

Scilab Manual for  
Control System Engineering  
by Prof Priyen S. Patel  
Electrical Engineering  
Swarnnim Institute Of Technology<sup>1</sup>

Solutions provided by  
Prof Priyen S. Patel  
Electrical Engineering  
Swarnnim Institute Of Technology

May 31, 2026

<sup>1</sup>Funded by a grant from the National Mission on Education through ICT, <http://spoken-tutorial.org/NMEICT-Intro>. This Scilab Manual and Scilab codes written in it can be downloaded from the "Migrated Labs" section at the website <http://scilab.in>



# Contents

List of Scilab Solutions	3
1 To study open loop control system.	5
2 To study closed loop control system.	7
3 Develop a program to study transfer function from poles and zeros.	9
4 Develop a program to study transient response for given system.	12
5 Develop a program to study the controllability and observability of a given system.	16
6 Develop a program to study routh stability criterion.	19
7 Develop a program to study step responses of a 2nd order system.	23
8 Develop a program to obtain Root locus using Scilab.	25
9 Develop a program to obtain Nyquist Plot using Scilab.	30
10 Develop a program to obtain Bode Plot using Scilab.	35

# List of Experiments

Solution 3.01	Obtain Transfer Function . . . . .	9
Solution 4.01	Transient Response . . . . .	12
Solution 5.01	Controllability and Observability . . . . .	16
Solution 6.01	Routh Stability Criterion . . . . .	19
Solution 7.01	Step Response of 2nd order system . . . . .	23
Solution 8.01	Root Locus . . . . .	25
Solution 9.01	Nyquist plot . . . . .	30
Solution 10.01	Bode Plot . . . . .	35

# List of Figures

1.1	Open Loop system	6
1.2	Open Loop system	6
2.1	Close Loop system	8
2.2	Close Loop system	8
3.1	Obtain Transfer Function	11
4.1	Transient Response	13
4.2	Transient Response	13
4.3	Transient Response	15
5.1	Controllability and Observability	18
6.1	Routh Stability Criterion	20
6.2	Routh Stability Criterion	22
7.1	Step Response of 2nd order system	24
8.1	Root Locus	27
8.2	Root Locus	28
8.3	Root Locus	29
9.1	Nyquist plot	33
9.2	Nyquist plot	34
9.3	Nyquist plot	34
10.1	Bode Plot	38
10.2	Bode Plot	38
10.3	Bode Plot	39

# **Experiment: 1**

**To study open loop control system.**

This code can be downloaded from the website [www.scilab.in](http://www.scilab.in)

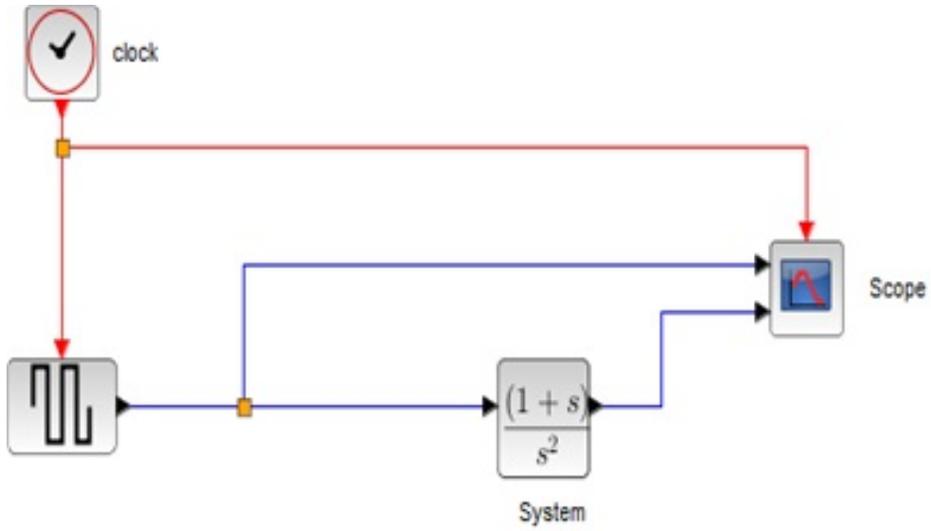


Figure 1.1: Open Loop system

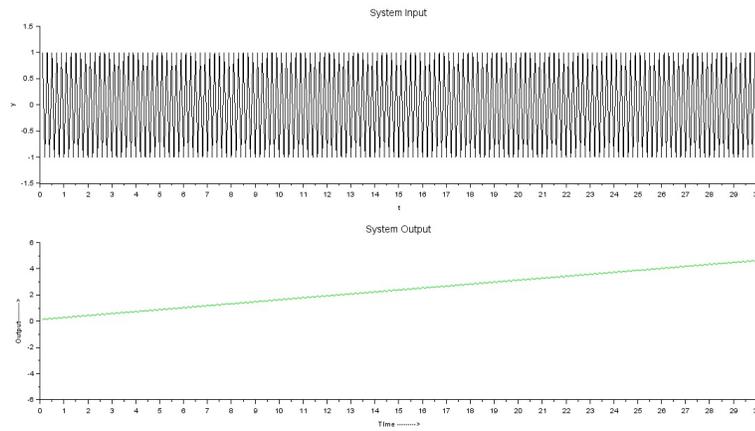


Figure 1.2: Open Loop system

## **Experiment: 2**

**To study closed loop control system.**

This code can be downloaded from the website [www.scilab.in](http://www.scilab.in)

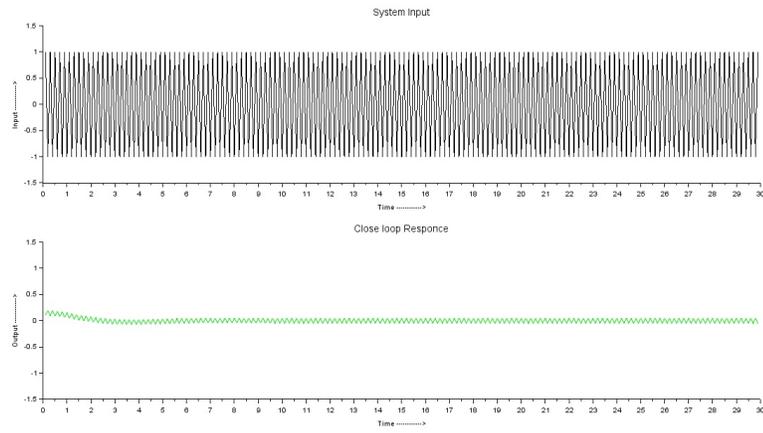


Figure 2.1: Close Loop system

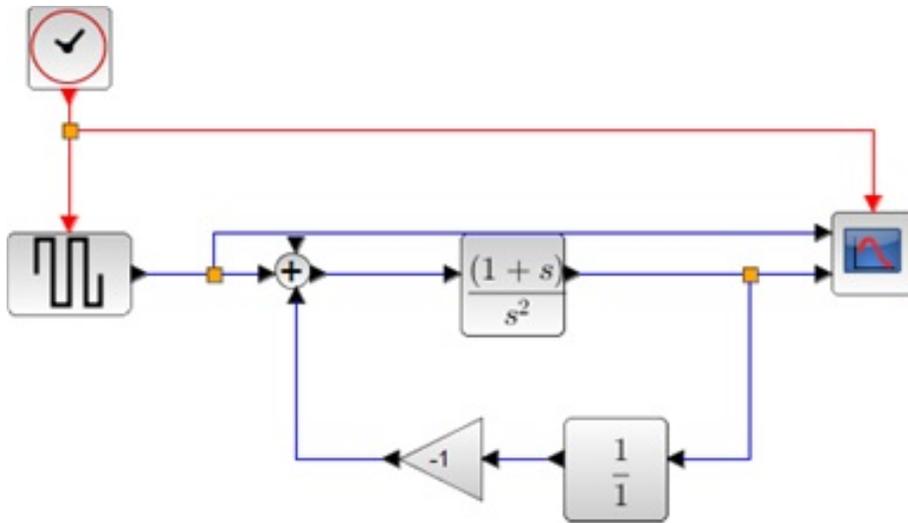


Figure 2.2: Close Loop system

## Experiment: 3

Develop a program to study transfer function from poles and zeros.

Scilab code Solution 3.01 Obtain Transfer Function

```
1 // OS : Windows 7
2 // Scilab : 6.0.1
3
4 // To find Transfer function of given system
5
6 clc
7 close
8 s=%s;
9 // From Pole & Zero
10 // For First order transfer function
11 z1=input('enter the value of z1 = ')//z1=0
12 p1=input('enter the value of p1 = ')//p1=-5
13 tf1=(s-z1)/((s-p1))
14 disp(tf1,"Transfer function of first order system =
    ")
15
16
```

```

17 // For Second order transfer function
18 z1=input('enter the value of z1 = ')//z1=-2
19 p1=input('enter the value of p1 = ')//p1=-5
20 p2=input('enter the value of p2 = ')//p2=1
21 tf2=(s-z1)/((s-p1)*(s-p2))
22 disp(tf2,"Transfer function of Second order system
    = ")
23
24 // From Numerator & Denominator
25
26 numm = input('enter the Co-efficient of numerator:')
    ; // [1]
27 denn = input('enter the Co-efficient of denominator:
    ');//[-7 3 3 1]
28
29 A = poly([numm], 's', 'c')
30 B = poly([denn], 's', 'c')
31
32 tff = syslin('c',A,B)
33 disp(tff, 'Transfer Function is:')

```

---

```

Scilab 6.0.1 Console
File Edit Control Applications ?
Scilab 6.0.1 Console

enter the value of z1 = 0
enter the value of p1 = -5

Transfer function of first order system =

      s
-----
    5 + s

enter the value of z1 = -2
enter the value of p1 = -5
enter the value of p2 = 1

Transfer function of Second order system =

      2 + s
-----
    -5 + 4s + s2

enter the Co-efficient of numerator:[1]
enter the Co-efficient of denominator:[-7 3 3 1]

Transfer Function is:

      1
-----
    -7 + 3s + 3s + s2

-->

```

Figure 3.1: Obtain Transfer Function

## Experiment: 4

Develop a program to study transient response for given system.

Scilab code Solution 4.01 Transient Response

```
1 //  
-----  
2 // To study Transient Response of given system.  
3 //  
-----  
4  
5  
6 // OS : Windows 7  
7 // Scilab : 6.0.1  
8  
9 clc
```

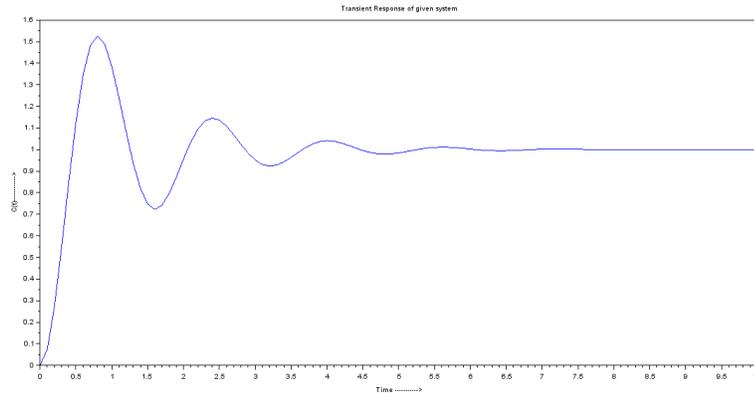


Figure 4.1: Transient Response



Figure 4.2: Transient Response

```

10 close
11 s=%s
12 // To form Numerator in form of  $wn^2$ 
13 num = input('Enter the value of Numerator =')//num
    =16
14 wn=sqrt(num)
15 // To form denominator in form of  $S^2+(2*zeta*wn)*s+wn^2$ 
16 a= input('Enter the value of coefficient of term  $S^2$ 
    =')//a=1
17 b= input('Enter the value of coefficient of term  $S=$ 
    )//b=1.6
18 den= [a*s^2+b*s+wn^2]
19 TF = syslin('c',num,den)//transfer function
20 disp(TF,"Transfer Function of system = ")
21 t=0:0.1:10;
22 y1 = csim('step', t, TF);//time response
23 title('Transient Response of given system');
24 xlabel('Time ----->');
25 ylabel('C(t)----->');
26 plot(t, y1)
27 disp(wn,"Value of wn =")
28 z=b/(2*wn)
29 disp(z,"Value of zeta =")
30 if z<1 then
31     disp("-----")
32     disp("system Is Under Damped")
33     disp("-----")
34 elseif z==1 then
35     disp("-----")
36     disp("system Is Critically Damped")
37     disp("-----")
38 else
39     disp("-----")
40     disp("system Is Critically Damped")
41     disp("-----")
42 end
43 wd=wn*(sqrt(1-z^2));

```



Figure 4.3: Transient Response

```

44 disp(wd,"Damped frequency =")
45 theta=atan((sqrt(1-z^2)/z));
46 disp(theta, "Angle in radians=")
47 Tr = ((%pi-theta)/wd)
48 disp(Tr," Rise TIme =")
49 Tp=%pi/wd
50 disp(Tp," Peak TIme =")
51 Ts=4/(z*wn)
52 disp(Ts," Settling Time =")
53 Mp=100*%e^((-z*%pi)/(sqrt(1-z^2)))
54 disp(Mp," Peak overshoot % =")

```

---

## Experiment: 5

Develop a program to study the controllability and observability of a given system.

Scilab code Solution 5.01 Controllability and Observability

```
1
2 //


---


3 // To Study the controllability and observability of
4 // a given system.


---


5
6 // OS : Windows 7
7 // Scilab : 6.0.1
8 clc;
9 clear all;
10
11 // State space representation
12
```

```

13 A=input('Enter the value of Matrix A =');//[0 0 0; 1
    0 -3; 0 1 -4];
14 B=input('Enter the value of Matrix B =');//[40; 10;
    0];
15 C=input('Enter the value of Matrix C =');//[0 0 1];
16 D=input('Enter the value of Matrix D =');//[0];
17 sys=syslin('c',A,B,C,D)
18
19 // For Controllability
20 n=cont_mat(sys)
21 mprintf('Controllability matrix is ')
22 disp(n)
23
24 if rank(n)==3 then
25     disp('System is controllable as rank is 3')
26 else
27     disp('System is uncontrollable')
28 end
29
30 // For Observability
31 m=obsv_mat(sys)
32 mprintf('Observability matrix is ')
33 disp(m)
34
35 if rank(m)==3 then
36     disp('System is observable as rank is 3')
37 else
38     disp('System is unobservable')
39 end

```

---

```
Scilab 6.0.1 Console
File Edit Control Applications ?
Scilab 6.0.1 Console
Enter the value of Matrix A = [0 0 0; 1 0 -3; 0 1 -4]
Enter the value of Matrix B =[40; 10; 0]
Enter the value of Matrix C =[0 0 1]
Enter the value of Matrix D =0
Controllability matrix is
40.  0.  0.
10.  40. -30.
0.  10.  0.
System is controllable as rank is 3
Observability matrix is
0.  0.  1.
0.  1. -4.
1. -4. 13.
System is observable as rank is 3
--> |
```

Figure 5.1: Controllability and Observability

## Experiment: 6

Develop a program to study routh stability criterion.

Scilab code Solution 6.01 Routh Stability Criterion

```
1
2 //


---


3 // To Study the Routh–Hurwitz Criterion
4 //


---


5
6 // OS : Windows 7
7 // Scilab : 6.0.1
8
9 clc;
10 clear all;
11
12 D=input('Input coefficients of characteristic
           equation, i.e: [a0 an+1 an+2_ _ _ _an]= ');
```



Figure 6.1: Routh Stability Criterion

```

13 //Case- 1 enter cvalue of D = [1 2 8 4 3]
14 //Case- 2 enter cvalue of D = [1 1 3 6 6]
15 l=length (D);
16
17 disp('Roots of characteristic equation are=')
18 roots(D)
19 if modulo(l,2) = =0
20     m=zeros(1,l/2);
21     [cols ,rows]=size(m);
22     for i=1:rows
23         m(1,i)=det(1,(2*i)-1);
24         m(2,i)=det(1,(2*i));
25     end
26 else
27     m=zeros(1,(l+1)/2);
28     [cols ,rows]=size(m);
29     for i=1:rows
30         m(1,i)=D(1,(2*i)-1);
31     end
32     for i=1:((l-1)/2)

```

```

33         m(2,i)=D(1,(2*i));
34     end
35 end
36
37 for j=3:cols
38
39     if m(j-1,1)==0
40         m(j-1,1)=0.001;
41     end
42
43     for i=1:rows-1
44         m(j,i)=(-1/m(j-1,1))*det([m(j-2,1) m(j-2,i
45             +1);m(j-1,1) m(j-1,i+1)]);
46     end
47 end
48 disp('—————The Routh–Hurwitz array is=—————',
49     m)
49 //—————End of Bulding array
50
51 //Checking for sign change
52 Temp=sign(m);a=0;
53 for j=1:cols
54     a=a+Temp(j,1);
55 end
56 if a==cols
57     disp('          ———> Sign Not Changed in first
58         Column so System is Stable <————')
59 else
60     disp('          ———> Sign  Changed in first
61         Column so System is Unstable <————')
62 end

```

---

```
Scilab 6.0.1 Console
File Edit Control Applications ?
Scilab 6.0.1 Console
Input coefficients of characteristic equation, i.e.:[a0 an+1 an+2 _ _ _ _an]* [1 1 3 6 6]

Roots of characteristic equation are-
1. 3. 6.
1. 6. 0.
-3. 6. 0.
8. 0. 0.
6. 0. 0.

-----The Routh-Hurwitz array is-----
----> Sign Changed in first Column so System is Unstable <----
-->
```

Figure 6.2: Routh Stability Criterion

## Experiment: 7

Develop a program to study step responses of a 2nd order system.

Scilab code Solution 7.01 Step Response of 2nd order system

```
1 //  


---

  
2 // TO STUDY STEP RESPONSES OF A 2ND ORDER SYSTEM  
3 //  


---

  
4 // OS : Windows 7  
5 // Scilab : 6.0.1  
6 clc;  
7 clear all;  
8 t=0:0.000001:0.0002;  
9 zeta=input('Enter the values for zeta =')//[0.4 1  
10      1.6];  
10 cv=[1 2 3];
```

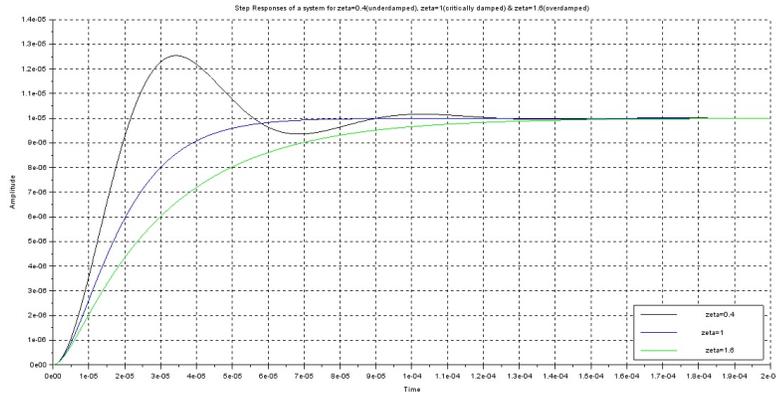


Figure 7.1: Step Response of 2nd order system

```

11 s=%s;
12 wn = input('Enter the value for wn =')//10^5;
13 for n=1:3
14 den = s^2 + 2*zeta(n)*(wn)*s+(wn^2);
15 P = syslin('c',wn,den);
16 Ps=csim('step',t,P);
17 plot2d(t,Ps,style=cv(n));
18 end;
19 xgrid;
20 xtitle(['Step Responses of a system for zeta=0.4(
        underdamped), zeta=1(critically damped) & zeta
        =1.6(overdamped)'], 'Time', 'Amplitude');
21 legends(['zeta=0.4'; 'zeta=1'; 'zeta=1.6'], [1,2,3], opt
        =4);
22
23
24 //————— OUTPUT—————
25
26 //Enter the values for zeta =[0.4 1 1.6]
27
28 //Enter the value for wn =10^5

```

## Experiment: 8

Develop a program to obtain  
Root locus using Scilab.

Scilab code Solution 8.01 Root Locus

```
1
2 //


---


3 // To Study the Root Locus
4 //


---


5
6 // OS : Windows 7
7 // Scilab : 6.0.1
8
9 clc
10 close
11 s=%s
12 num=input('Enter the Numerator =')
13 // Case - 1 Enter the Numerator = (s+1)
14 // Case - 2 Enter the Numerator = (s+1)
15 // Case - 3 Enter the Numerator = 1
```

```

16 den=input('Enter the Denominator =')
17 // Case - 1 Enter the Denominator = (s^2*(s+3)*(s+5)
18 // Case - 2 Enter the Denominator = (s*(s+2)*(s^2+2*
19 // Case - 3 Enter the Denominator =(s*(s+2)*(s+5))
20 TF = syslin('c',num,den)//Transfer function
21 disp(TF,"Transfer Function of system = ")
22 h=syslin('c',num,den)
23 evans(h,100)
24
25 //-----Output-----
26 //Case 1
27 //-----
28 //Enter the Numerator =s+1
29
30 //Enter the Denominator =(s^2*(s+3)*(s+5))
31
32
33 // Transfer Function of system =
34
35
36 //      1 + s
37 //      -----
38 //      2      3      4
39 //      15s + 8s + s
40 //-----
41 //Case 2
42 //-----
43 //Enter the Numerator =s+1
44
45 //Enter the Denominator =s*(s+2)*(s^2+2*s+5)
46
47
48 //Transfer Function of system =
49
50
51 //      1 + s

```

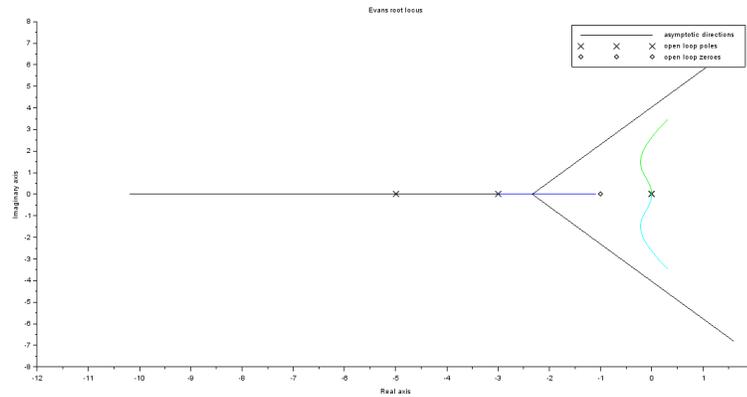


Figure 8.1: Root Locus

```

52 // _____
53 //          2    3    4
54 //    10s + 9s + 4s + s
55 // _____
56 // Case 3
57 // _____
58 // Enter the Numerator =1
59
60 // Enter the Denominator =(s*(s+2)*(s+5))
61
62
63 // Transfer Function of system =
64
65
66 //          1
67 // _____
68 //          2    3
69 //    10s + 7s + s

```

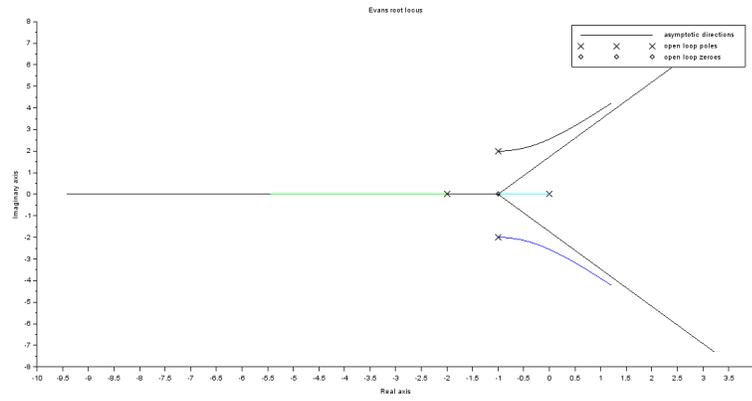


Figure 8.2: Root Locus

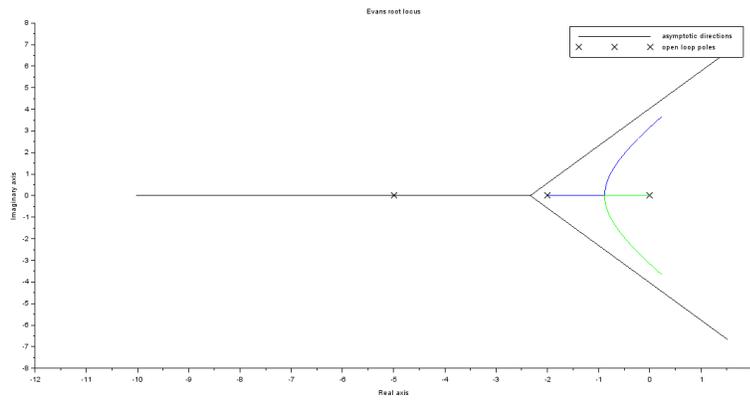


Figure 8.3: Root Locus

## Experiment: 9

# Develop a program to obtain Nyquist Plot using Scilab.

Scilab code Solution 9.01 Nyquist plot

```
1
2 //


---


3 // To Study the Nyquist plot
4 //


---


5
6 // OS : Windows 7
7 // Scilab : 6.0.1
8 // Toolboxes : Maxima 5.20.1 and ActivePerl 5.20.2
9 clc
10 close
11
12 s=poly(0, 's')
13 num=input('Enter the Numerator =')
14 // Case - 1 Enter the Numerator = 1
15 // Case - 2 Enter the Numerator = (5+s)*(s+40)
```

```

16 // Case - 3 Enter the Numerator = (s+1)
17 den=input('Enter the Denominator =')
18 // Case - 1 Enter the Denominator = (s*(s+1)*(2*s+1)
19 // Case - 2 Enter the Denominator = (s^3)*(s+200)*(s
+1000)
20 // Case - 3 Enter the Denominator =(s^2)*(s+5)*(s
+10)
21 TF = syslin('c',num,den)//Transfer function
22 disp(TF,"Transfer Function of system = ")
23 h=syslin('c',num,den)
24 nyquist(h);
25
26 //-----Output-----
27 //Case 1
28 //-----
29 //Enter the Numerator =1
30
31 //Enter the Denominator = (s*(s+1)*(2*s+1))
32
33
34 // Transfer Function of system =
35
36
37 //          1
38 //  -----
39 //          2    3
40 //   s  + 3s  + 2s
41
42 //-----
43 //Case 2
44 //-----
45
46 //Enter the Numerator =(5+s)*(s+40)
47
48 //Enter the Denominator =(s^3)*(s+200)*(s+1000)
49
50

```

```

51 // Transfer Function of system =
52
53 //
54 //      2
55 //      200 + 45s + s
56 //      -----
57 //      3      4      5
58 //      200000s + 1200s + s
59 //-----
60 //Case 3
61 //-----
62 //Enter the Numerator =(s+1)
63
64 //Enter the Denominator =(s^2)*(s+5)*(s+10)
65
66
67 // Transfer Function of system =
68
69
70 //
71 //      1 + s
72 //      -----
73 //      2      3      4
74 //      50s + 15s + s

```

---

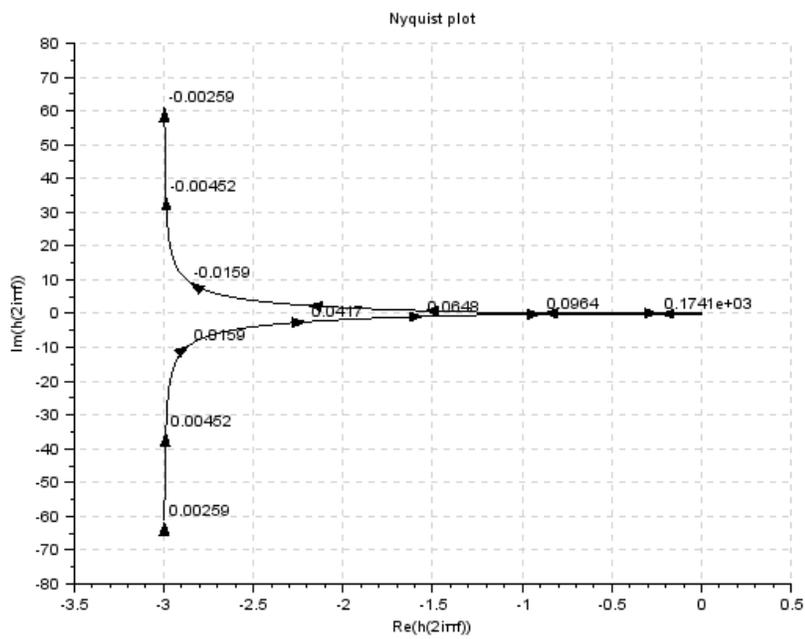


Figure 9.1: Nyquist plot

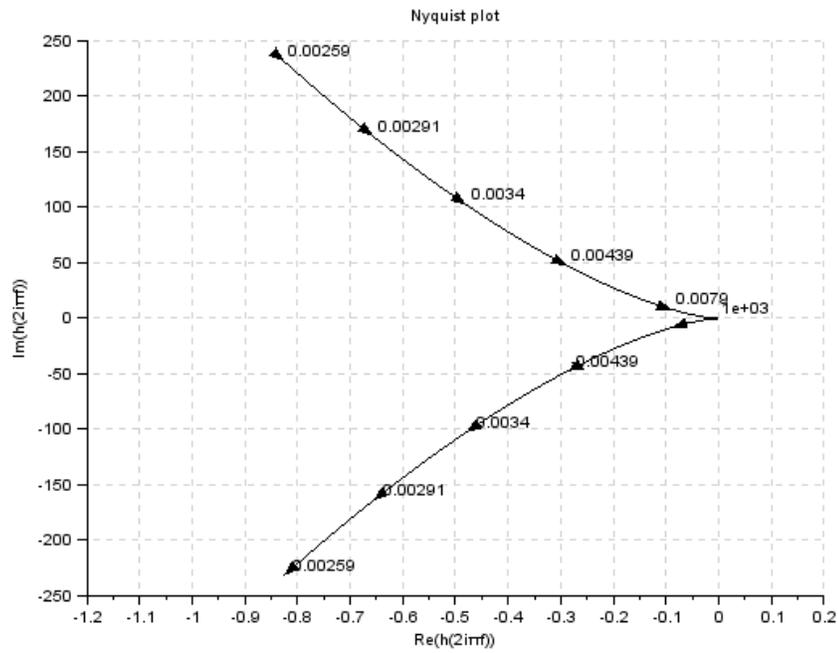


Figure 9.2: Nyquist plot

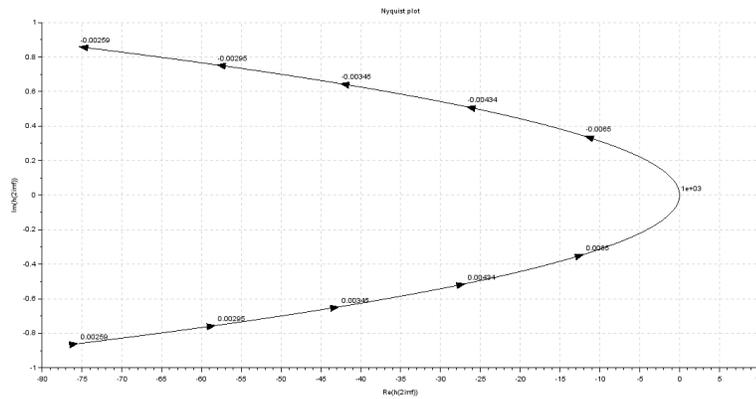


Figure 9.3: Nyquist plot

# Experiment: 10

## Develop a program to obtain Bode Plot using Scilab.

Scilab code Solution 10.01 Bode Plot

```
1 //  


---

  
2 // To Study the Bode plot  
3 //  


---

  
4  
5 // OS : Windows 7  
6 // Scilab : 6.0.1  
7  
8 clc  
9 close  
10  
11 s=poly(0, 's')  
12 num=input('Enter the Numarator =')  
13 // Case - 1 Enter the Numarator = 20  
14 // Case - 2 Enter the Numarator = (s+1)  
15 // Case - 3 Enter the Numarator = 1
```

```

16 den=input('Enter the Denominator =')
17 // Case - 1 Enter the Denominator = (s)*(1+s)
    *(1+0.5*s)
18 // Case - 2 Enter the Denominator = (s)*(s+2)*(s+5)
19 // Case - 3 Enter the Denominator =(s)*(s+1)*(s+5)
    *(s+10)
20 TF = syslin('c',num,den)//Transfer function
21 disp(TF,"Transfer Function of system = ")
22 h=syslin('c',num,den)
23 clf();
24 bode(h,0.1,100)
25 g_margin(h)
26 show_margins(h)
27 p_margin(h)
28 show_margins(h)
29
30 //-----Output-----
31 //Case 1
32 //-----
33 //Enter the Numarator =20
34
35 //Enter the Denominator =(s)*(1+s)*(1+0.5*s)
36
37
38 // Transfer Function of system =
39
40
41 //          20
42 //  _____
43 //          2      3
44 //   s + 1.5s + 0.5s
45 //-----
46 //Case 2
47 //-----
48 //Enter the Numarator =s+1
49
50 //Enter the Denominator =(s)*(s+2)*(s+5)
51

```

```

52
53 //Transfer Function of system =
54
55
56 //      1 + s
57 //      -----
58 //           2   3
59 //      10s + 7s + s
60 //-----
61 //Case 3
62 //-----
63 //Enter the Numarator =1
64
65 //Enter the Denominator = (s)*(s+1)*(s+5)*(s+10)
66
67
68 // Transfer Function of system =
69
70
71 //      1
72 //      -----
73 //           2   3   4
74 //      50s + 65s + 16s + s

```

---

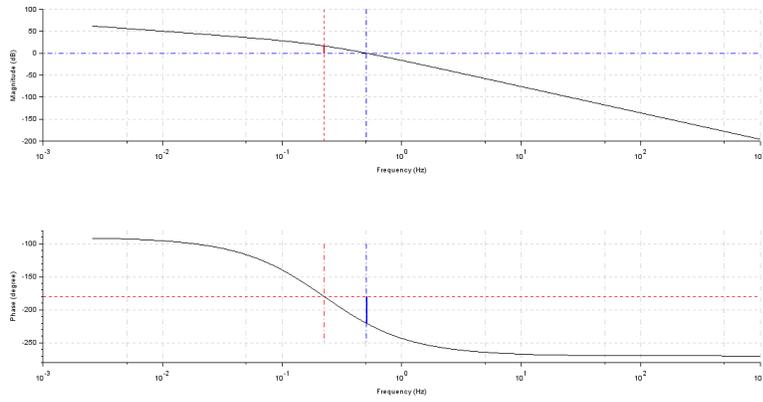


Figure 10.1: Bode Plot

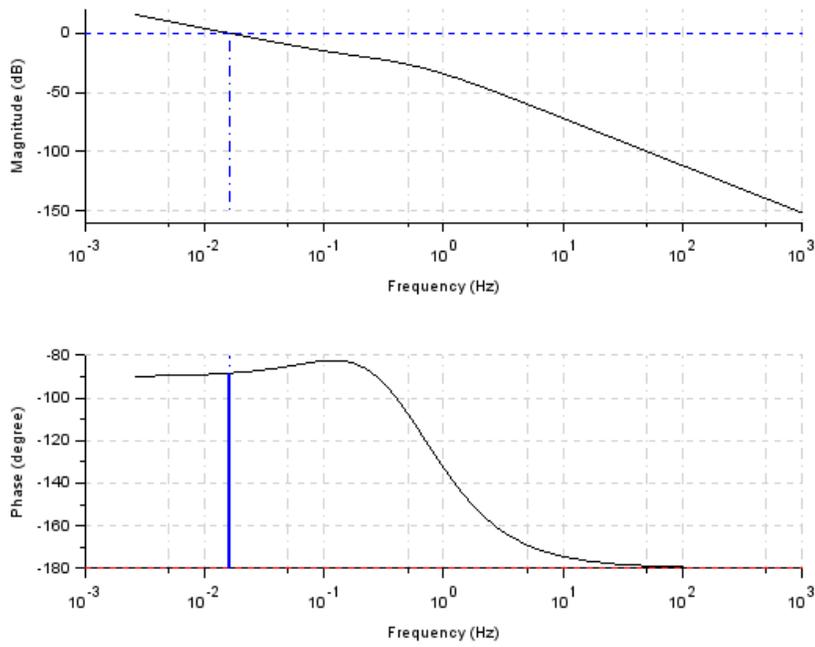


Figure 10.2: Bode Plot

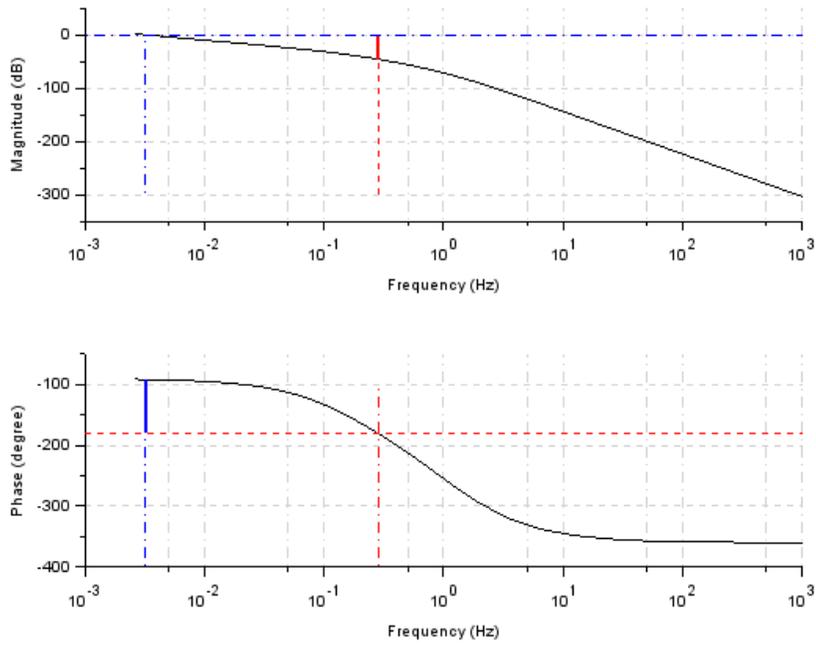


Figure 10.3: Bode Plot