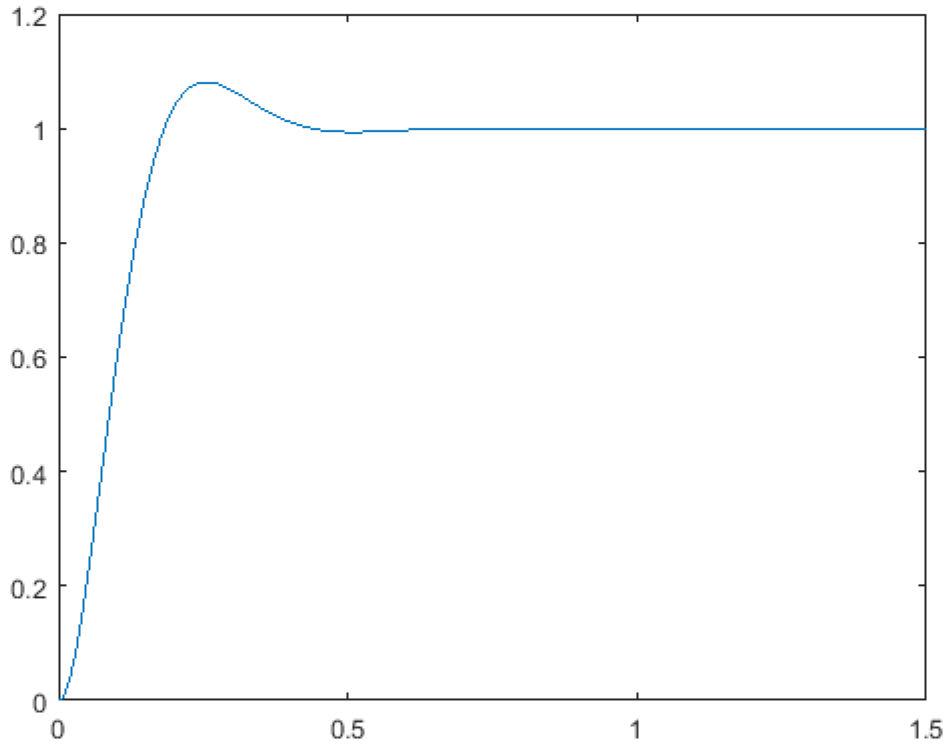
after plug in G1(s)G2(s) from previous computation, we have

$$\frac{Ka * Km}{LJs^3 + (Lb + RJ)s^2 + Rbs + KaKm}$$

Try plugging in Ka = 50, we can get the following plot

```
Ka = 50;
t=[0:0.005:1.5];
ProportionalTF =(Ka*G1*G2)/(1+Ka*G1*G2)
y = step(ProportionalTF, t);
plot(t,y);
```

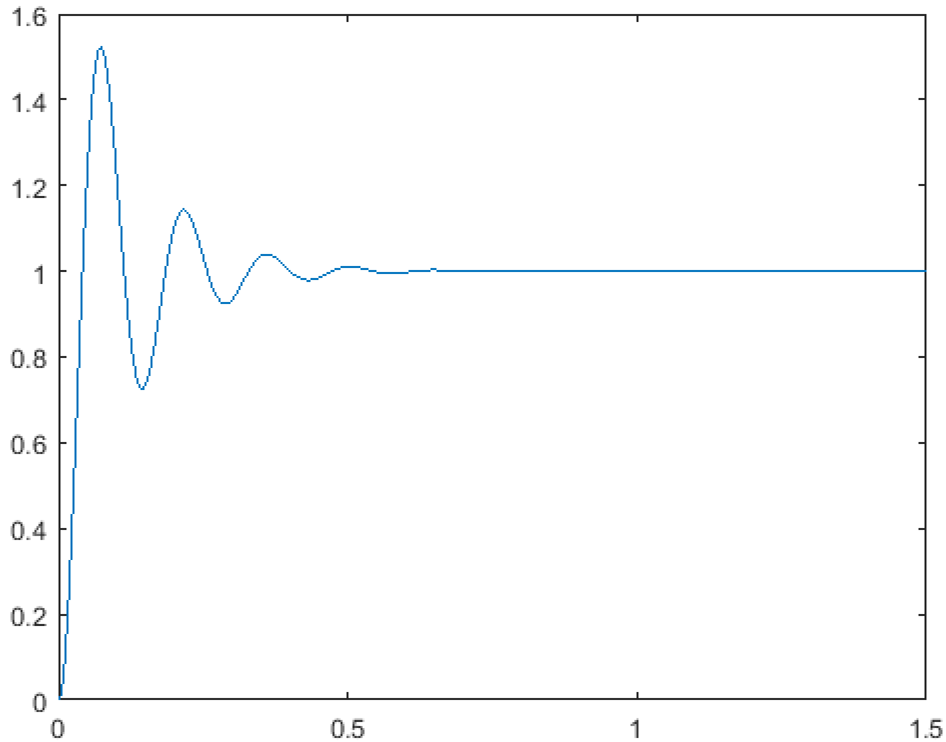
```
ProportionalTF =

                     0.25 s^3 + 255 s^2 + 5000 s
  -----------------------------------------------------------------
  1e-06 s^6 + 0.00204 s^5 + 1.08 s^4 + 41.05 s^3 + 655 s^2 + 5000 s

Continuous-time transfer function.
```




Then applied step when Ka = 400

```
Ka = 400;
ProportionalTF = (Ka*openTF)/(1+Ka*openTF)
y = step(ProportionalTF, t);
plot(t,y);
```

```
ProportionalTF =

                      2 s^3 + 2040 s^2 + 40000 s
  -----------------------------------------------------------------
  1e-06 s^6 + 0.00204 s^5 + 1.08 s^4 + 42.8 s^3 + 2440 s^2 + 40000 s

Continuous-time transfer function.
```




Clearly you can see that the Ka=50 has less over shoot than Ka=400 does. This is because for Ka=400, the feedback amplification is too big and make system too sensitive.

# Task 3b

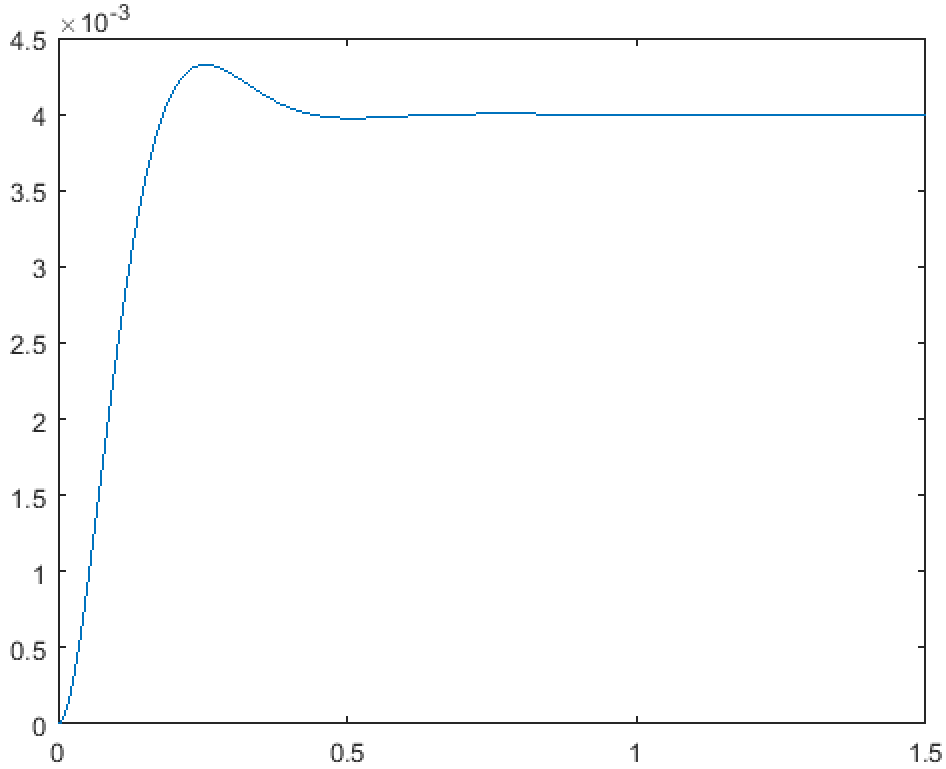
So when disturbance is Introduced to the system, the disturbance input flows into system between G1 and G2. So that we can derive the transfer function for disturbance:

$$Tw = \frac{G2(s)}{1 + Ka * G1(s) * G2(s)}$$

First plot the response of disturbance in case of Ka = 50

```
Ka = 50;
Tw = G2/(1+Ka*G1*G2)
y = step(Tw, t);
plot(t,y);


Tw =

             0.001 s^3 + 1.02 s^2 + 20 s
  --------------------------------------------------
  0.001 s^5 + 1.04 s^4 + 40.4 s^3 + 650 s^2 + 5000 s
```

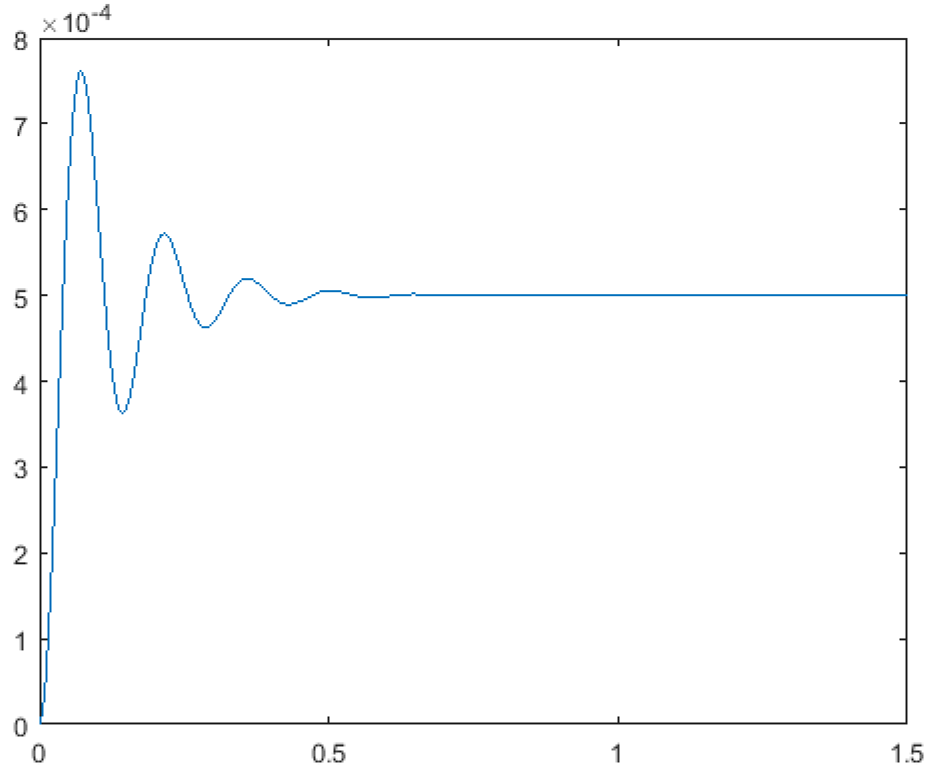
*Continuous-time transfer function.*




Then plot the response of disturbance in case of Ka = 400

```
Ka = 400;
Tw = G2/(1+Ka*G1*G2)
y = step(Tw, t);
plot(t,y);
```

```
Tw =

              0.001 s^3 + 1.02 s^2 + 20 s
  ----------------------------------------------------
  0.001 s^5 + 1.04 s^4 + 40.4 s^3 + 2400 s^2 + 40000 s

Continuous-time transfer function.
```

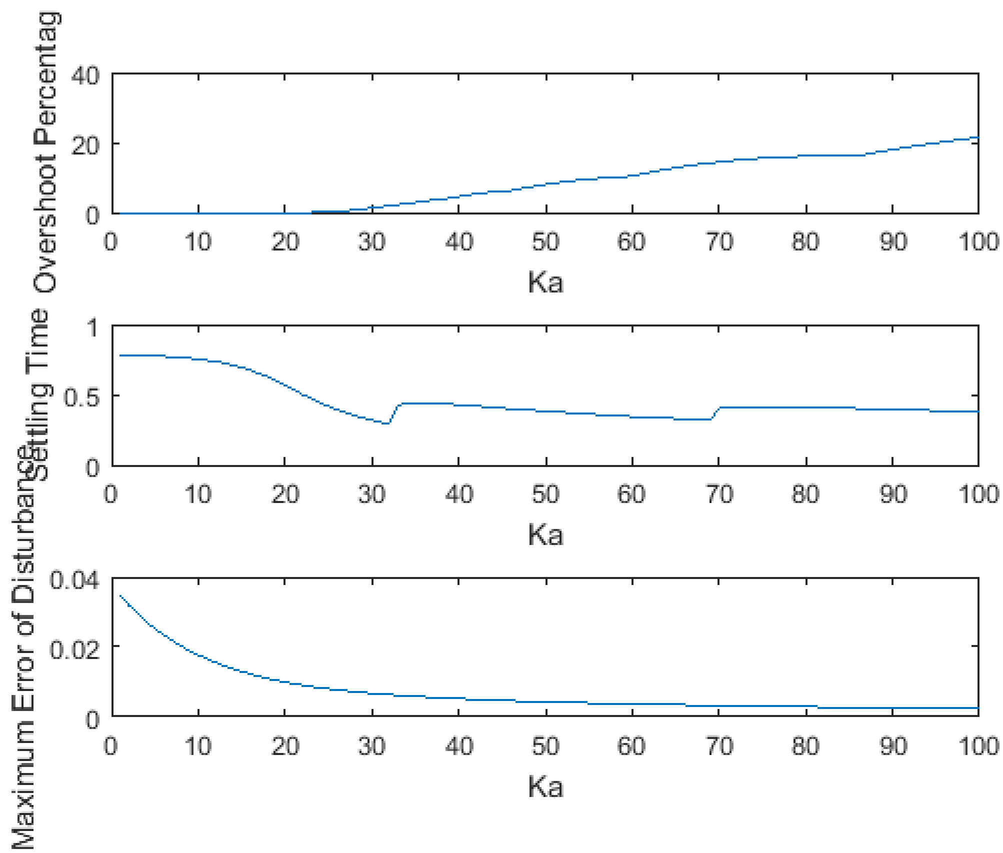



# Task 3C

We can plot overshoot percentage and settling time as a function of Ka, and analyze the optiomal Ka for the system

```
overshoot = [];
settlingTime = [];
maxError = [];
n=[1:100];
t=[0:0.05:0.8];
for  Ka=n
    ProportionalTF = (Ka*openTF)/(1+Ka*openTF);
    y = step(ProportionalTF, t);
    info = stepinfo(y, t, 'SettlingTimeThreshold', 0.02);
    overshoot = [overshoot, info.Overshoot];
    settlingTime = [settlingTime info.SettlingTime];

    Tw = G2/(1+Ka*G1*G2);
    y = step(Tw, t);
    maxError = [maxError max(y)];
end
subplot(3,1,1);
plot(n,overshoot);xlabel('Ka');ylabel('Overshoot Percentage');
subplot(3,1,2);
plot(n,settlingTime);xlabel('Ka');ylabel('Settling Time');
subplot(3,1,3);
```

```
plot(n,maxError);xlabel('Ka');ylabel('Maximum Error of Disturbance');
```




For overshoot less than 5%, Ka is required to be equal or less than 41, Ka value that satisfy settling time requirement is more than 100 and is not in the graph. For disturbance less than 0.005, Ka is required to be equal or bigger than 41 Clearly, you cannot satisfy three requirements at the same time.

# Task 4

The closed loop transfer function in this case would be

$$\frac{KaG1(s)G2(s)}{1 + KaH(s)G1(s)G2(s)}$$

So both Ka and Kh are varying, so we can plot a 3-D graph in which x aixs is Ka and Y axis is Kh and Z axis is the property under examination like Settling Time, Overshoot and disturbance. We can find the candidate that sastisfies all three constraints by finding the overlap of Ka and Kh values that satisfies three contraints.

```
[KaRange, KhRange] = meshgrid(55:65, 0:0.01:0.1);
overshootMatrix = [];
settlingTimeMatrix = [];
candidatePairs = [];
t=[0:0.05:0.8];
G12= G1*G2;
```

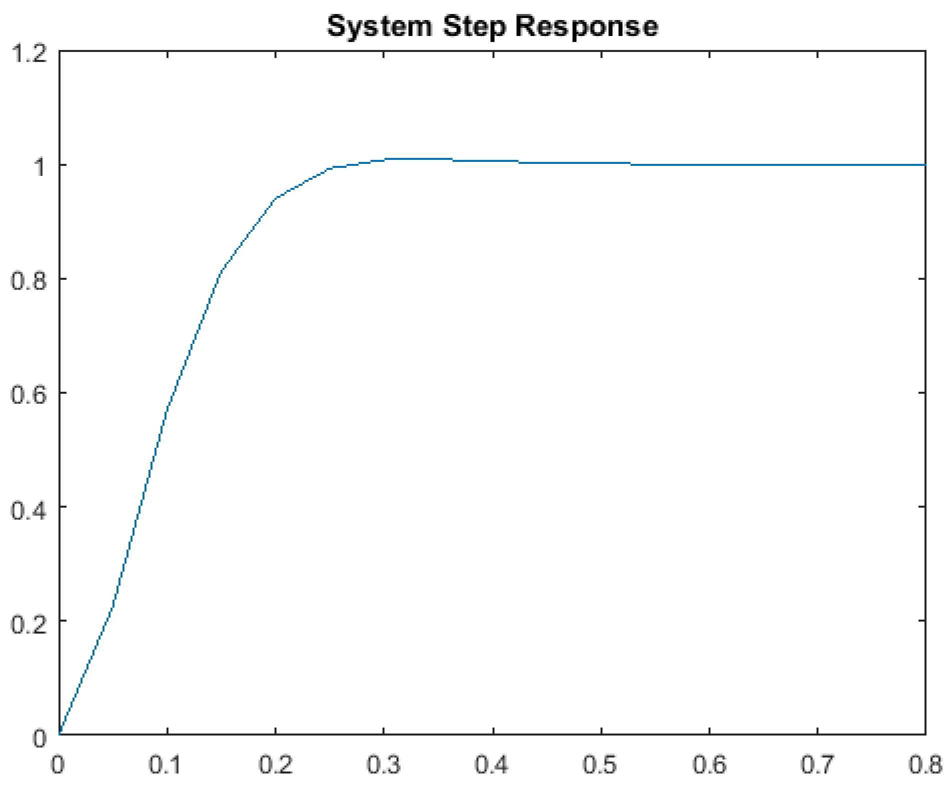
```
for Kh = KhRange(:,1)'
    %overshootArr=[];
    %settlingTimeArr=[];
    for Ka = KaRange(1,:)
        CLTF = (Ka*G12)/(1+Ka*(1+Kh*s)*G12);
        y = step(CLTF, t);
        info = stepinfo(y, t, 'SettlingTimeThreshold', 0.02);
        %overshootArr = [overshootArr, info.Overshoot];
        %settlingTimeArr = [settlingTimeArr, info.SettlingTime];
        Tw = G2/(1+Ka*(1+Kh*s)*G12);
        y = step(Tw,t);
        maxDisturbance = max(y);

        if(info.Overshoot <= 5 & info.SettlingTime <0.25 & y<0.005)
            candidatePairs = [candidatePairs; Ka, Kh];
        end
    end
    %overshootMatrix = [overshootMatrix; overshootArr];
    %settlingTimeMatrix = [settlingTimeMatrix; settlingTimeArr];
end
%mesh(KaRange, KhRange, settlingTimeMatrix);
%mesh(KaRange, KhRange, overshootMatrix);
candidatePairs
```

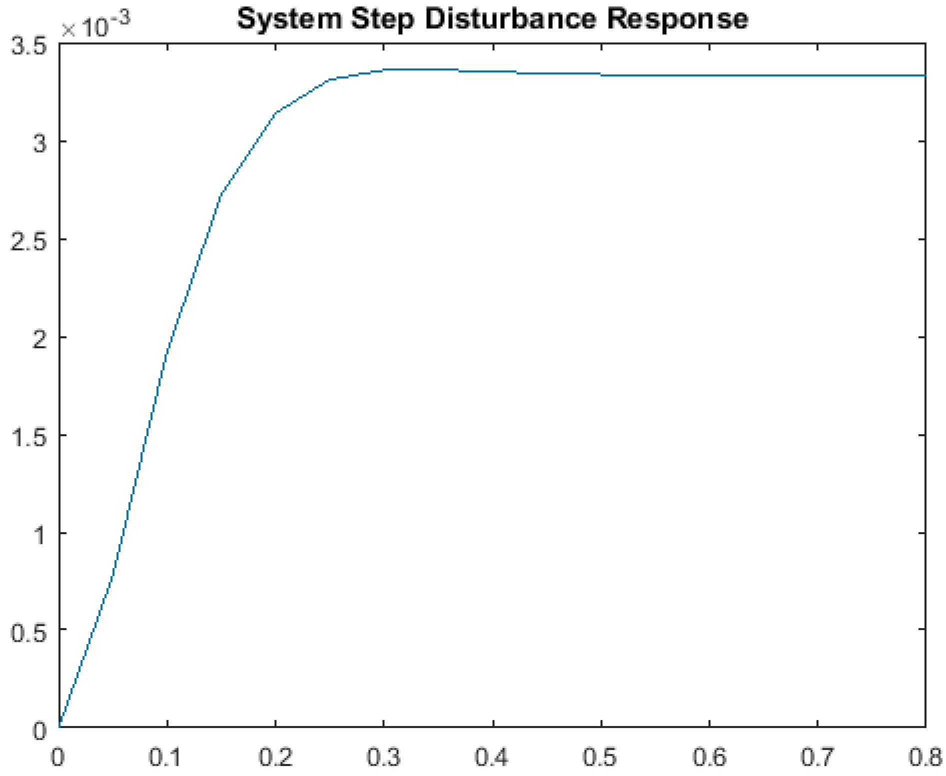
```
candidatePairs =

   56.0000    0.0300
   57.0000    0.0300
   58.0000    0.0300
   59.0000    0.0300
   60.0000    0.0300
   61.0000    0.0300
   62.0000    0.0300
   63.0000    0.0300
   64.0000    0.0300
   65.0000    0.0300
```

We can see that there are bunch of valid pair of Ka and Kh that satisfies the design, we can select Ka = 60 and Kh = 0.03 to examine.

```
Ka = 60; Kh = 0.03;
CLTF = (Ka*G12)/(1+Ka*(1+Kh*s)*G12);
y = step(CLTF, t);
figure(6);
plot(t,y);title('System Step Response');
info = stepinfo(y, t, 'SettlingTimeThreshold', 0.02)


Tw = G2/(1+Ka*(1+Kh*s)*G12);
y = step(Tw,t);
figure(7);
plot(t,y);title('System Step Disturbance Response');
```

```
maxDisturbance = max(y)


info =

         RiseTime: 0.1621
     SettlingTime: 0.2380
      SettlingMin: 0.9401
      SettlingMax: 1.0084
        Overshoot: 0.8390
       Undershoot: 0
             Peak: 1.0084
         PeakTime: 0.3500


maxDisturbance =

    0.0034
```






So all of the contraints are satisfied.

# Task 5

So now the closed loop transfer function becomes

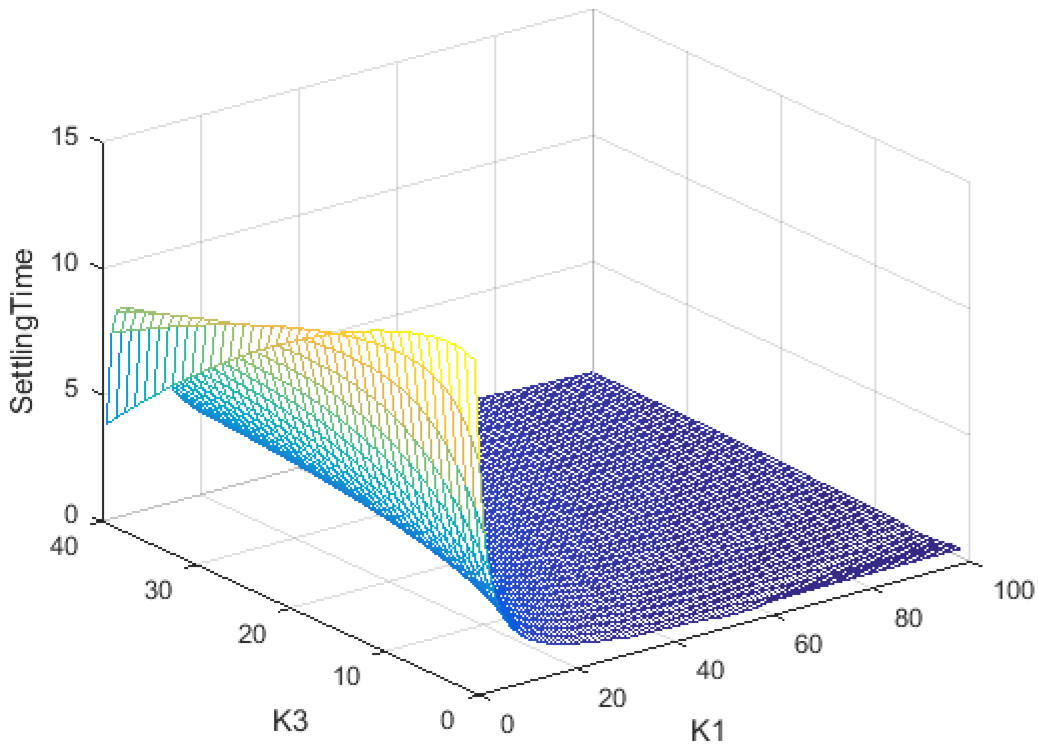
$$CLTF = \frac{F(s)G1(s)G2(s)}{1 + F(s)G1(s)G2(s)}$$

where $F(s) = K1 + K3s$$

we can applied the same precedure as previouse task and get the 3-D graph of constraints we are evaluating.

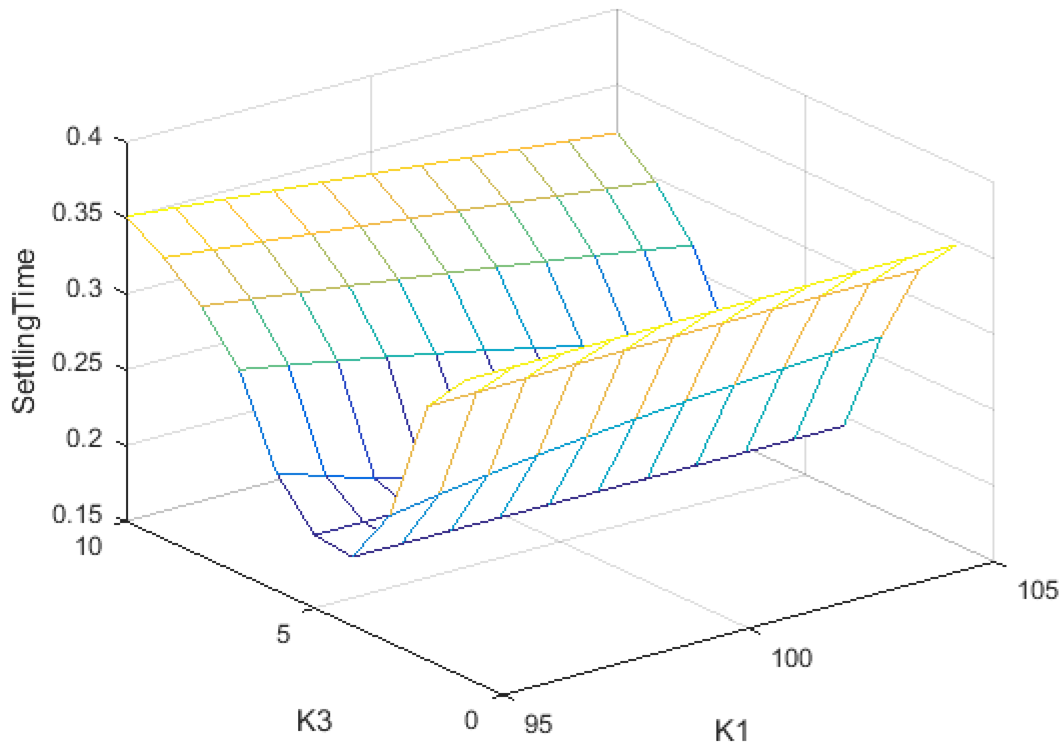
we look at settling time first.

```
openfig('K1-K3 settling time 100x40.fig');
```



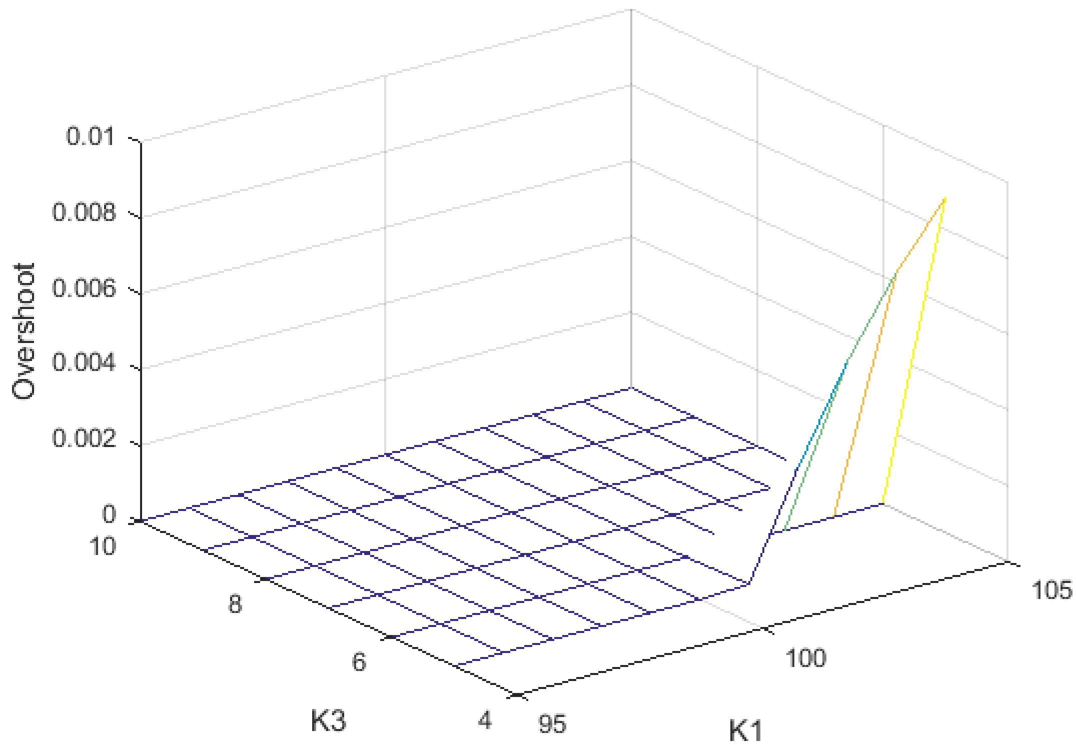

You can see that when K1 is around 100 and K2 is around 5 to 10, settling time is satisfied, then we may take a closer look.

```
openfig('K1-K3 settling time [95,105]x[1,10].fig');
```



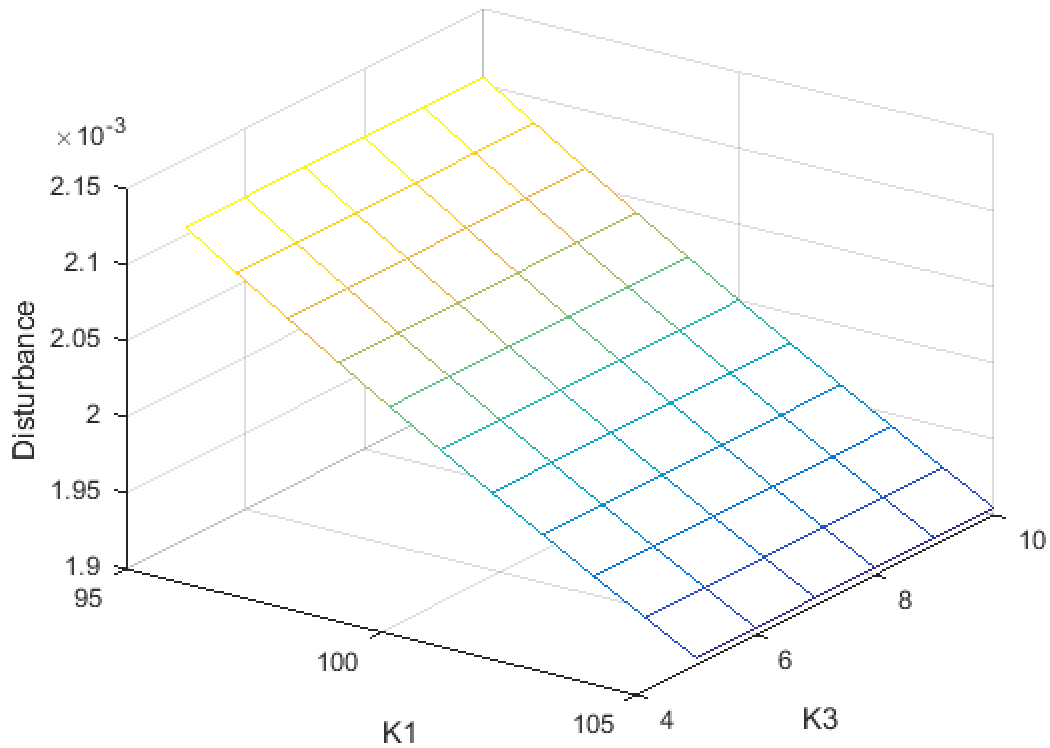

So there are couples of K1 K3 pair meet the settling time reqirement, take K1=100 and K3=5 for example, it is under 250ms. We can also evaluate overshoot around this area.

```
openfig('K1-K3 overshoot [95,105]x[5,10].fig');
```




Observe that K1,K3 = [100 5] has no apparent overshoot. Then we examine the disturbance of this area.

```
openfig('K1-K3 disturbance [95,105]x[5,10].fig');
```




You can see that disturbance of K1,K3=[95, 100] meet the requirement.

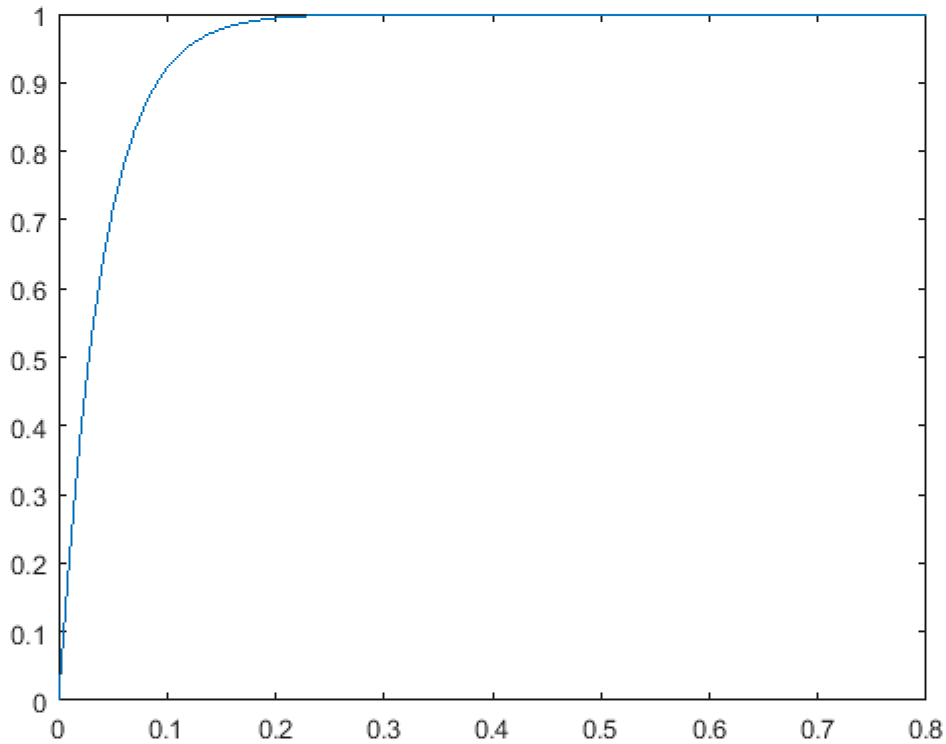
The following is the code to generate graph.

```
% [K1Range K3Range] = meshgrid(95:105, 5:10);
% overshootMatrix = [];
% settlingTimeMatrix = [];
% disturbanceMatrix=[];
% candidatePairs = [];
% t=[0:0.2:15];
% G12= G1*G2;
% for K3 = K3Range(:,1)'
%     overshootArr=[];
%     settlingTimeArr=[];
%     disturbanceArr=[];
%     for K1 = K1Range(1,:)
%         CLTF = ((K1+K3*s)*G12)/(1+(K1+K3*s)*G12);
%         y = step(CLTF, t);
%         info = stepinfo(y, t, 'SettlingTimeThreshold', 0.02);
%         overshootArr = [overshootArr, info.Overshoot];
%         settlingTimeArr = [settlingTimeArr, info.SettlingTime];
%
%         Tw = G2/(1+(K1+K3*s)*G12);
%         y = step(Tw,t);
%         maxDisturbance = max(y);
%         disturbanceArr = [disturbanceArr, maxDisturbance];
%     end
```

```
%     overshootMatrix = [overshootMatrix; overshootArr];
%     settlingTimeMatrix = [settlingTimeMatrix; settlingTimeArr];
%     disturbanceMatrix = [disturbanceMatrix; disturbanceArr];
% end
%
% mesh(K1Range, K3Range,
 overshootMatrix);xlabel('K1');ylabel('K3');zlabel('Overshoot')
% mesh(K1Range, K3Range,
 settlingTimeMatrix);xlabel('K1');ylabel('K3');zlabel('SettlingTime')
% mesh(K1Range,
 K3Range,disturbanceMatrix);xlabel('K1');ylabel('K3');zlabel('Disturbance');view(3
```

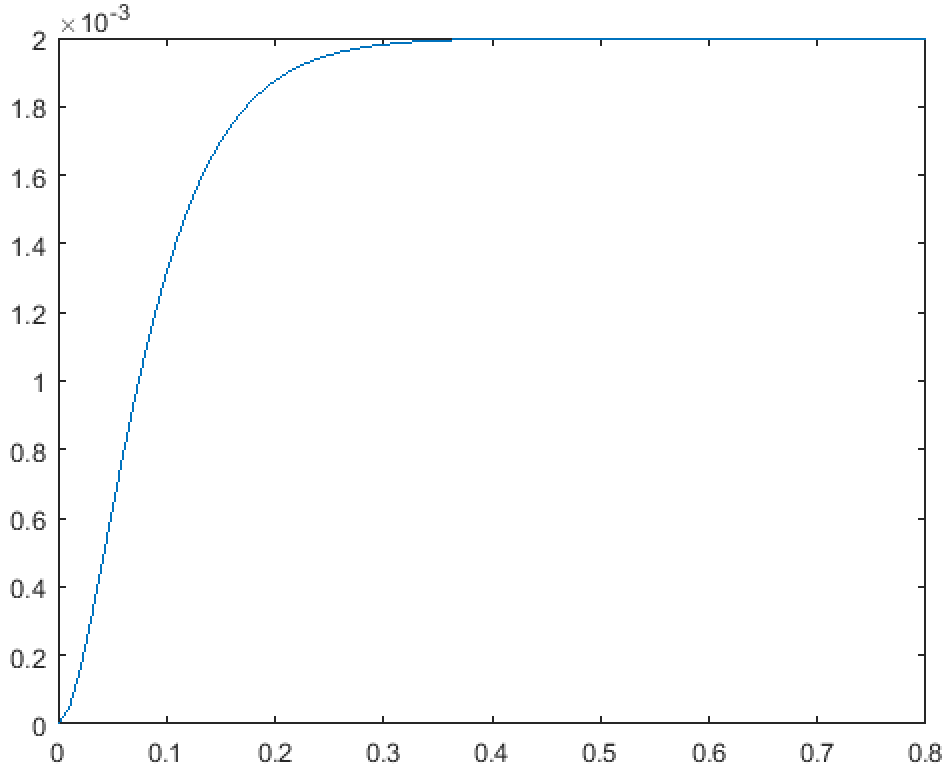
We can take K1=100 and K3=5, and evaluete it more quantatively.

```
t=[0:0.01:0.8];
K1=100; K3=5;
CLTF = ((K1+K3*s)*G12)/(1+(K1+K3*s)*G12);
y = step(CLTF, t);
info = stepinfo(y, t, 'SettlingTimeThreshold', 0.02)
plot(t,y);


info =

        RiseTime: 0.0860
    SettlingTime: 0.1538
     SettlingMin: 0.9211
     SettlingMax: 1.0000
       Overshoot: 0
      Undershoot: 0
            Peak: 1.0000
        PeakTime: 0.8000
```




We see that there is no overshoot because the K3 term acts as a damping factor that prevents the response from overshooting.

```
Tw = G2/(1+(K1+K3*s)*G12);
y = step(Tw,t);
plot(t,y);
```




The disturbance only reaches 2e-3.

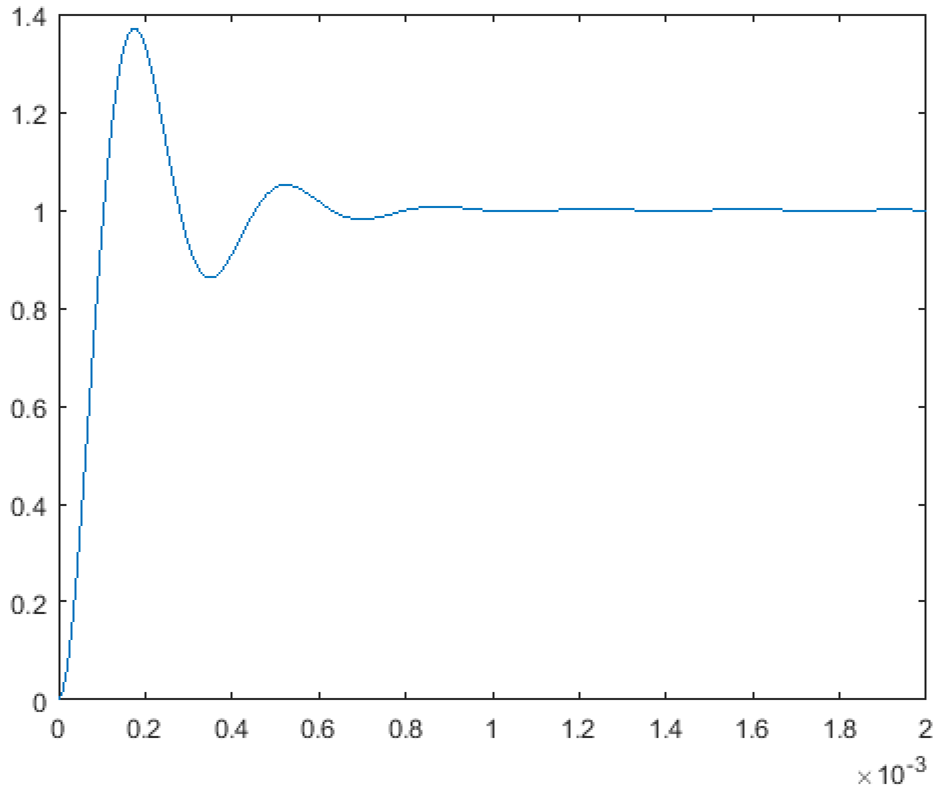
# Task 6A

According to the spec, the transfer function of spring block is

$$\frac{1}{1+\frac{2\zeta * s}{Wn}+\frac{s^2}{Wn^2}}$$

Where $\zeta$ is 0.3 and Wn is 18850

```
Ze = 0.3;
Wn = 18850;
G3 = 1/(1+(2*Ze*s)/Wn+(s/Wn)^2);
t = [0:0.00001:0.002];
y = step(G3,t);
plot(t,y);
```




This is expected output, since second order system is introduced, the oscillation is due to spring.

# Task 6B

The closed loop transfer function becomes

\frac{F(s)G1(s)G2(s)G3(s)}{1+F(s)G1(s)G2(s)G3(s)}

Where $F(s) = K1 + K3s$ $

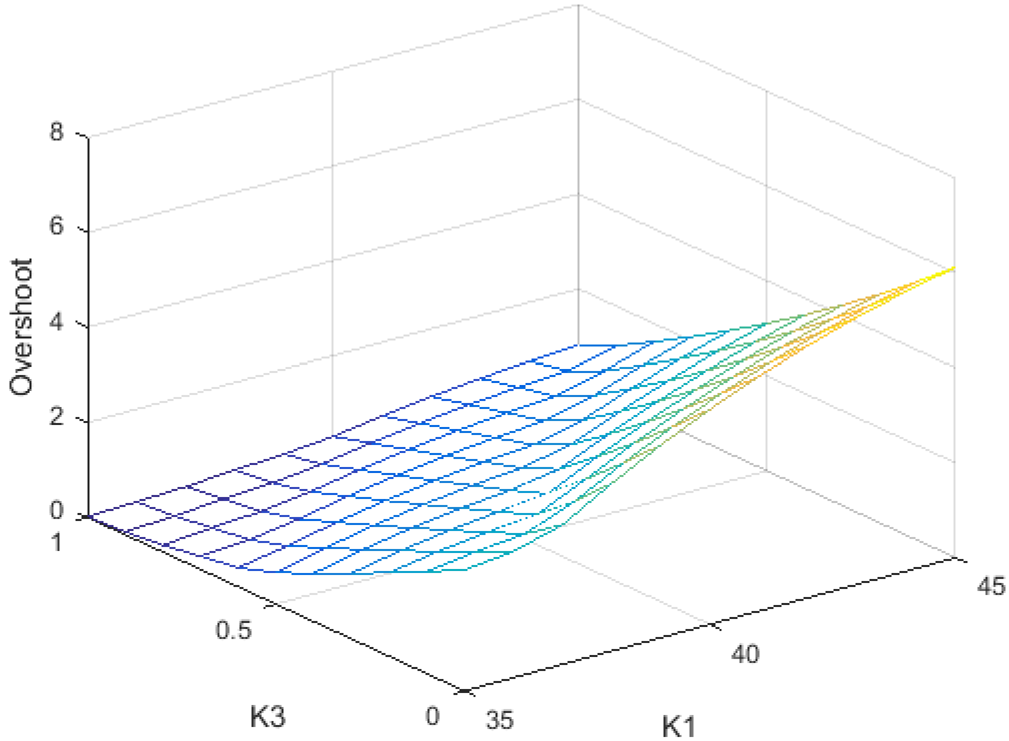
It is found that resonant frequency requirement is met in region where K1=40 and K3=1, So we can plot the settling time and overshoot of that region and find the candidate pair of K1 and K3.

```
t = [0:0.05:1];
G123 = G12*G3;

[K1Range K3Range] = meshgrid(35:1:45, 0:0.1:1);
overshootMatrix = [];
settlingTimeMatrix = [];
disturbanceMatrix=[];
candidatePairs = [];
t=[0:0.1:10];
G12= G1*G2;
for K3 = K3Range(:,1)'
    overshootArr=[];
    settlingTimeArr=[];
```

```
    disturbanceArr=[];
    for K1 = K1Range(1,:)
        CLTF = ((K1+K3*s)*G123)/(1+(K1+K3*s)*G123);
        y = step(CLTF, t);
        info = stepinfo(y, t, 'SettlingTimeThreshold', 0.02);
        overshootArr = [overshootArr, info.Overshoot];
        settlingTimeArr = [settlingTimeArr, info.SettlingTime];

        if(info.Overshoot <= 5 & info.SettlingTime <0.25)
            candidatePairs = [candidatePairs; K1, K3];
        end
    end
    overshootMatrix = [overshootMatrix; overshootArr];
    settlingTimeMatrix = [settlingTimeMatrix; settlingTimeArr];
end
```

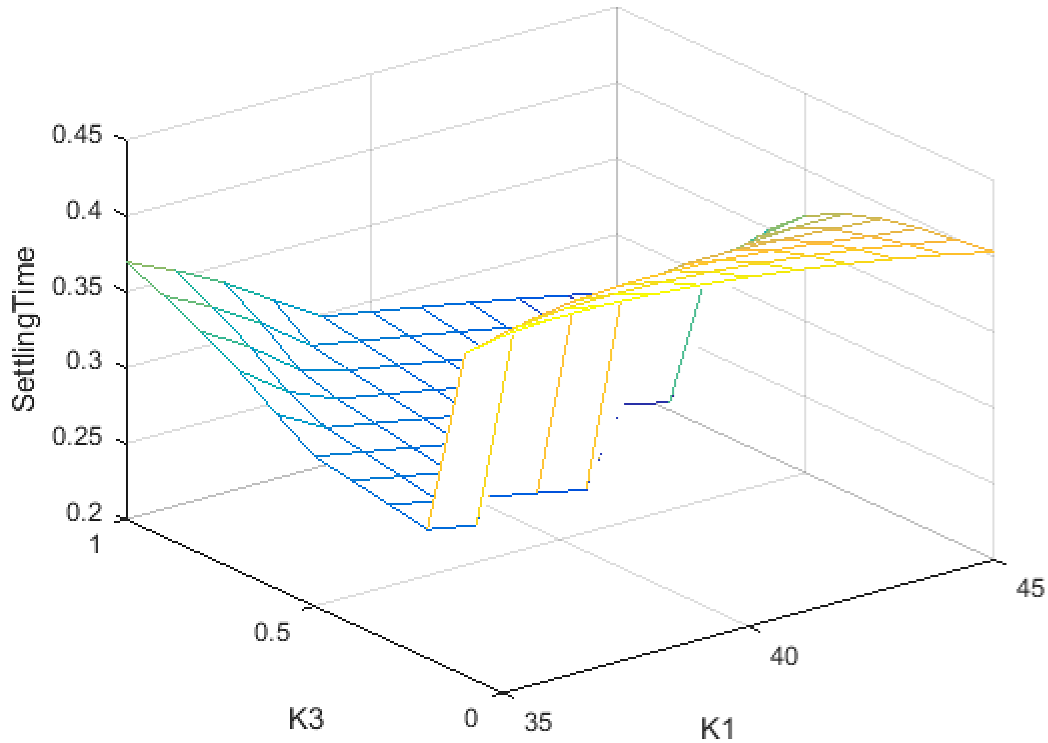
The overshoot plot around K1=40 and K2=1

```
figure();
mesh(K1Range, K3Range,
 overshootMatrix);xlabel('K1');ylabel('K3');zlabel('Overshoot')
```




The settling time plot

```
figure();
mesh(K1Range, K3Range,
 settlingTimeMatrix);xlabel('K1');ylabel('K3');zlabel('SettlingTime')
```



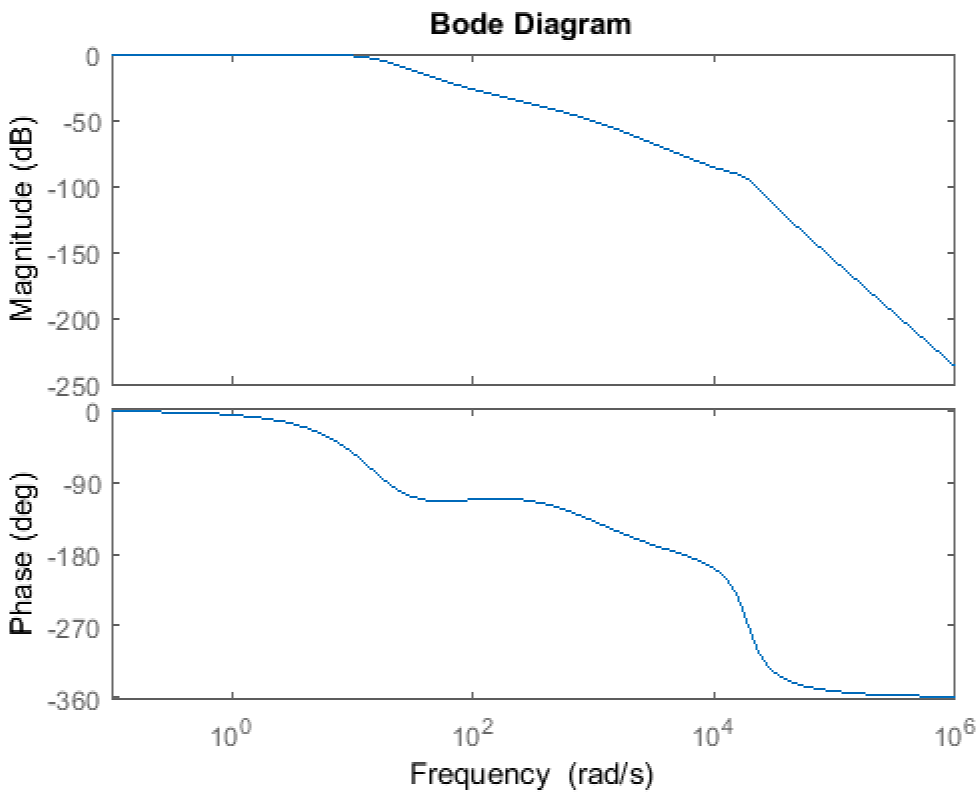

We were able to find some candidates pair of K1 and K3

```
candidatePairs
```

```
candidatePairs =

   45.0000    0.8000
```

So lucky that we found one. We can take K1 = 45 and K3 = 0.8 and draw the bode plot

```
K1 = 45;
K3 = 0.8;
t=[0:0.01:1];
CLTF = ((K1+K3*s)*G123)/(1+(K1+K3*s)*G123);
figure();
bode(CLTF);
```



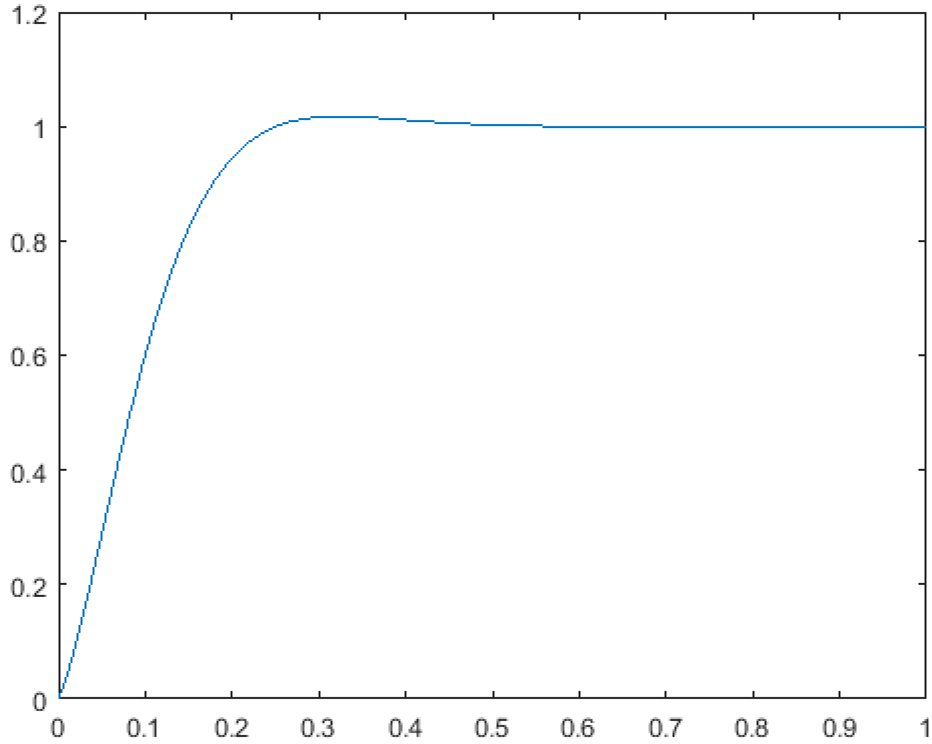

and we can get the -3dB frequency

```
bandWidth = bandwidth(CLTF)

% The system fulfils the transient requirements
info = stepinfo(y, t, 'SettlingTimeThreshold', 0.02)
```

```
bandWidth =

   13.6610


info =

        RiseTime: 0.0173
    SettlingTime: 0.0262
     SettlingMin: 0.9356
     SettlingMax: 1.0082
       Overshoot: 0.8169
      Undershoot: 0
            Peak: 1.0082
        PeakTime: 0.0400
```

We can also plot the unit step response

```
y = step(CLTF,t);
figure();
plot(t,y);
```




# Task 6C

It is easy to get Gain margin and phase margin, just with a function call

```
[GainMargin, PhaseMargin, Wgm, Wpm] = margin(CLTF);
GainMargin
PhaseMargin
```

```
Warning: The closed-loop system is unstable.

GainMargin =

   6.6390e+03


PhaseMargin =

  -180
```

*Published with MATLAB® R2015a*