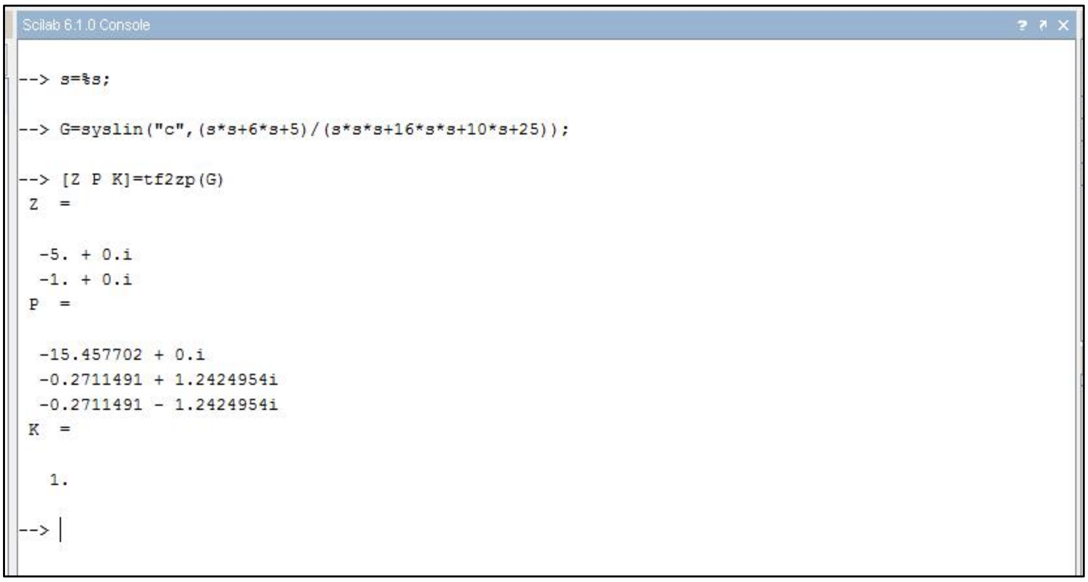
Simulation Assignment

Name :-BAJANIA DARSHAN KUMAR
Enroll. No.:-180110120001

The platform used for to complete this assignment is Scilab.

Sol.1

```
s=%s;
//To create transfer function
G=syslin(″c″,(s*s+6*s+5)/(s*s*s+16*s*s+10*s+25));

[Z P K]=tf2zp(G)//to extract zeroes ,poles and gain
```



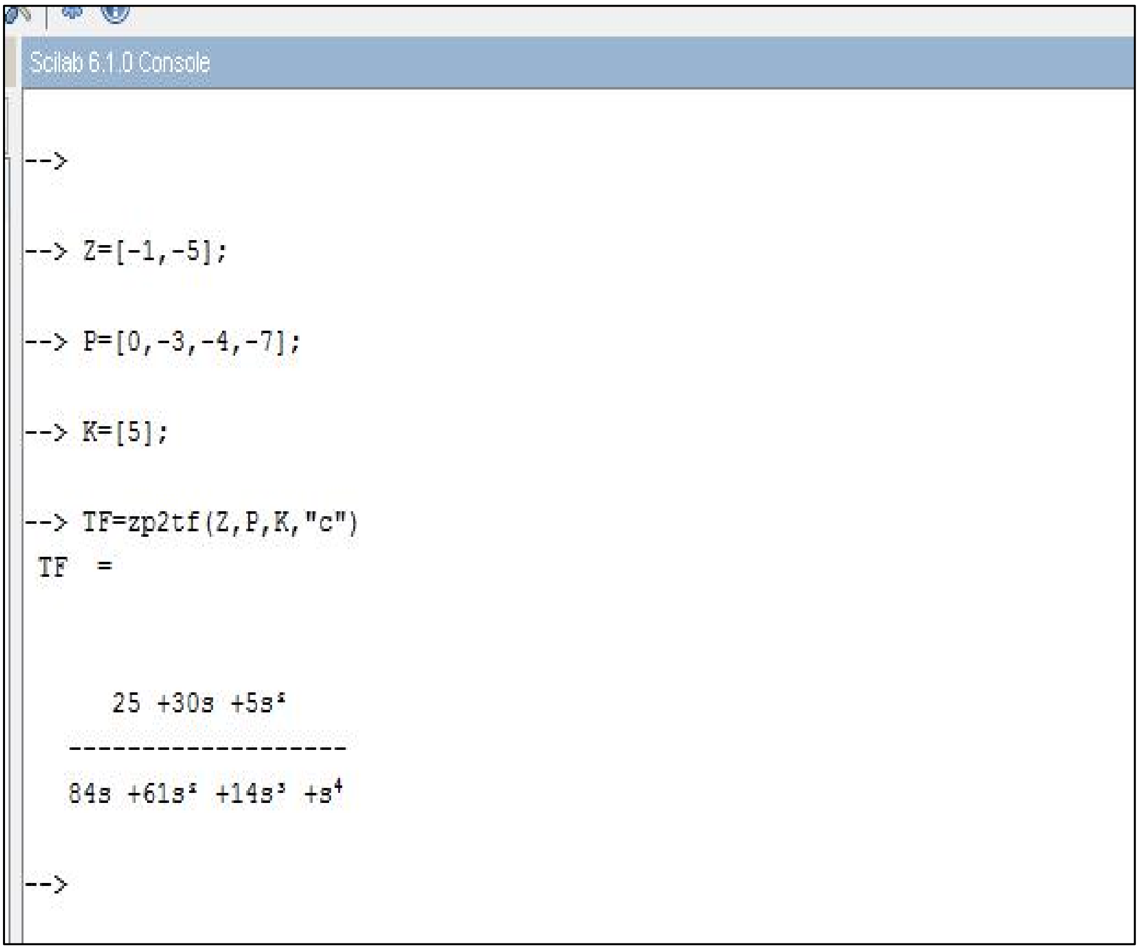
Scilab 6.1.0 Console

```
--> s=%s;

--> G=syslin("c",(s*s+6*s+5)/(s*s*s+16*s*s+10*s+25));

--> [Z P K]=tf2zp(G)
 Z  =

  -5. + 0.i
  -1. + 0.i
 P  =

  -15.457702 + 0.i
  -0.2711491 + 1.2424954i
  -0.2711491 - 1.2424954i
 K  =

   1.

--> |
```

Values of zeros , poles and gain

Sol. 2

```
Z=[-1,-5];
P=[0,-3,-4,-7];
K=[5];
TF=zp2tf(Z,P,K,” c”)//to make T.F. out from zeros,poles,& gain
```

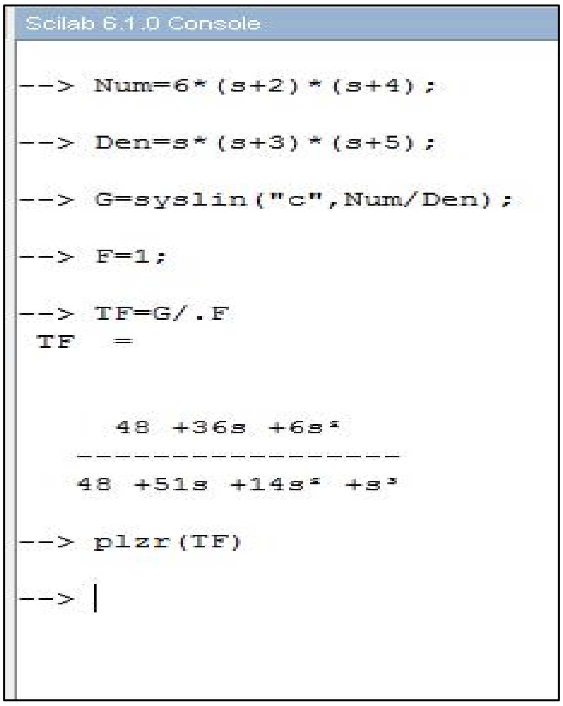


```
-->

--> Z=[-1,-5];

--> P=[0,-3,-4,-7];

--> K=[5];

--> TF=zp2tf(Z,P,K,"c")
 TF  =


     25 +30s +5s²
   -------------------
   84s +61s² +14s³ +s⁴

-->
```

Transfer function from zeros poles and gain

Sol. 3

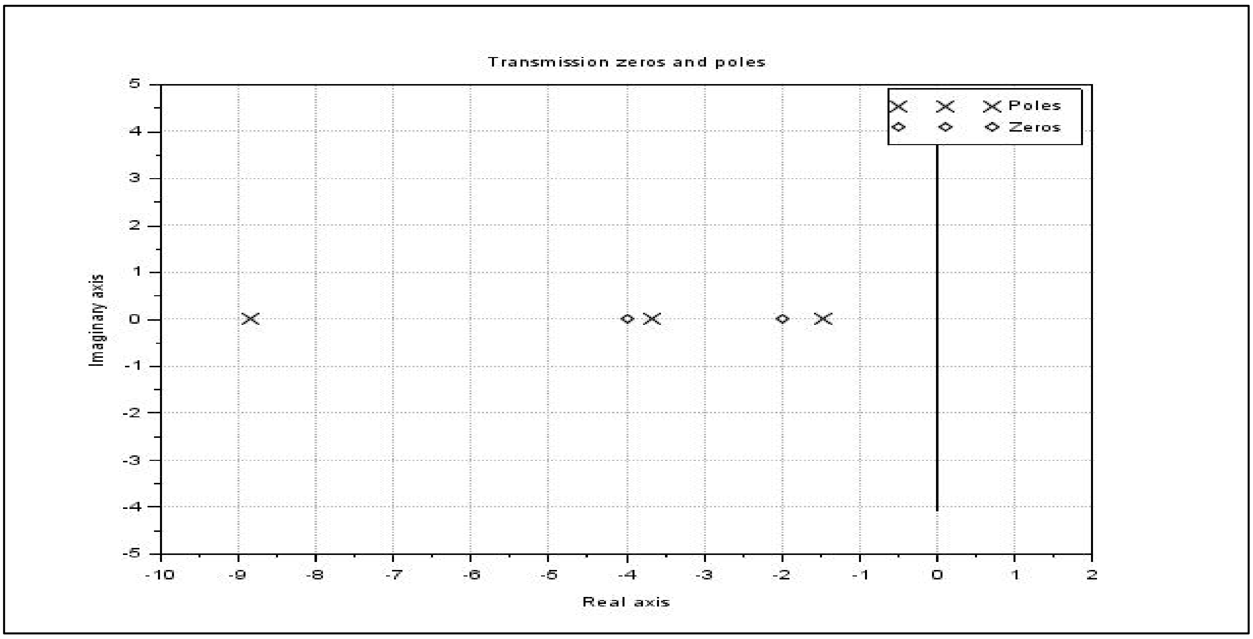
```
Num=6*(s+2)*(s+4);
Den=s*(s+3)*(s+5);
G=syslin( “c” ,Num/Den);//to create transfer function
F=1;
TF=G/.F;       //to create closed loop T.F.
plzr(TF)      //to plot poles and zeros
```



```
--> Num=6*(s+2)*(s+4);

--> Den=s*(s+3)*(s+5);

--> G=syslin("c",Num/Den);

--> F=1;

--> TF=G/.F
 TF  =


    48 +36s +6s²
  ----------------
   48 +51s +14s² +s³

--> plzr(TF)

--> |
```

<-- Code for TF and feedback

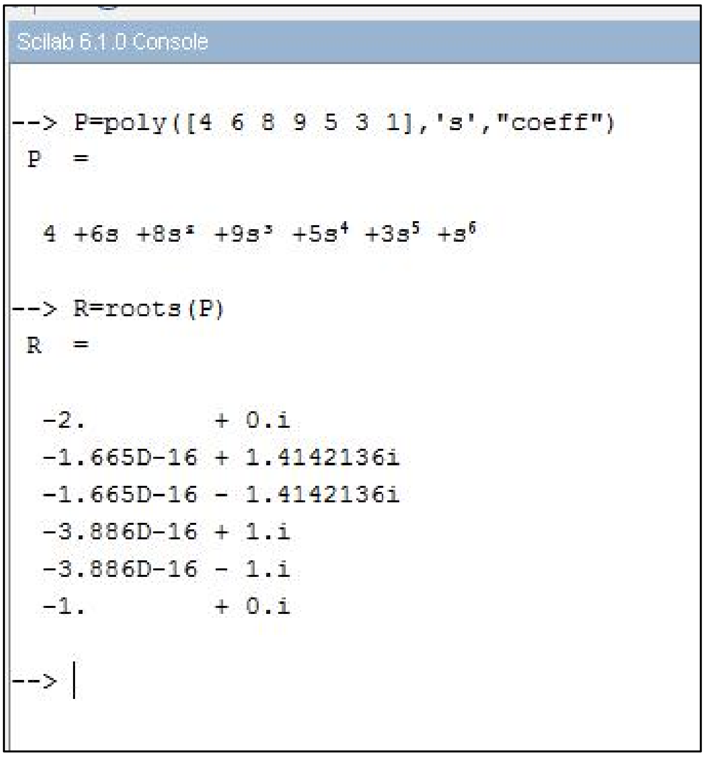
Plot for zeros and poles



SOL.4

(i)

```
P=poly((4 6 8 9 5 3 1),’ s’ ,” coeff” );//to create a polynomial
R=roots(P);//to find the roots
```

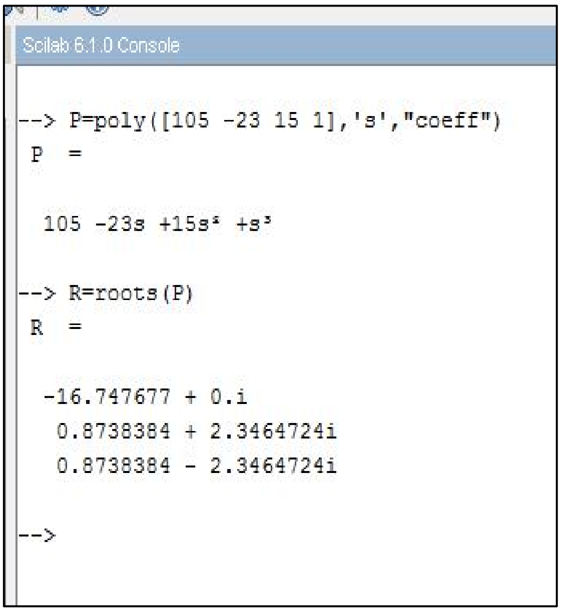

Scilab 6.1.0 Console

```
--> P=poly([4  6  8  9  5  3  1],'s',"coeff")
 P  =

  4 +6s +8s² +9s³ +5s⁴ +3s⁵ +s⁶

--> R=roots(P)
 R  =

  -2.         + 0.i
  -1.665D-16 + 1.4142136i
  -1.665D-16 - 1.4142136i
  -3.886D-16 + 1.i
  -3.886D-16 - 1.i
  -1.         + 0.i

-->
```

Roots of the polynomial

(ii)

```
P=poly((105 -23 15 1),’ s’ ,” coeff” );
R=roots(P);
```


Scilab 6.1.0 Console

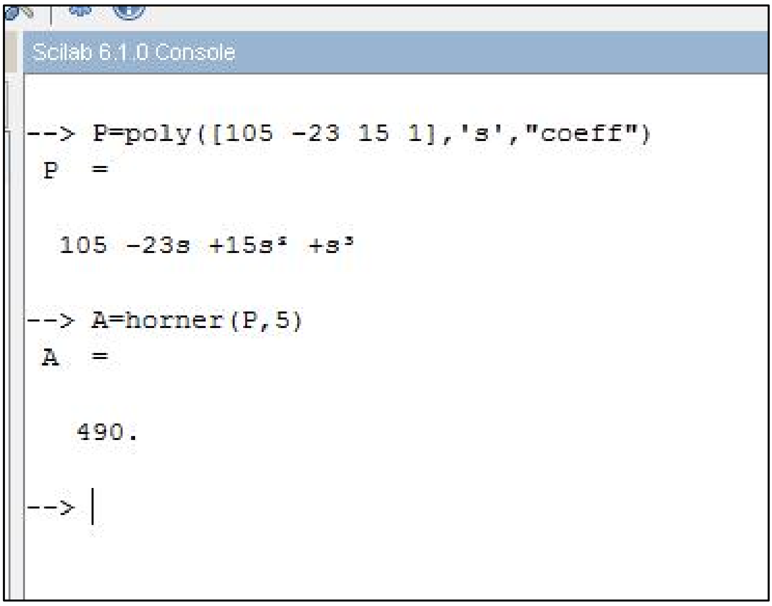
```
--> P=poly([105 -23 15 1],'s',"coeff")
 P  =

  105 -23s +15s² +s³

--> R=roots(P)
 R  =

  -16.747677 + 0.i
    0.8738384 + 2.3464724i
    0.8738384 - 2.3464724i

-->
```

Roots of the polynomial

Sol 5

(i)

```
P=poly([7 -8 7 -3 2 1],’ s’ ,” coeff”);//to create a polynomial
A=horner(P,5)   //to find the value of poly. at respective value
```

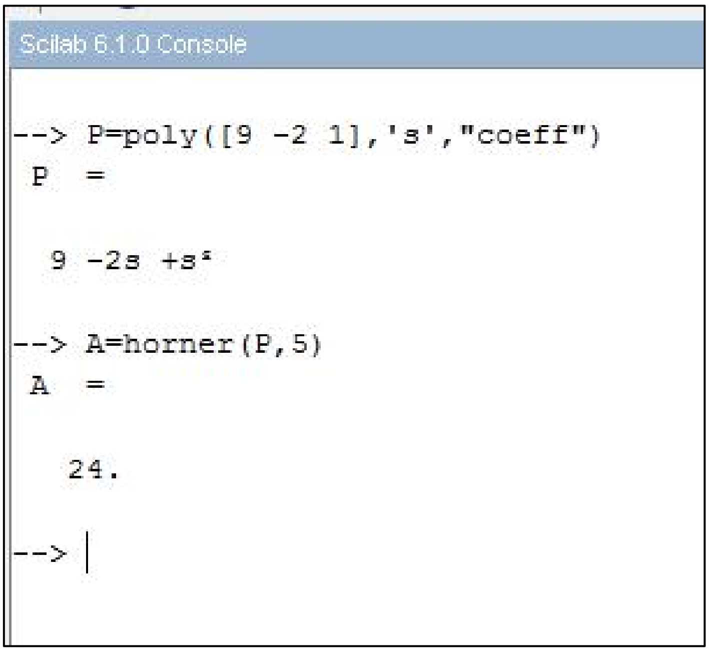


```
--> P=poly([105 -23 15 1],'s',"coeff")
 P  =

  105 -23s +15s² +s³

--> A=horner(P,5)
 A  =

   490.

-->
```

value of the polynomial

(ii)

```
P=poly([9 -2 1],’ s’ ,” coeff”);
A=horner(P,5)
```

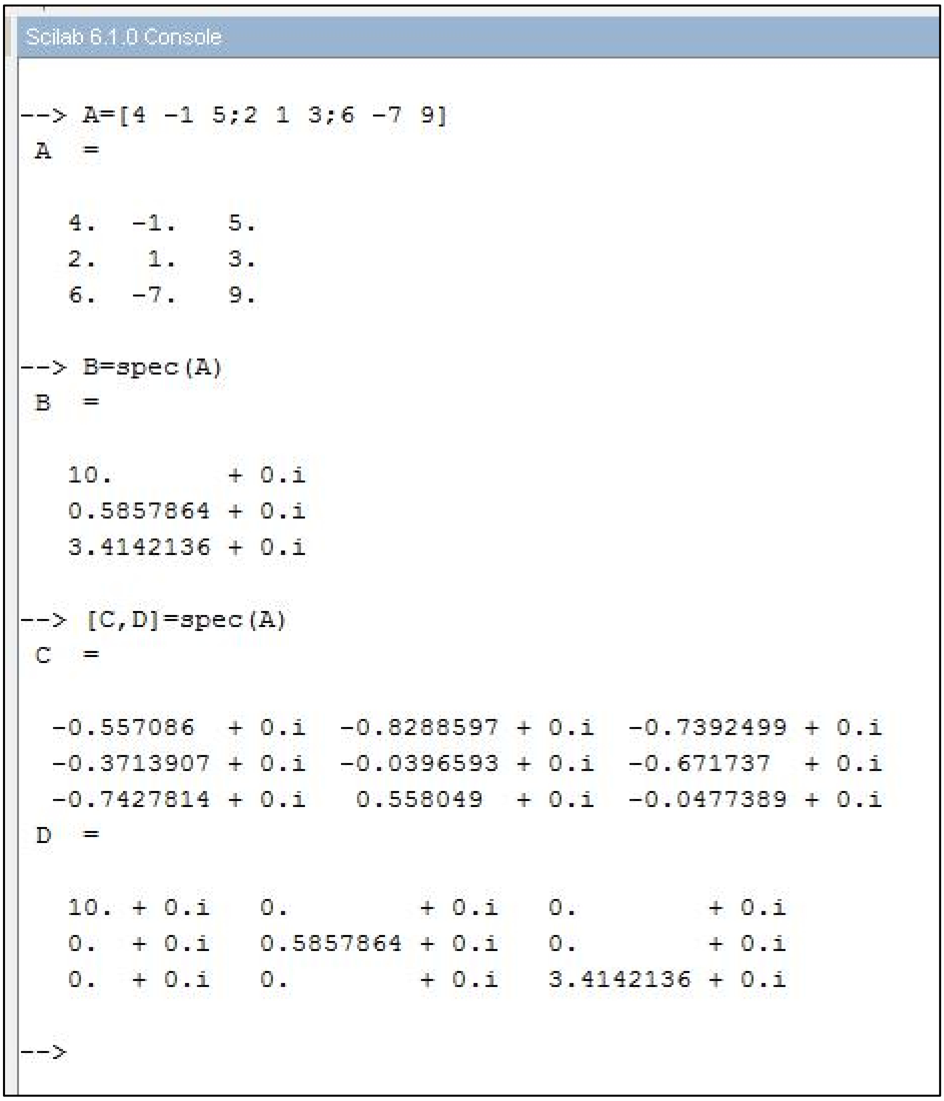


```
--> P=poly([9 -2 1],'s',"coeff")
 P  =

  9 -2s +s²

--> A=horner(P,5)
 A  =

   24.

-->
```

value of the polynomial

Sol 6

```
A=[4 -1 5;2 1 3;6 -7 9];
B=spec(A); //for eigenvalues
[C,D]=spec(A)  // to extract eigenvectors
```



```
--> A=[4 -1 5;2 1 3;6 -7 9]
 A  =

   4.  -1.   5.
   2.   1.   3.
   6.  -7.   9.

--> B=spec(A)
 B  =

   10.       + 0.i
   0.5857864 + 0.i
   3.4142136 + 0.i

--> [C,D]=spec(A)
 C  =

  -0.557086  + 0.i  -0.8288597 + 0.i  -0.7392499 + 0.i
  -0.3713907 + 0.i  -0.0396593 + 0.i  -0.671737  + 0.i
  -0.7427814 + 0.i   0.558049  + 0.i  -0.0477389 + 0.i
 D  =

   10. + 0.i   0.        + 0.i   0.        + 0.i
   0.  + 0.i   0.5857864 + 0.i   0.        + 0.i
   0.  + 0.i   0.        + 0.i   3.4142136 + 0.i

-->
```

Values of B are Eigenvalues and values of C are Eigenvectors

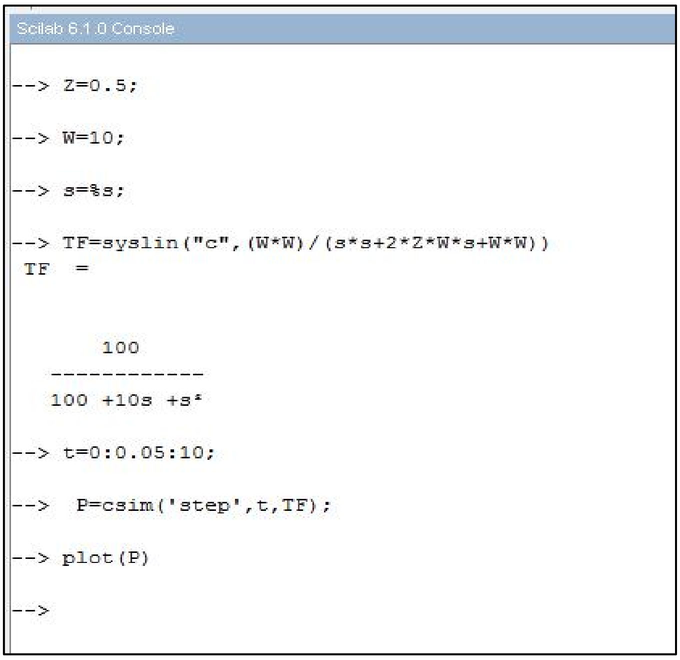
Sol 7

Given,

z=0.5 and w=10rad/sec

Therefore,

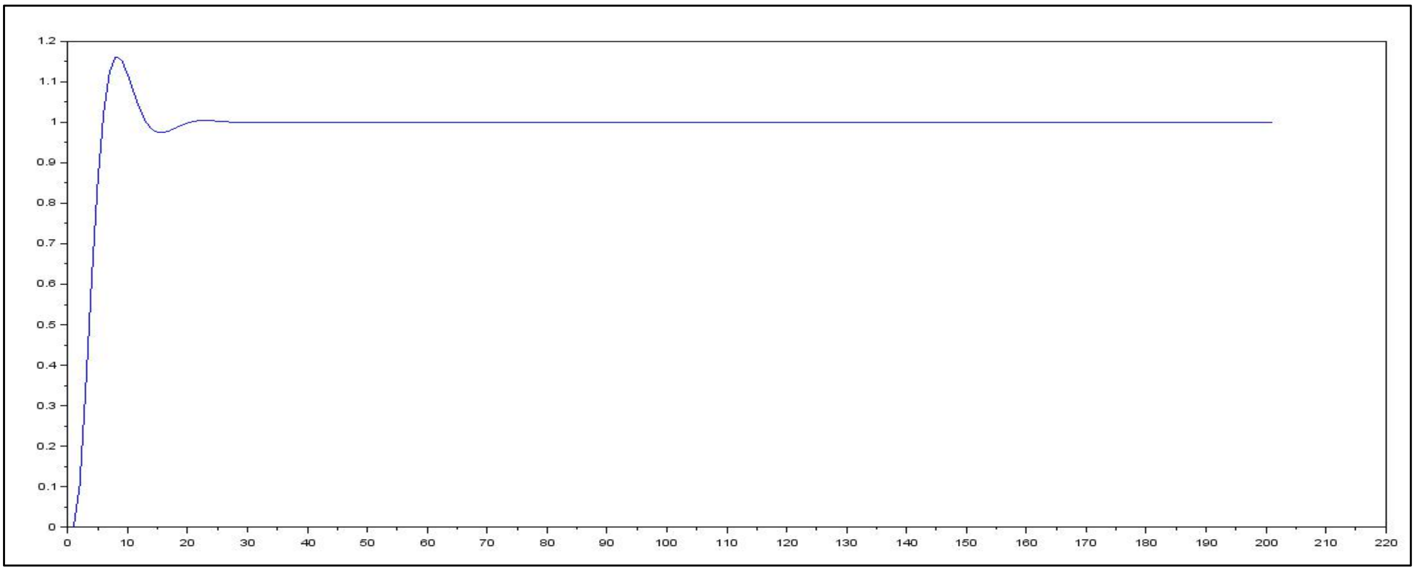
```
Z=0.5;//Zeta
W=10;//Natural frequecy
s=%s;
TF=syslin((W*W)/(s*s+2*Z*W*s+W*W));//to create T.F.
t=0:0.05:10;          //an array of nos.from 0-10
P=csim( ‘step’ ,t,TF);    //to get the step response
Plot2d(P)//to display the plot
```



```
--> Z=0.5;

--> W=10;

--> s=%s;

--> TF=syslin("c",(W*W)/(s*s+2*Z*W*s+W*W))
 TF  =


      100
   ------------
   100 +10s +s²

--> t=0:0.05:10;

-->  P=csim('step',t,TF);

--> plot(P)

-->
```

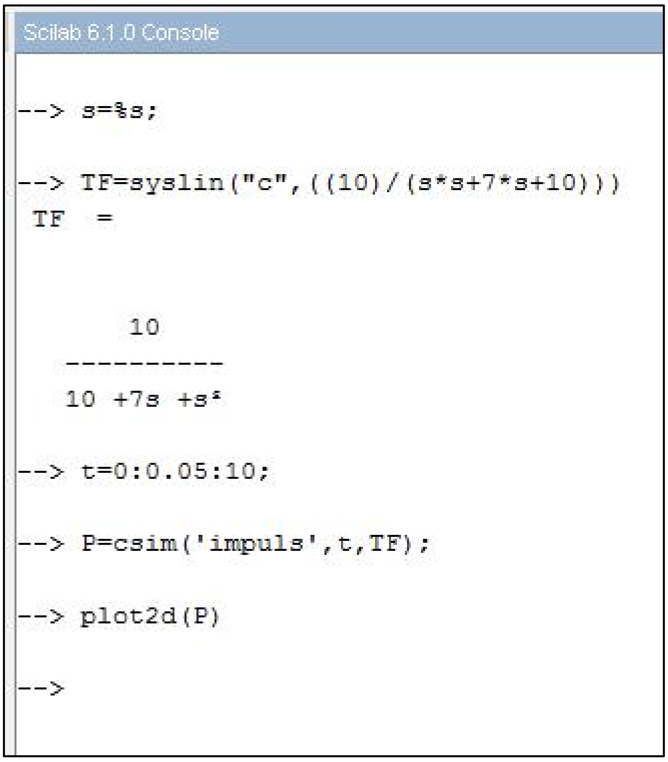
<--code for plotting step response

Step response



Sol 8

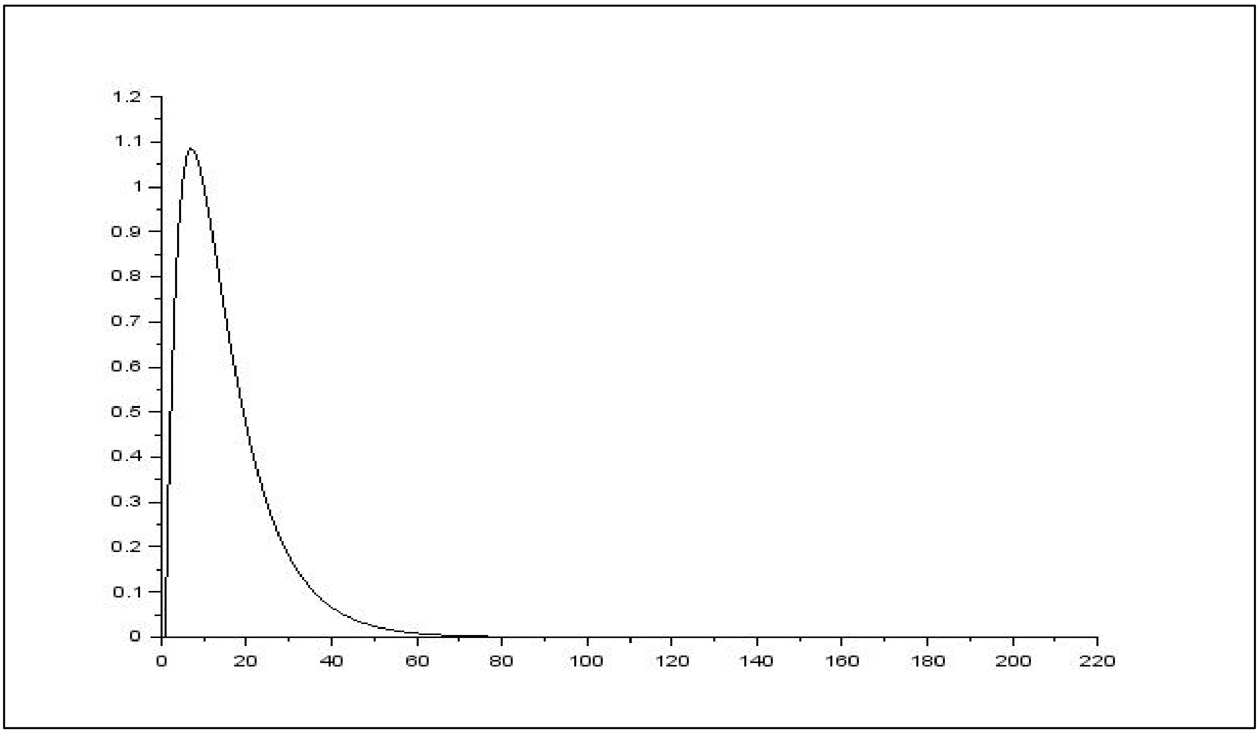
```
s=%s;
TF=syslin( “c” ,(10/(s*s+7*s+10));//to create T.F.
t=0:0.05:10;      //an array of nos. From 0-10
P=csim( ‘impuls’ ,t,TF);  to get unit impulse response
plot2d(P)  //to display the plot
```



```
--> s=%s;

--> TF=syslin("c",((10)/(s*s+7*s+10)))
 TF  =


       10
   ----------
   10 +7s +s²

--> t=0:0.05:10;

--> P=csim('impuls',t,TF);

--> plot2d(P)

-->
```

Code for ploting impulse response

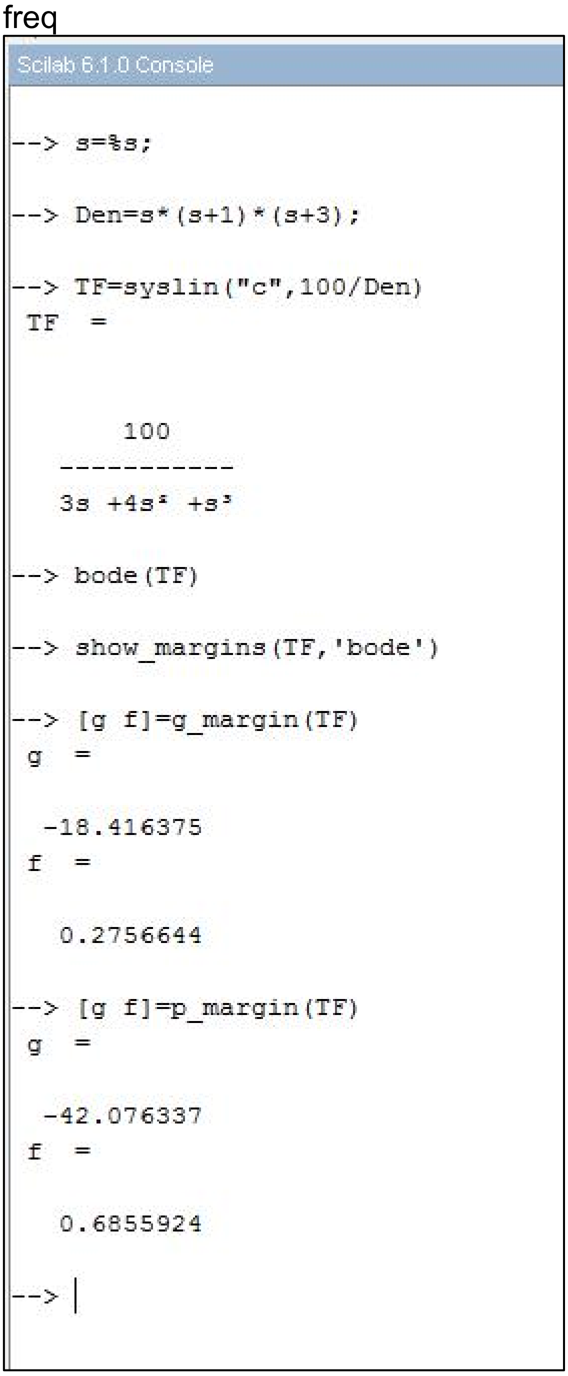
Impulse response



Sol 9

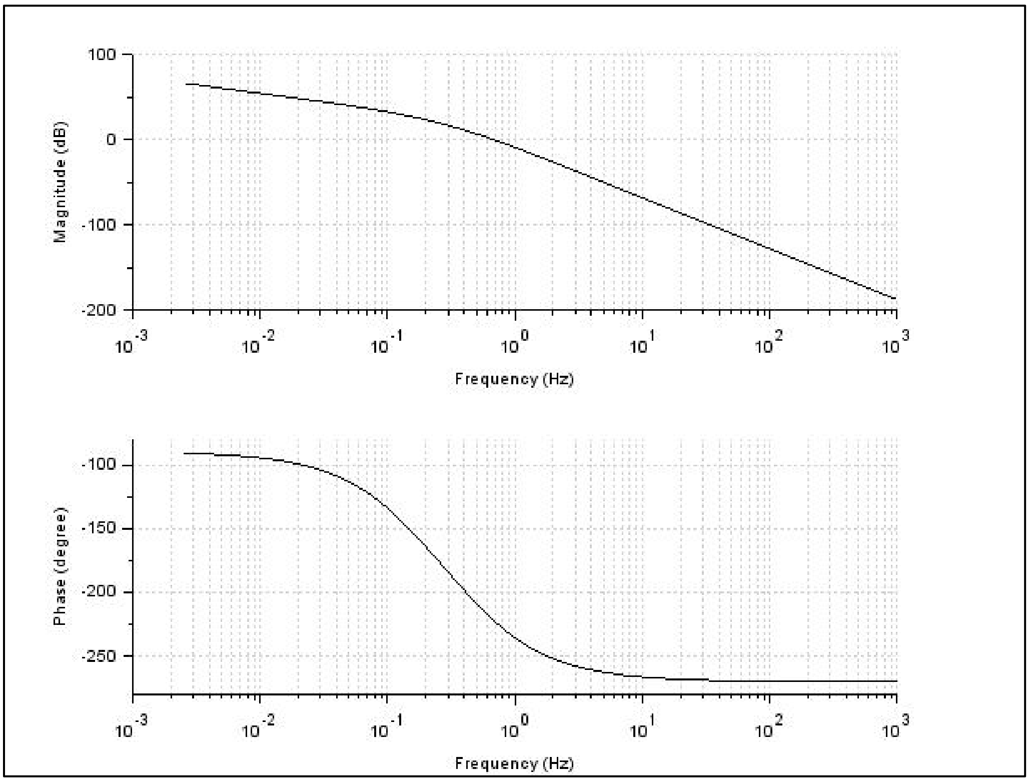
```
s=%s;
Den=s*(s+2)*(s+1)
TF=syslin( “c” ,(100)/Den)//to form T.f.
bode(TF)  //to display bode plot
show_margins(TF,’ bode’ ) //to show gain & phase margins
[g f]=g_margin(TF) //to find gain value & phase crossover freq.
[g f]=p_margin(TF)//to find phase value & phase crossover freq.
```

code to display bode plot and find gain,phase,phase over freq. and gain over freq



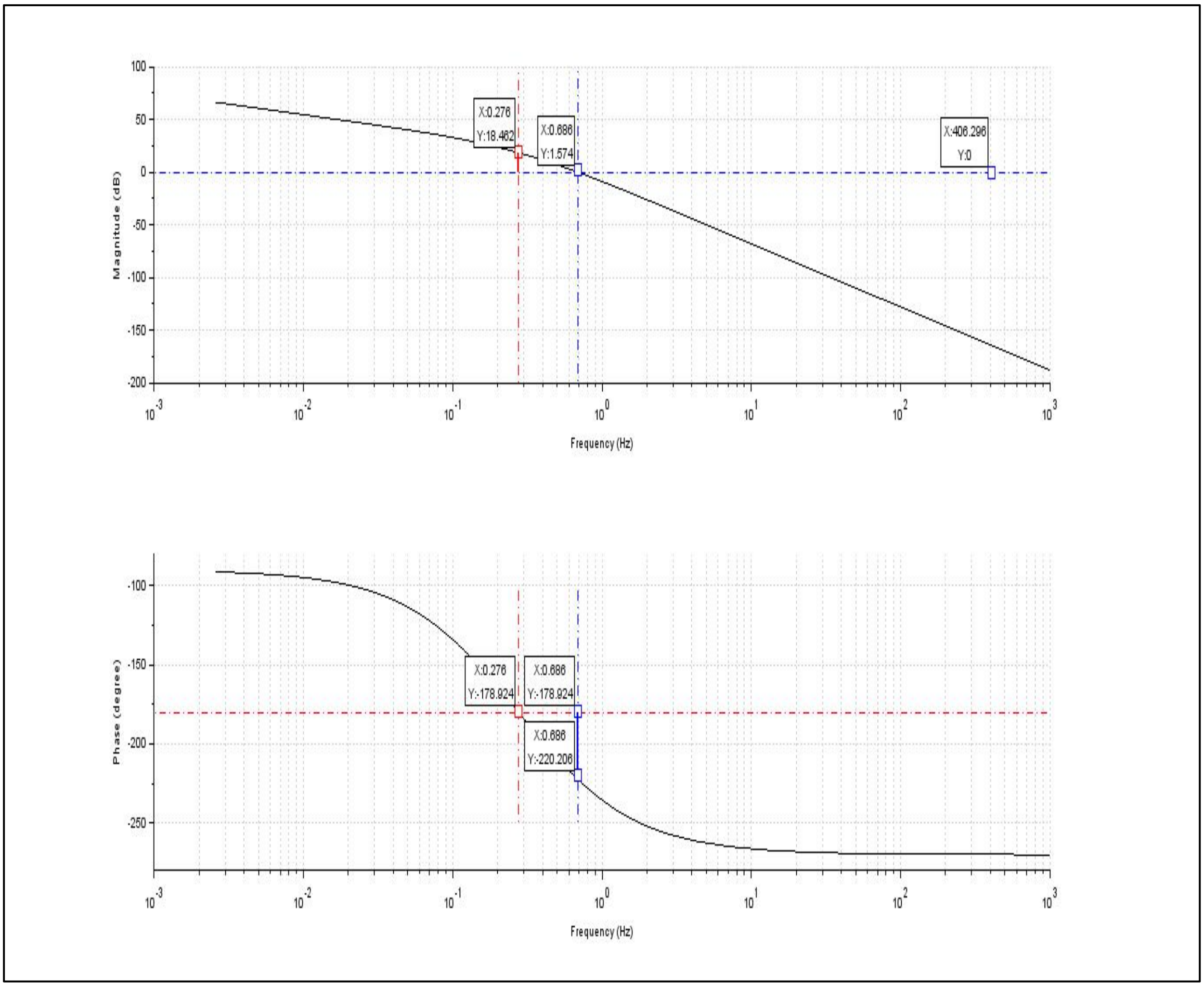
```
--> s=%s;

--> Den=s*(s+1)*(s+3);

--> TF=syslin("c",100/Den)
 TF  =


        100
   -----------
   3s +4s² +s³

--> bode(TF)

--> show_margins(TF,'bode')

--> [g f]=g_margin(TF)
 g  =

  -18.416375
 f  =

   0.2756644

--> [g f]=p_margin(TF)
 g  =

  -42.076337
 f  =

   0.6855924

--> |
```

code to display bode plot and



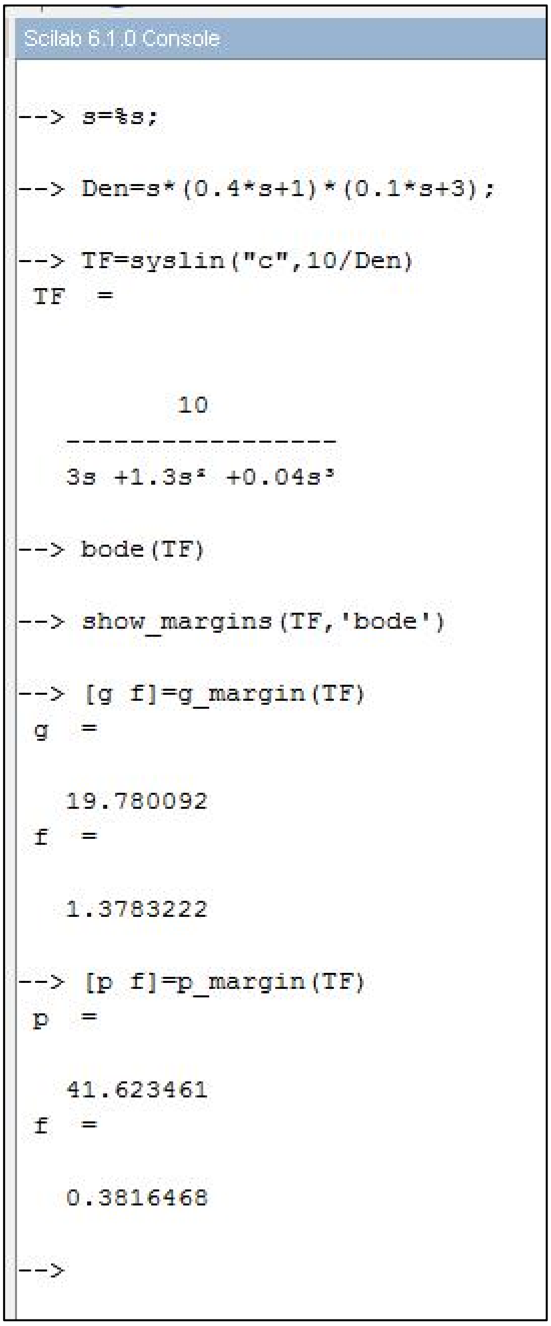
bode plots

Gain margin and phase margin



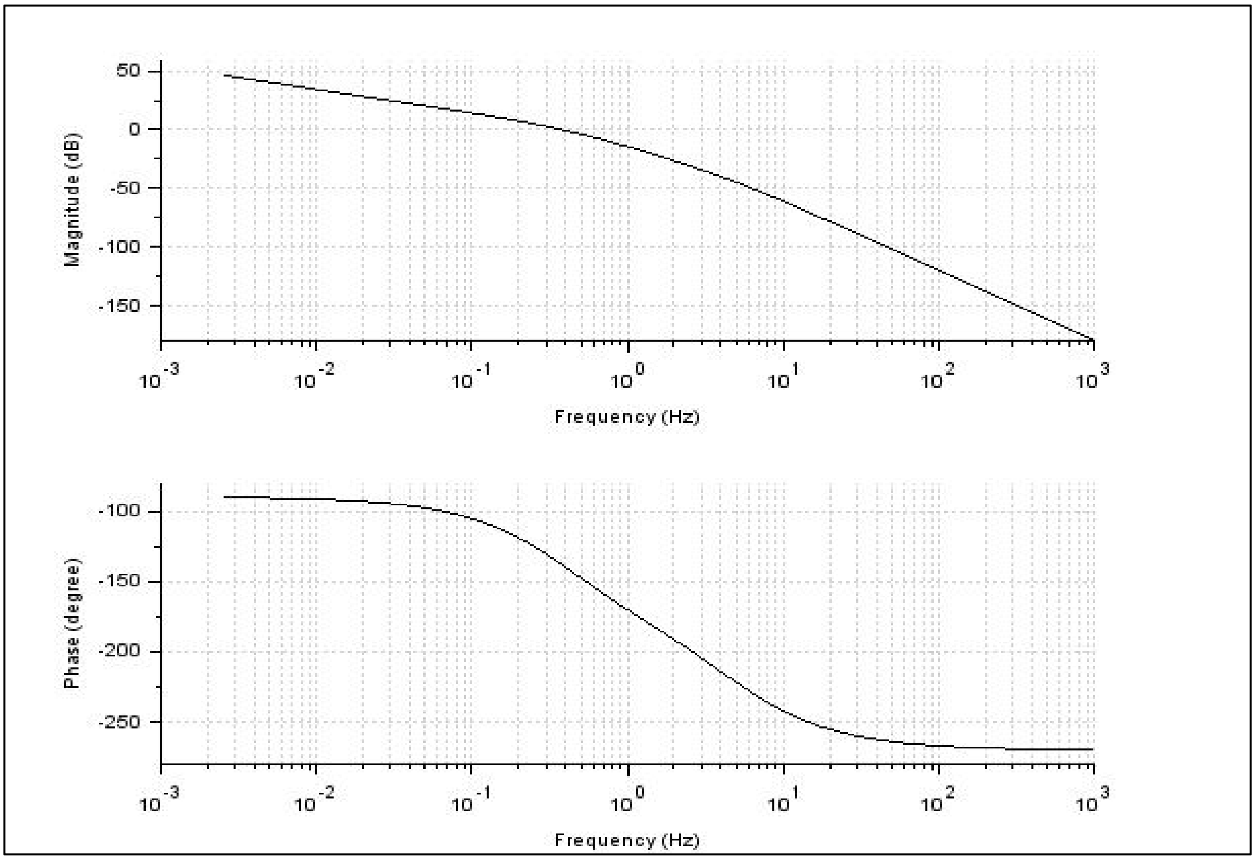
Sol 10

```
s=%s;
Den=s*(1+0.4*s)*(0.1*s+1)
TF=syslin( “c” , (10)/Den)//to form T.F.
bode(TF)//to display bode plot
show_margins(TF,’ bode’ )  //to show gain and phase margins
[g f]=g_margin(TF)//to find gain value & phase crossover freq.
[p f]=p_margin(TF)//to find phase value & gain crossover freq.
```

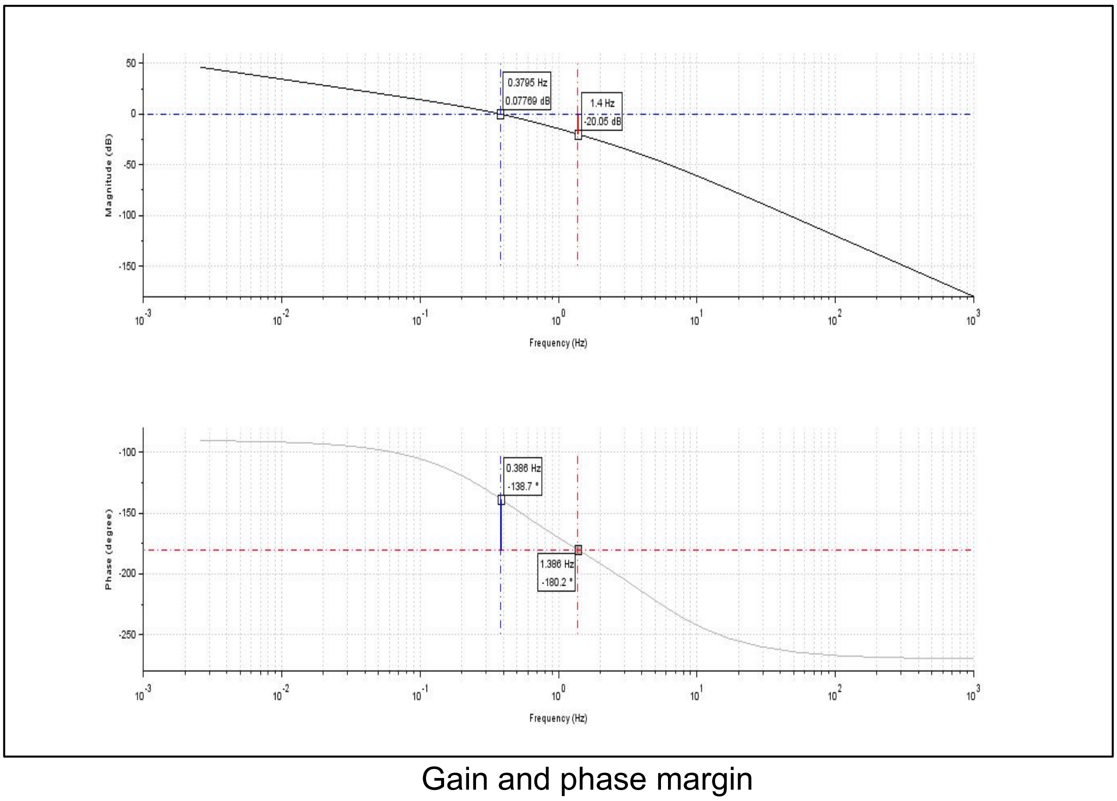


```
--> s=%s;

--> Den=s*(0.4*s+1)*(0.1*s+3);

--> TF=syslin("c",10/Den)
 TF  =


         10
   -----------------
   3s +1.3s² +0.04s³

--> bode(TF)

--> show_margins(TF,'bode')

--> [g f]=g_margin(TF)
 g  =

   19.780092
 f  =

   1.3783222

--> [p f]=p_margin(TF)
 p  =

   41.623461
 f  =

   0.3816468

-->
```

Code for bode plot and to find gain,phase,gain crossover freq. And phase crossover freq.



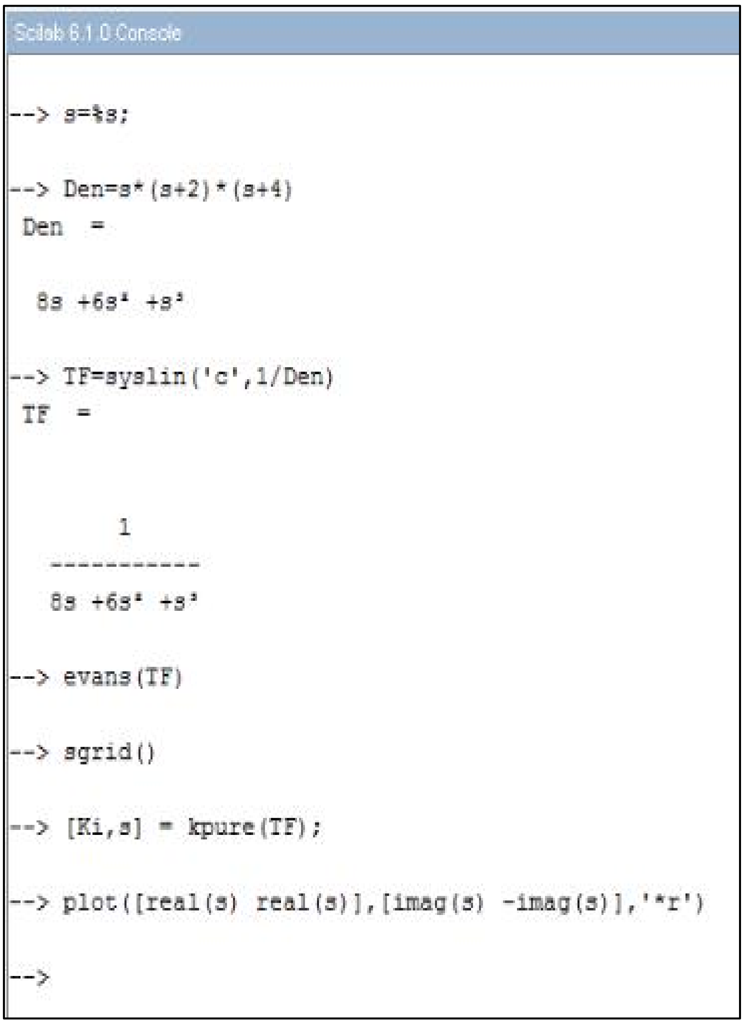
bode plot



Gain and phase margin

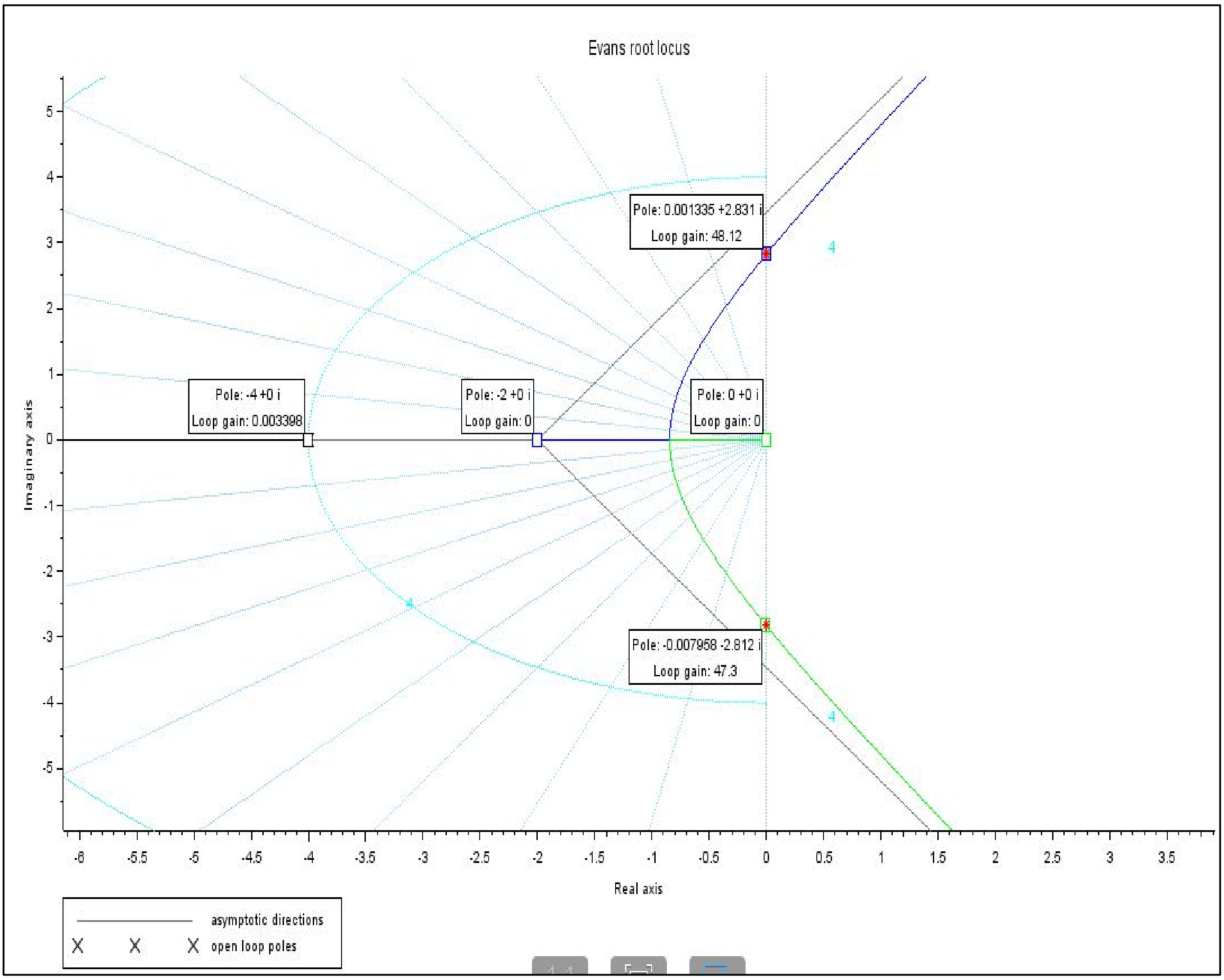
Sol 12

```
s=%s;
Den=s*(s+2)*(s+4);
TF=syslin( ‘c’ ,1/Den);//to form T.F.
evans(TF)      //to create root locus
sgrid();      // to display grid
[Ki,s] = kpure(H) // Gains that give pure imaginary closed loop
//To display intersection point of img. axis and curves
polesplot([real(s) real(s)],[imag(s) -imag(s)],’*r’)
```



```
--> s=%s;

--> Den=s*(s+2)*(s+4)
 Den  =

  8s +6s² +s³

--> TF=syslin('c',1/Den)
 TF  =


       1
   -----------
   8s +6s² +s³

--> evans(TF)

--> sgrid()

--> [Ki,s] = kpure(TF);

--> plot([real(s) real(s)],[imag(s) -imag(s)],'*r')

-->
```

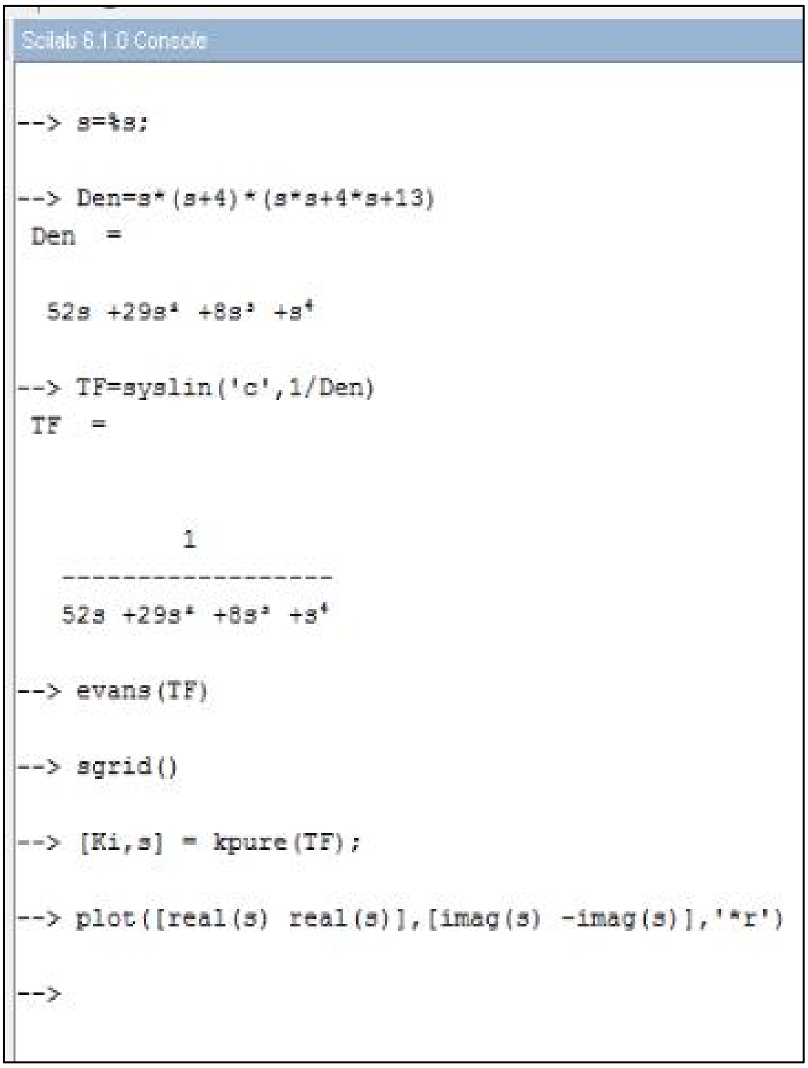
code to plot root locus


Evans root locus
Pole: 0.001335 +2.831 i
Loop gain: 48.12
Pole: -4 +0 i
Loop gain: 0.003398
Pole: -2 +0 i
Loop gain: 0
Pole: 0 +0 i
Loop gain: 0
Pole: -0.007958 -2.812 i
Loop gain: 47.3
4
4
4
Imaginary axis
5
4
3
2
1
0
-1
-2
-3
-4
-5
-6
-5.5
-5
-4.5
-4
-3.5
-3
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
3
3.5
Real axis
asymptotic directions
X X X open loop poles

root locus

Sol 13

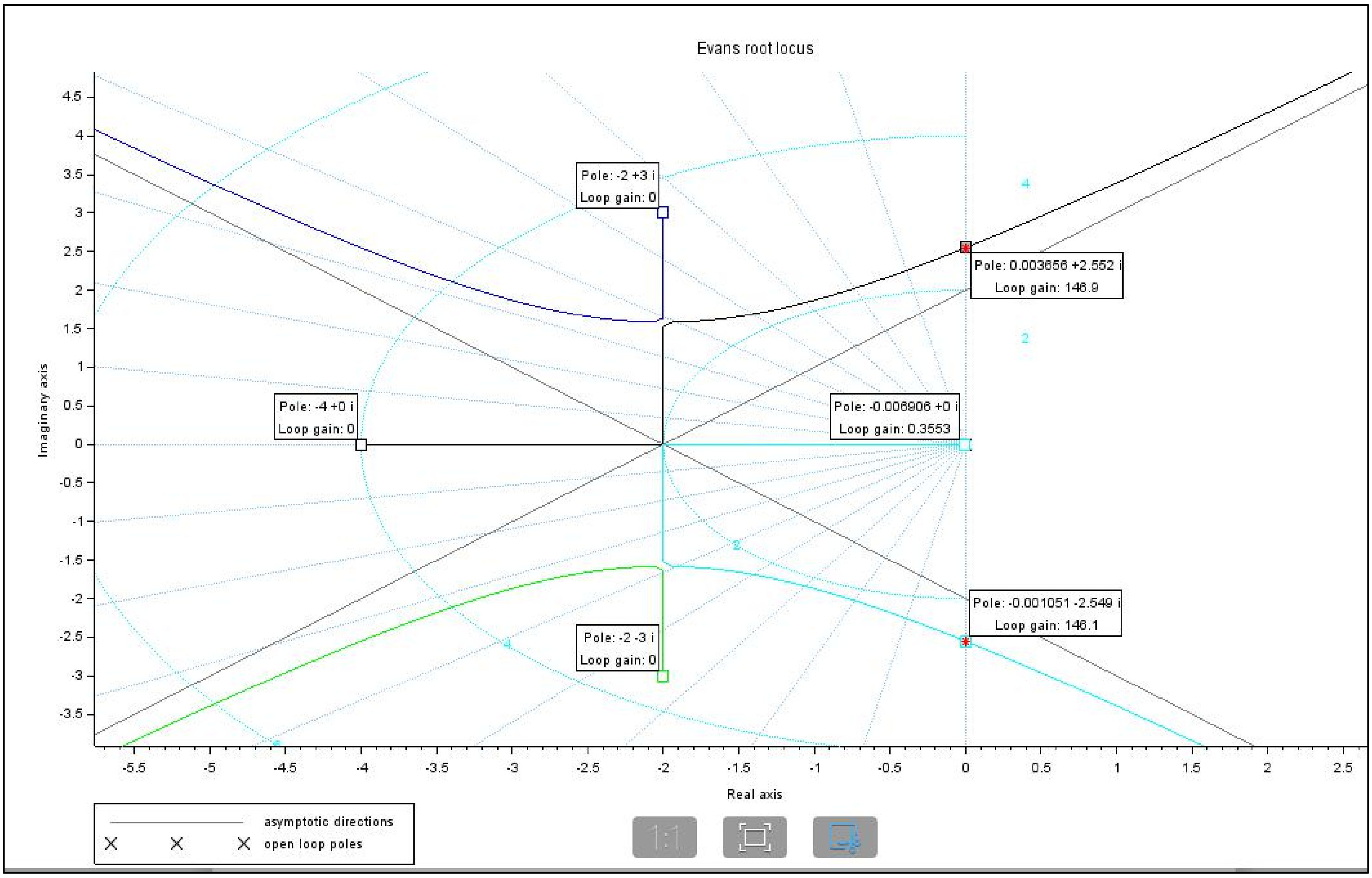
```
s=%s;
Den=s*(s+4)*(s*s+4*s+13);
TF=syslin( ‘c’ ,1/Den);//to form T.F
evans(TF)//to create root locus
sgrid();// to add grid
[Ki,s] = kpure(H) // Gains that give pure imaginary closed loop
//To display intersection point of img. axis and curves
polesplot([real(s) real(s)],[imag(s) -imag(s)],’*r’)
```



```
--> s=%s;

--> Den=s*(s+4)*(s*s+4*s+13)
 Den  =

  52s +29s^2 +8s^3 +s^4

--> TF=syslin('c',1/Den)
 TF  =


           1
   -------------------
   52s +29s^2 +8s^3 +s^4

--> evans(TF)

--> sgrid()

--> [Ki,s] = kpure(TF);

--> plot([real(s) real(s)],[imag(s) -imag(s)],'*r')

-->
```

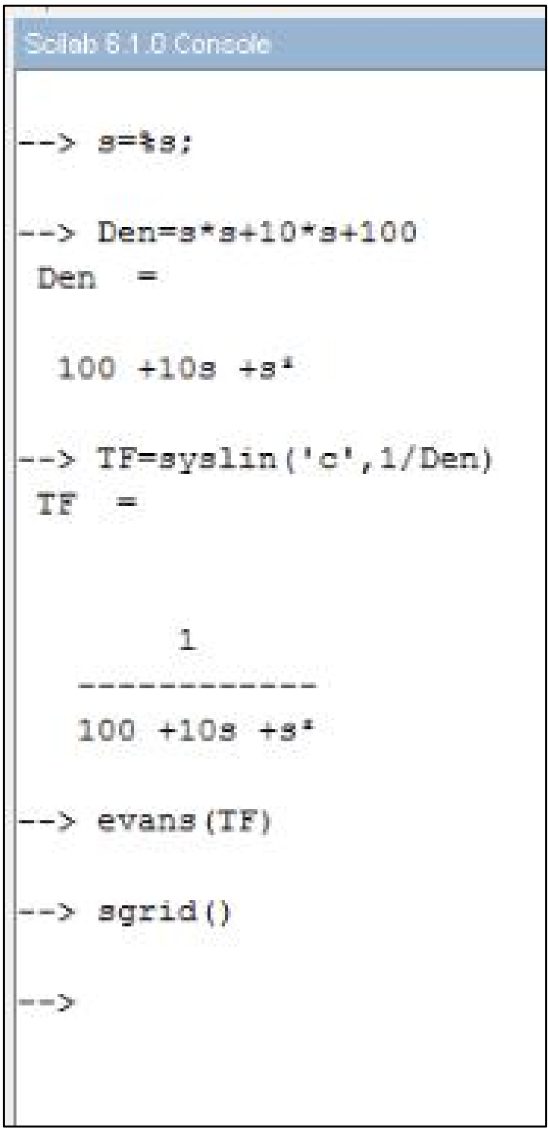
Code to plot root locus

Root locus


Evans root locus
Pole: -2 +3 i
Loop gain: 0
Pole: 0.003656 +2.552 i
Loop gain: 146.9
Pole: -4 +0 i
Loop gain: 0
Pole: -0.006906 +0 i
Loop gain: 0.3553
Pole: -0.001051 -2.549 i
Loop gain: 146.1
Pole: -2 -3 i
Loop gain: 0
4
2
2
4
Imaginary axis
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
-3
-3.5
-5.5
-5
-4.5
-4
-3.5
-3
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
Real axis
asymptotic directions
× × × open loop poles
1:1

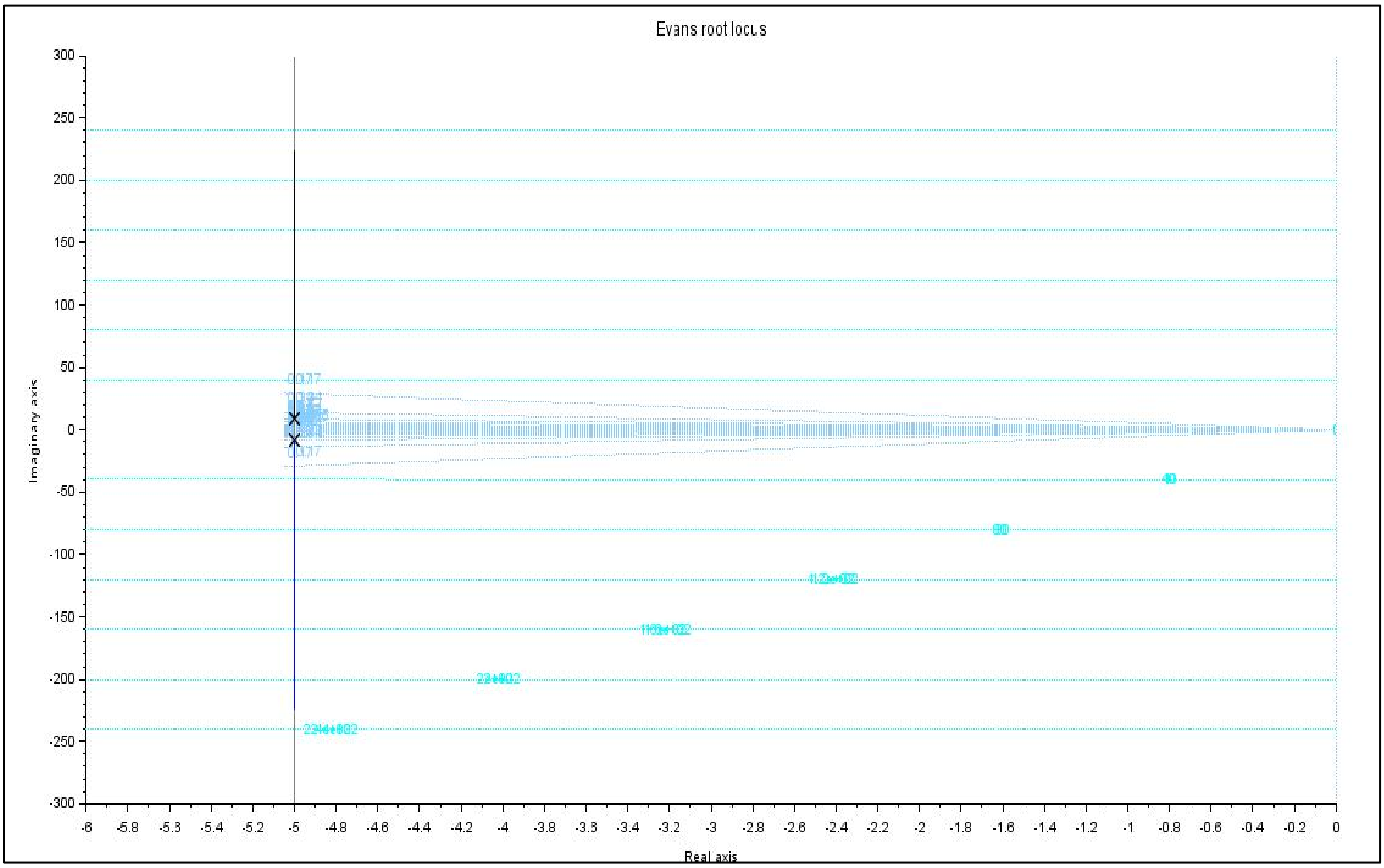
Sol 14

```
s=%s;
Den=(s*s+10*s+100);
TF=syslin( ‘c’ ,1/Den);//to create transfer function
evans(TF)//to create root locus
sgrid();to add grid
```

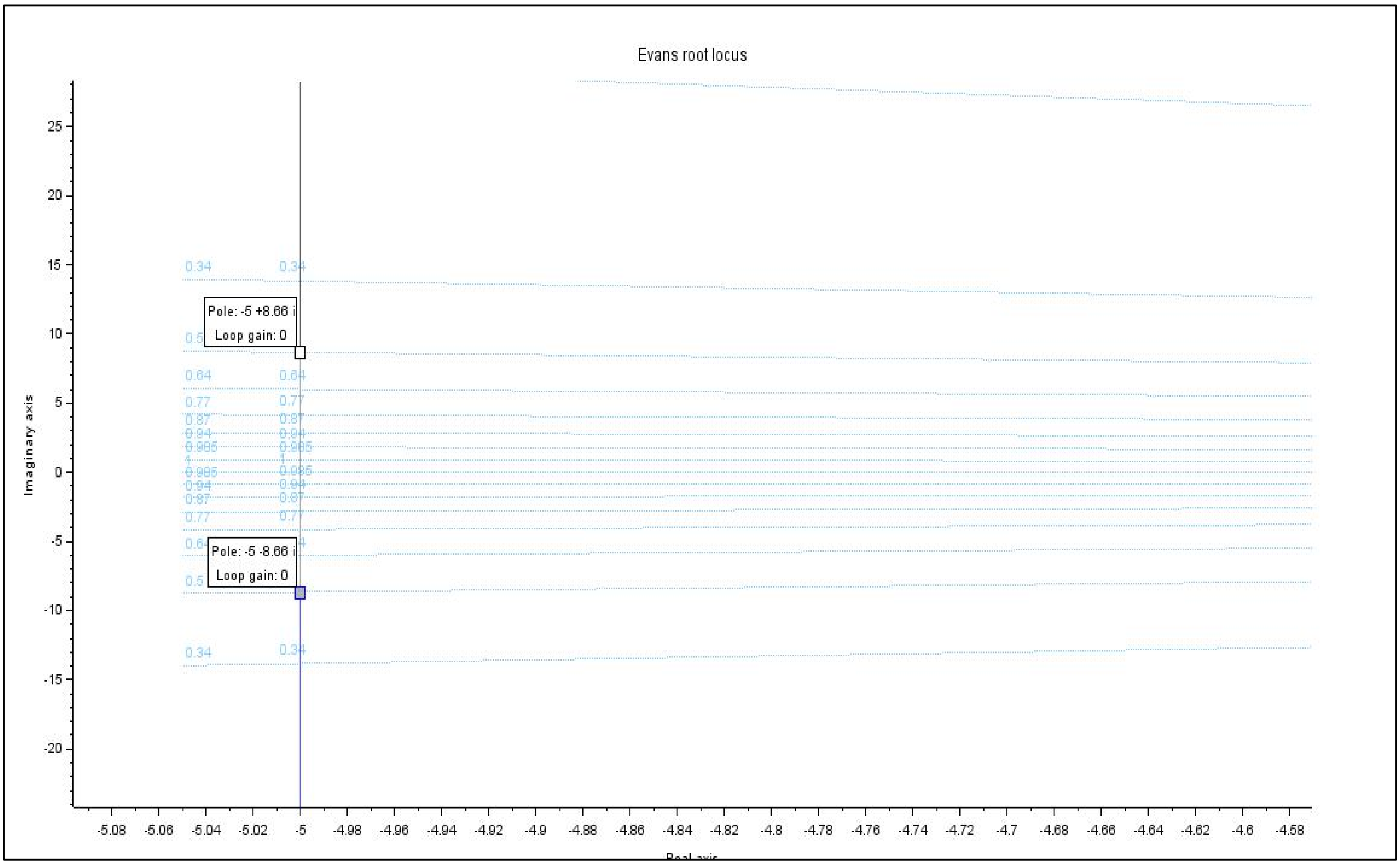


```
--> s=%s;

--> Den=s*s+10*s+100
 Den  =

  100 +10s +s²

--> TF=syslin('c',1/Den)
 TF  =


        1
   ------------
   100 +10s +s²

--> evans(TF)

--> sgrid()

-->
```

code to plot root locus



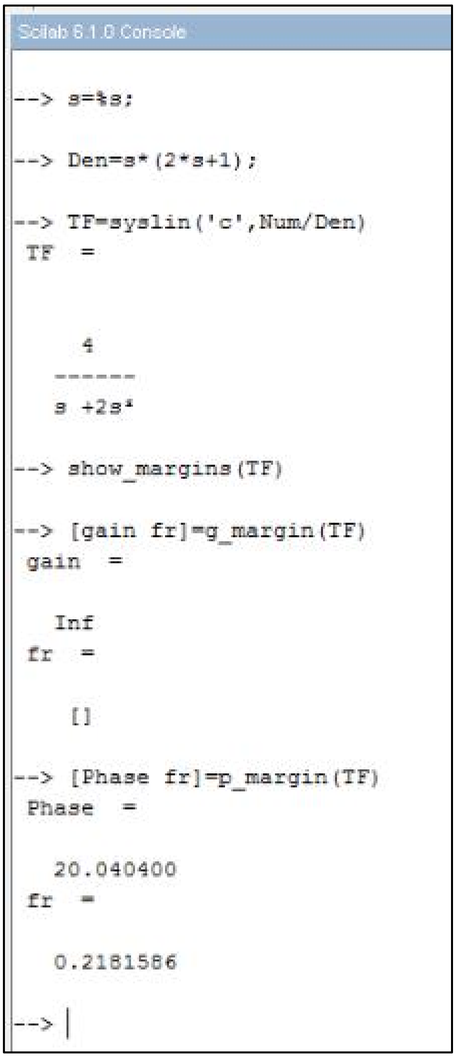
Root locus



Root locus

Sol 15

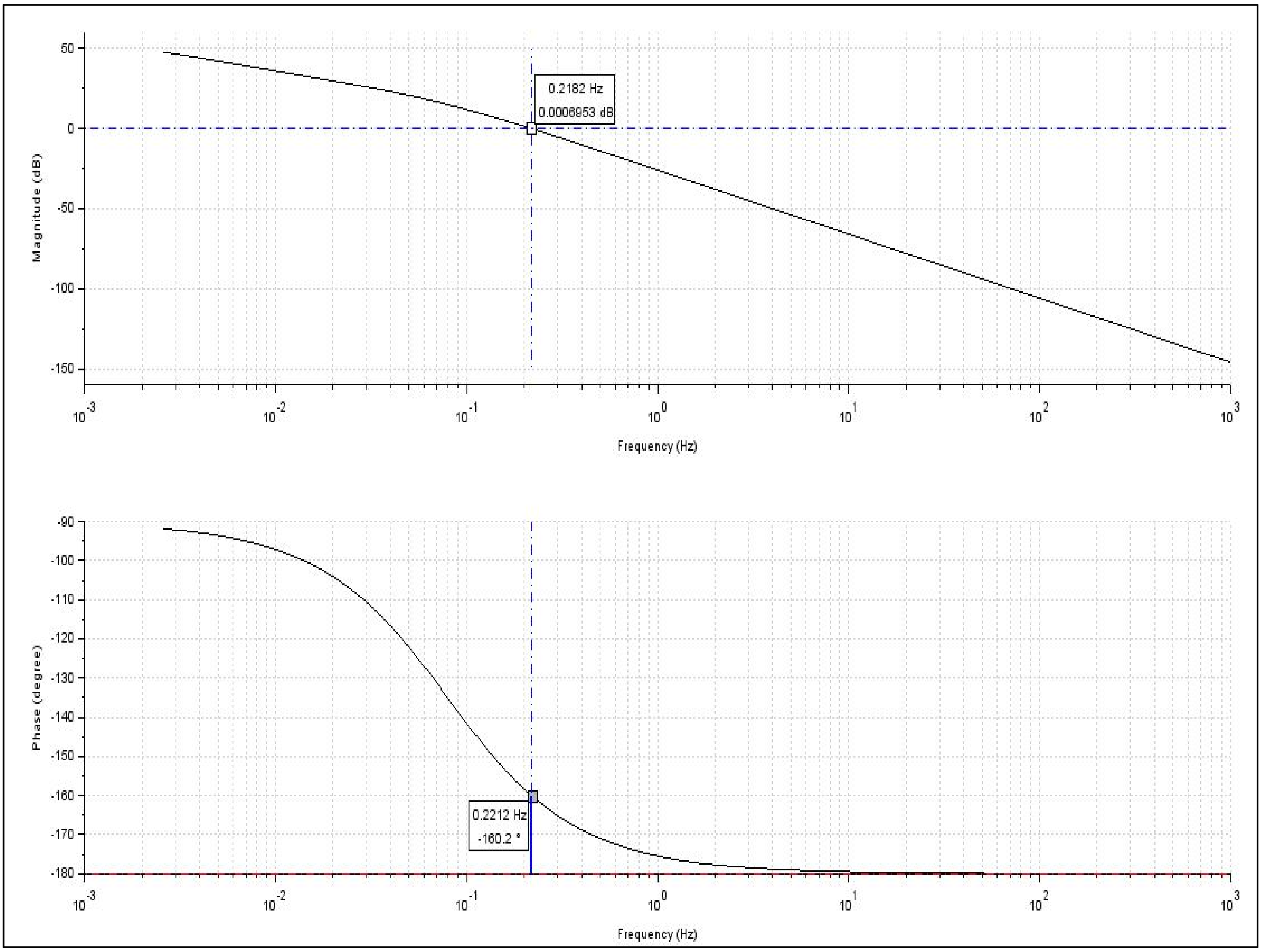
```
s=%s;
Num=4;
Den=s*(2*s+1);
TF=syslin( ‘c’ ,Mum/Den)//to create transfer function
show_margins(TF)//to show gain and phase margins
[gain fr]=g_margin(TF)// for gain value & phase crossover
frequency
[phase fr]=p_margin(TF)//for gain value & phase crossover
frequency
```



```
--> s=%s;

--> Den=s*(2*s+1);

--> TF=syslin('c',Num/Den)
 TF  =


     4
   ------
   s +2s²

--> show_margins(TF)

--> [gain fr]=g_margin(TF)
 gain  =

   Inf
 fr  =

    []

--> [Phase fr]=p_margin(TF)
 Phase  =

   20.040400
 fr  =

   0.2181586

--> |
```

code to find phase margin, gain margin and phase crossover frequency and gain crossover frequency

Bode plot


50
0
-50
-100
-150
Magnitude (dB)
0.2182 Hz
0.0006953 dB
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
Frequency (Hz)
-90
-100
-110
-120
-130
-140
-150
-160
-170
-180
Phase (degree)
0.2212 Hz
-160.2 °
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
Frequency (Hz)