

Scilab Manual for  
Control System Design  
by Prof Deepti Khimani  
Instrumentation Engineering  
VESIT<sup>1</sup>

Solutions provided by  
Prof Mrs. Deepti Khimani  
Instrumentation Engineering  
Mumbai University/VES Institute of Technology

June 10, 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 Obtain state model of the second order system cascaded with active lead circuit and show its step response.	5
2 Determine eigen values of the state model. Also convert the state model into transfer function.	8
3 Transform the given system having distinct eigen values into controllable canonical and diagonal form.	10
4 Obtain the step and impulse response of the state model.	13
5 Check for the controllability and observability of a given system.	17
6 Obtain state feedback gain matrix for the given system.	19
7 Design a full state observer for the system.	22
8 Determine steady state error of the given system.	25
9 Compensation of system using lead compensator designed via root locus technique.	28
10 Design a lead compensator for the given system using bode plot.	30

# List of Experiments

Solution 1.01	Lab 01	5
Solution 2.02	Lab02	8
Solution 3.03	Lab3	10
Solution 4.04	Lab 04	13
Solution 5.05	Lab05	17
Solution 6.06	Lab 06	19
Solution 7.07	Lab 07	22
Solution 8.08	Lab 08	25
Solution 9.09	Lab 09	28
Solution 10.10	Lab 10	30

# List of Figures

1.1	Lab 01	7
4.1	Lab 04	15
4.2	Lab 04	16
6.1	Lab 06	21
7.1	Lab 07	24
8.1	Lab 08	27
9.1	Lab 09	29
10.1	Lab 10	32
10.2	Lab 10	33

# Experiment: 1

Obtain state model of the second order system cascaded with active lead circuit and show its step response.

Scilab code Solution 1.01 Lab 01

```
1 //  


---

  
2 // Lab. 01: Obtain state model of the second order  
3 // system cascaded with active  
4 // lead circuit. Show its step response.  


---

  
5  
6 //scilab -5.5.0  
7 //Operating System : OS X 10.9.3  
8  
9 //Clean the environment  
10 clc;
```

```

11 clear all;
12 clf;
13
14 // Compensator model
15 R1=1000; R2=5e3; C1=1e-6; C2=1e-5;
16 kc=5;
17 s=poly(0, 's');
18 g=kc*(R1*C1*s+1)/(R2*C2*s+1);
19
20 // System transfer function
21 g1=0.2/(s^2+1.7*s+1);
22
23 // Overall transfer function
24 sys=tf2ss(g*g1);
25
26 // Unit step response
27 t=linspace(0,10,1000);
28 y=csim('step',t,sys);
29 plot(t,y);
30 title('Unit step response of the electrical system',
        'fontsize',4)
31 xlabel('Time t','fontsize',2)
32 ylabel('Response y(t)','fontsize',2)
33 //set(gca(),"grid",[0.3 0.3])

```

---

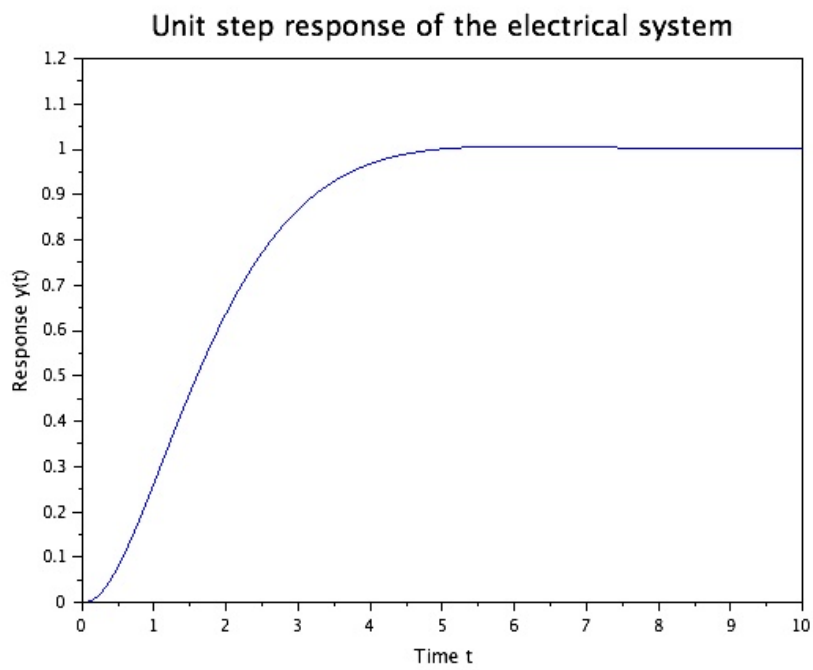


Figure 1.1: Lab 01

## Experiment: 2

Determine eigen values of the state model. Also convert the state model into transfer function.

Scilab code Solution 2.02 Lab02

```
1
2 //


---


3 // Lab. 02: Determine eigen values of the state
  model.
4 // Convert the state model into transfer function.
5 //


---


6
7 //scilab -5.5.0
8 //Operating System : OS X 10.9.3
9
10 //Clean the environment
```

```
11 clc;
12 clear all;
13 // clf;
14
15 // State space representation
16 A=[0 1 0; 0 0 1; -5 -25 -5];
17 B=[0; 25; -120];
18 C=[1 0 0];
19 D=0;
20
21 sys1=sslin('c',A,B,C,D);
22 mprintf('State space representation of the given
    system is')
23 disp(sys1)
24
25 // Eigen values of system matrix
26 eig_val=spec(A)
27 mprintf('Eigen values of the system matrix are')
28 disp(eig_val)
29
30 // Transfer function of the given system
31 g1=ss2tf(sys1)
32 mprintf('Transfer function representation of the
    given system is')
33 disp(g1)
```

---

## Experiment: 3

Transform the given system having distinct eigen values into controllable canonical and diagonal form.

Scilab code Solution 3.03 Lab3

```
1
2 //


---


3 // Lab. 03: Transform the given system having
4 // distinct eigen values into
5 // controllable canonical and diagonal form.


---


6
7 //scilab -5.5.0
8 //Operating System : OS X 10.9.3
9
10 //Clean the environment
```

```

11 clc;
12 clear all;
13 // clf;
14
15 // State space model
16 A=[-3 1; 1 -3];
17 B=[1;2];
18 C=[2 3];
19 D=0;
20
21 sys=sslin('c',A,B,C,D)
22 mprintf('State space representation of the given
    system is ')
23 disp(sys)
24
25
26 // Eigen values of system matrix
27 eig_val=spec(A)
28 mprintf('Eigen values of the system matrix are ')
29 disp(eig_val)
30
31 // Controllable canonical form
32 [Ac, Bc T]=canon(A,B)
33 T=flipdim(T,2);
34 Ac=T\A*T;
35 Bc=T\B;
36 Cc=C*T;
37 Dc=D;
38 sysc=sslin('c',Ac,Bc,Cc,Dc)
39 mprintf('State space representation of the given
    system in Controllable canonical form is ')
40 disp(sysc)
41
42 // Diagonal form
43 [Ad M]=bdiag(A);
44 Bd=M\B;
45 Cd=C*M;
46 Dd=D;

```

```
47 sysd=syslin('c',Ad,Bd,Cd,Dd)
48 mprintf('State space representation of the given
         system in Diagonal form is ')
49 disp(sysd)
```

---

# Experiment: 4

## Obtain the step and impulse response of the state model.

Scilab code Solution 4.04 Lab 04

```
1 //  


---

  
2 // Lab. 04: Obtain the step and impulse response of  
  the state model.  
3 //  


---

  
4  
5 //scilab -5.5.0  
6 //Operating System : OS X 10.9.3  
7  
8 //Clean the environment  
9 clc;  
10 clear all;  
11 clf;  
12  
13 // State space representation  
14 A=[-2 -1; -1 -1];
```

```

15 B=[1;1];
16 C=[0 2];
17 D=0;
18 x0=[0;5]; // Initial condition
19 sys=syslin('c',A,B,C,D)
20
21 // Response to a given input
22 figure(0)
23 t=linspace(0,20,1001);
24 temp=size(t);
25 u=ones(temp(1),temp(2)); // Exogenous signal(step)
26 y=csim(u,t,sys,x0)
27 plot(t,y)
28 title('Unit step response of the system','fontsize'
    ,4)
29 xlabel('Time t','fontsize',2)
30 ylabel('Response y(t)','fontsize',2)
31
32 // Response to a given input
33 figure(1)
34 t=linspace(0,10,1001);
35 y=csim('impuls',t,sys)
36 plot(t,y)
37 title('Impulse response of the system','fontsize',4)
38 xlabel('Time t','fontsize',2)
39 ylabel('Response y(t)','fontsize',2)

```

---

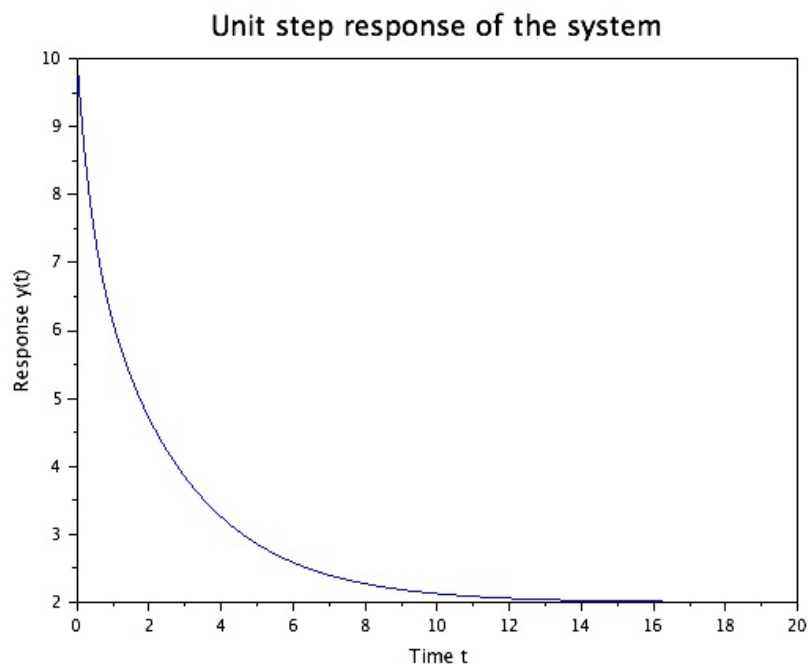


Figure 4.1: Lab 04

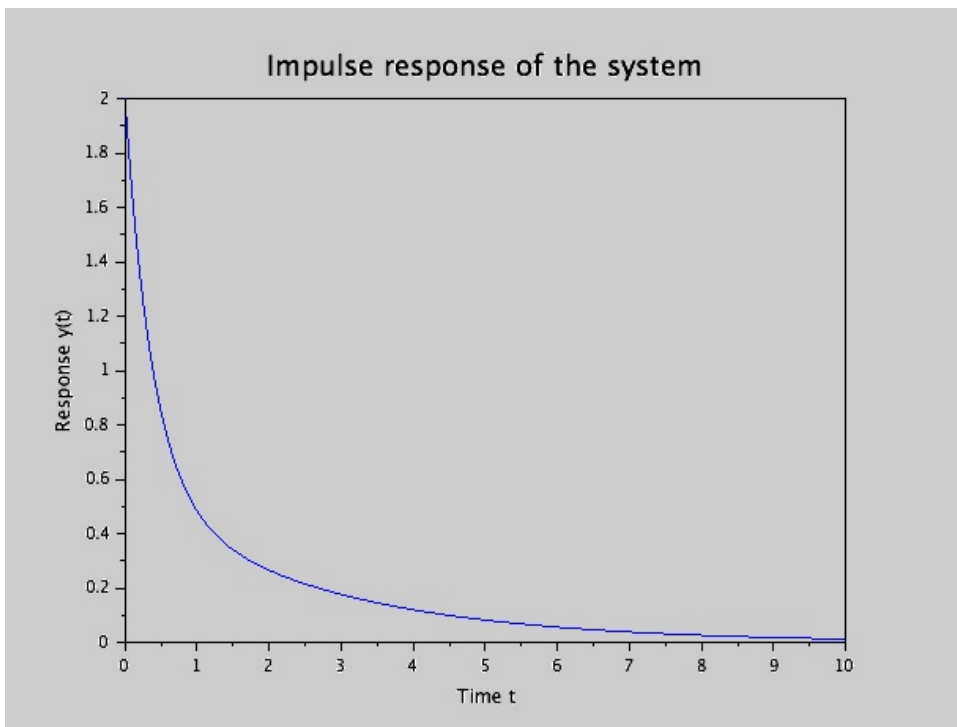


Figure 4.2: Lab 04

# Experiment: 5

Check for the controllability and observability of a given system.

Scilab code Solution 5.05 Lab05

```
1
2 //


---


3 // Lab. 05: Check for the controllability and
  observability of a given system.
4 //


---


5
6 //scilab -5.5.0
7 //Operating System : OS X 10.9.3
8
9 //Clean the environment
10 clc;
11 clear all;
12 //clf;
```

```
13
14 // State space representation
15 A=[-5 1 0; 0 -2 1; 0 0 -1];
16 B=[6 0 1]';
17 C=[1 0 0];
18 D=0;
19 sys=syslin('c',A,B,C,D)
20
21 // Controllability test
22 n=cont_mat(sys)
23 mprintf('Controllability matrix is ')
24 disp(n)
25
26 if rank(n)==3 then
27     disp('System is controllable')
28 else
29     disp('System is uncontrollable')
30 end
31
32 // Observability test
33 m=obsv_mat(sys)
34 mprintf('Observability matrix is ')
35 disp(m)
36
37 if rank(m)==3 then
38     disp('System is observable')
39 else
40     disp('System is unobservable')
41 end
```

---

# Experiment: 6

## Obtain state feedback gain matrix for the given system.

Scilab code Solution 6.06 Lab 06

```
1 //  


---

  
2 // Lab. 06: Obtain state feedback gain matrix for  
   the given system.  
3 //  


---

  
4  
5 //scilab -5.5.0  
6 //Operating System : OS X 10.9.3  
7  
8 //Clean the environment  
9 clc;  
10 clear all;  
11 clf;  
12  
13 // State space representation  
14 A=[0 1 0; 0 0 1; -1 -5 -6];
```

```
15 B=[0 0 1]';
16 C=[0 0 1];
17 D=0;
18
19 // Desired poles
20 Pd=[-1+2*i -1-2*i -10];
21
22 // State feedback gain matrix
23 K=ppol(A,B,Pd)
24
25 //Closed loop system
26 sys=syslin('c',A-B*K,B,C,D)
27
28 //Response of closed loop system
29 t=linspace(0,20,1001);
30 y=csim('step',t,sys)
31 plot(t,y)
32 title('Response of the closed loop system','fontsize
    ',4)
33 xlabel('Time t','fontsize',2)
34 ylabel('Response y(t)','fontsize',2)
```

---

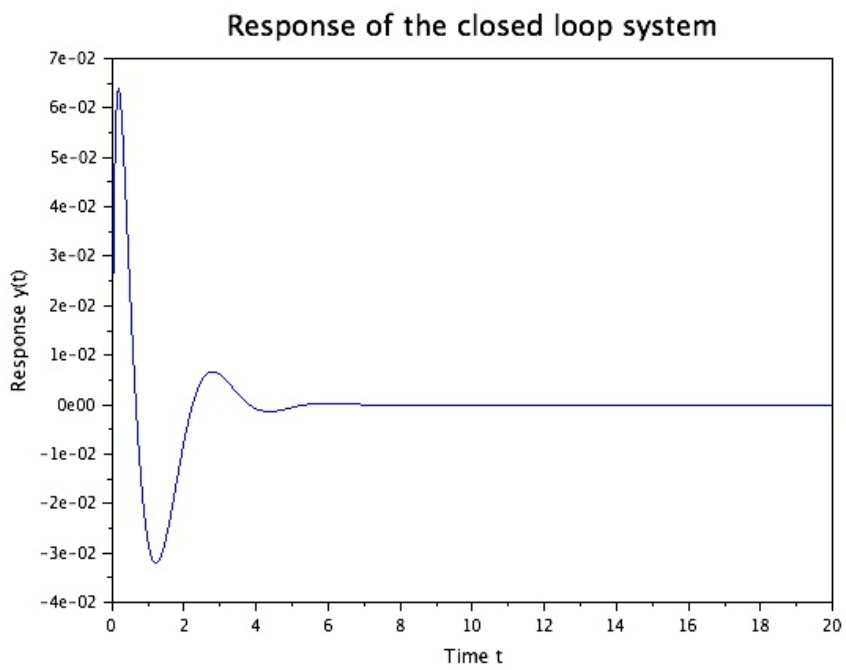


Figure 6.1: Lab 06

# Experiment: 7

## Design a full state observer for the system.

Scilab code Solution 7.07 Lab 07

```
1 //  
2 // Lab. 07: Design a full state observer for the  
   system.  
3 //  
4  
5  
6 //scilab -5.5.0  
7 //Operating System : OS X 10.9.3  
8  
9 //Clean the environment  
10 clc;  
11 clear all;  
12 clf;  
13  
14 //State space model
```

```

15 A=[1 -1 2; 2 -1 3; -1 -2 4];
16 B=[1 1 0]';
17 C=[1 1 0];
18 D=0;
19
20 // Stabilizer design
21 // Desired poles
22 Pd=[-7 -5 -10];
23
24 // State feedback gain matrix
25 K=ppol(A,B,Pd)
26
27 // Computation of observer gain
28 obsr_pol=[-20+0.5*i -20-0.5*i -60];
29 L=ppol(A',C',obsr_pol)'
30
31 // Augmented system
32 temp=size(A);
33 Aa=[A-B*K      B*K; zeros(temp(1),temp(2))      A-L*C
     ];
34 temp=size(Aa);
35 Ba=zeros(temp(1),1);
36 Ca=eye(6,6);
37 sys=syslin('c',Aa,Ba,Ca,zeros(6,1))
38
39 // Observer error
40 figure(0)
41 t=linspace(0,0.6,1001);
42 x0=[0 0 0 1 1 1]';
43 temp=size(t);
44 u=zeros(temp(1),temp(2)); // Exogenous signal(step)
45 y=csim(u,t,sys,x0)
46 plot(t,y(4:6,:))
47 title('Observer error','fontsize',4)
48 xlabel('$t$','fontsize',2)
49 ylabel('$x(t)-\hat{x}(t)$','fontsize',2)
50 legend('$x_1$', '$x_2$', '$x_3$')

```

---

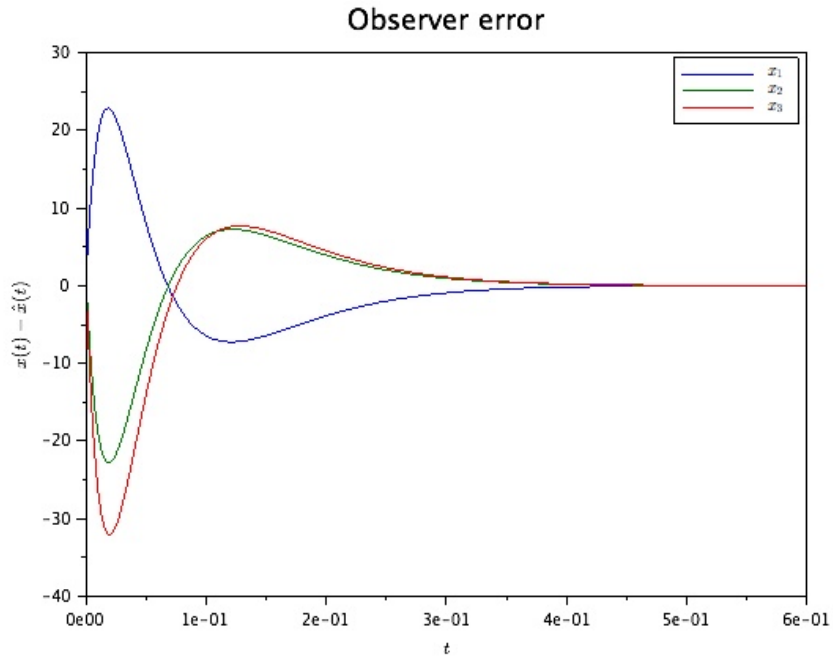


Figure 7.1: Lab 07

# Experiment: 8

## Determine steady state error of the given system.

Scilab code Solution 8.08 Lab 08

```
1 //  
2 // Lab. 08: Determine steady state error of the  
   given system.  
3 //  
4  
5 //scilab -5.5.0  
6 //Operating System : OS X 10.9.3  
7  
8 //Clean the environment  
9 clc;  
10 clear all;  
11 clf;  
12  
13 //State space model  
14 a=[0 1;-7 -9];
```

```
15 b=[0 1]';
16 c=[4 1];
17 d=0;
18 sys=syslin('c',a,b,c,d)
19
20 //Error in response of the system
21 t=linspace(0,20,1001);
22 y=csim('step',t,sys)
23 plot(t,1-y)
24 title('Error in response','fontsize',4)
25 xlabel('Time t','fontsize',2)
26 ylabel('Response y(t)','fontsize',2)
27
28 // Steady state error computation
29 ess=1+c*inv(a)*b
```

---

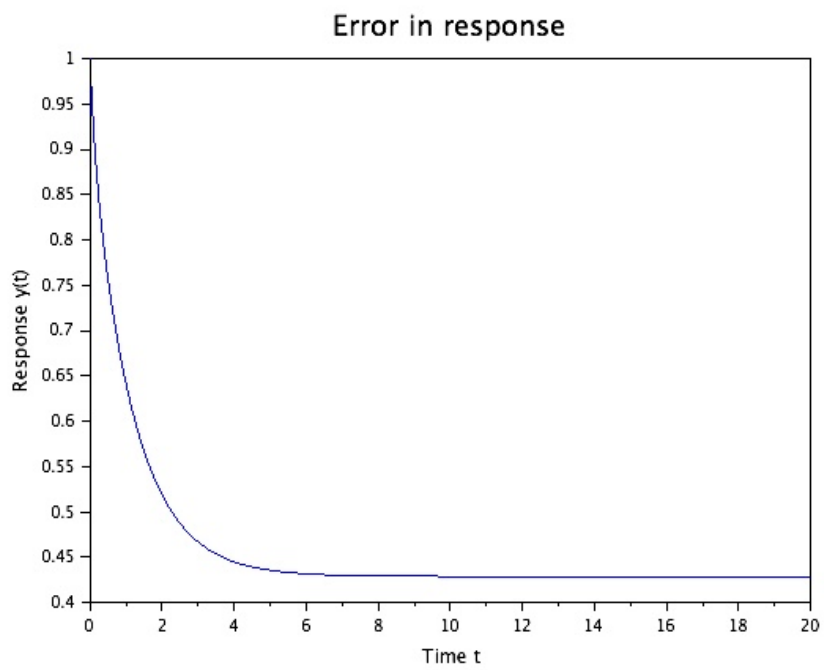


Figure 8.1: Lab 08

# Experiment: 9

## Compensation of system using lead compensator designed via root locus technique.

Scilab code Solution 9.09 Lab 09

```
1 //  
-----  
2 // Lab. 09: Compensation of system using lead  
   compensator designed via root  
3 //locus technique.  
4 //  
-----  
5  
6 //scilab -5.5.0  
7 //Operating System : OS X 10.9.3  
8  
9 //Clean the environment  
10 clc;  
11 clear all;  
12 clf;
```

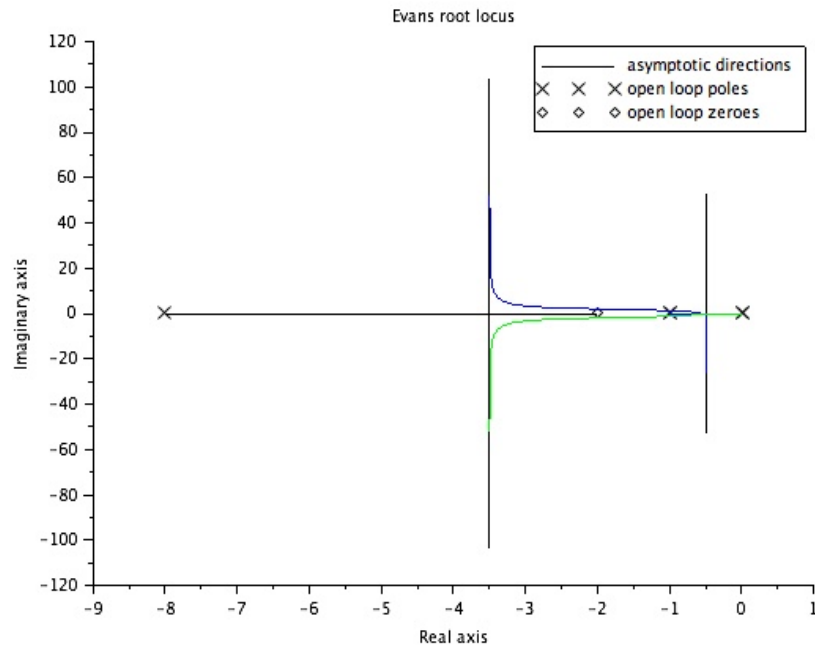


Figure 9.1: Lab 09

```

13
14 //System transfer function and its root locus
15 s=poly(0, 's ');
16 g=1/(s*(s+1));
17 evans(g)
18
19 //Designed compensator
20 gc=(s+2)/(s+8);
21
22 //Root locus of compensated system
23 evans(g*gc)

```

---

## Experiment: 10

Design a lead compensator for the given system using bode plot.

Scilab code Solution 10.10 Lab 10

```
1 //  
-----  
2 // Lab.10: Design a lead compensator for the given  
   system using bode plot.  
3 // System is  $g=K/s(s+2)$ . Design specifications:  $K_v$   
    $=20 \text{ sec}^{-1}$  and  $PM=45 \text{ deg}$ .  
4 //  
-----  
5  
6 //scilab -5.5.0  
7 //Operating System : OS X 10.9.3  
8  
9 //Clean the environment  
10 clc;  
11 clear all;
```

```

12 clf;
13
14 //Desired specifications
15 Phi_s=45;
16 K=40;
17
18 //Uncompensated system
19 s=poly(0, 's');
20 g=syslin('c', 40/(s*(s+2)));
21
22 //Bode plot of the uncompensated system
23 bode(g, 0.001, 1000)
24 title('uncompensated system')
25 gm=g_margin(g)
26 pm=p_margin(g)
27 eps1=10;
28 Phi_m=(Phi_s-pm+eps1)*%pi/180
29 alpha=(1-sin(Phi_m))/(1+sin(Phi_m))
30 gain_phi_m=-10*log10(1/alpha)
31
32 // Observed frequency at gain_phi_m
33 wc2=9.3
34
35 // Corner frequency
36 w1=wc2*sqrt(alpha)
37 w2=wc2/sqrt(alpha)
38 Gc=(s+w1)/(s+w2)
39
40 //The bode plot of compensated system
41 figure(1);
42 bode(Gc*g, 0.001, 1000),
43 title('Compensated system')

```

---

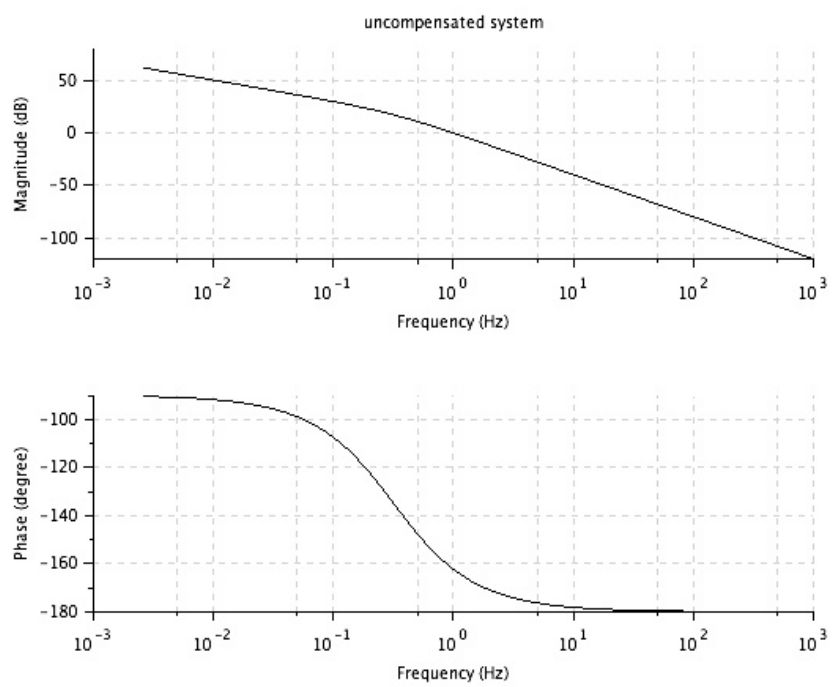


Figure 10.1: Lab 10

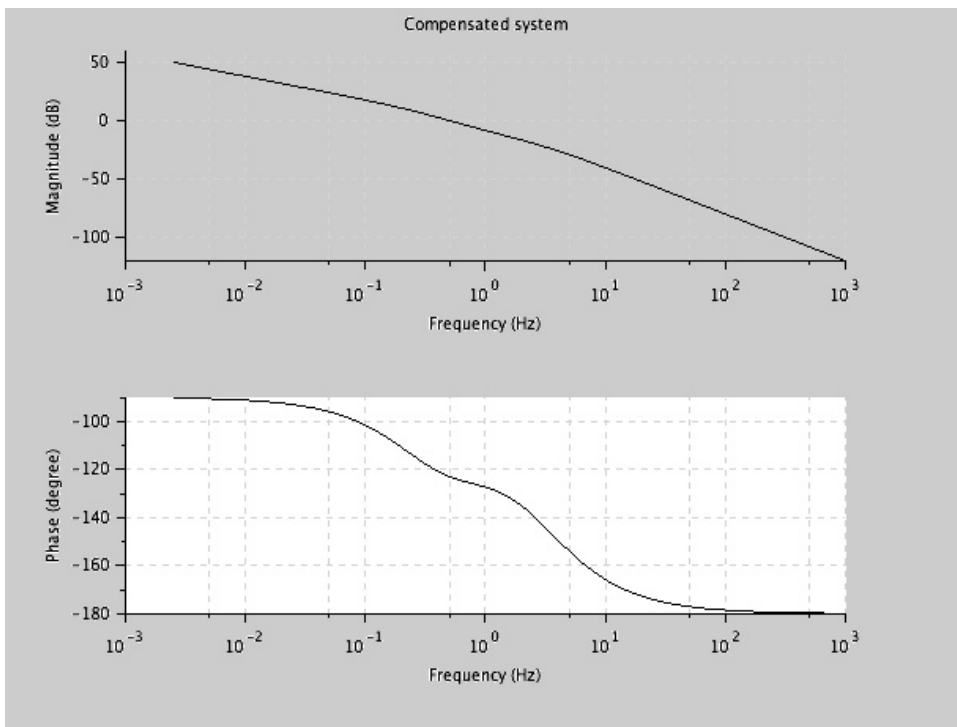


Figure 10.2: Lab 10