Scilab Manual for
Control System Design
by Prof Deepti Khimani
Instrumentation Engineering
VESIT[1]

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

February 20, 2021

[1]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 Experiments

List of Figures

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

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

//scilab-5.5.0
//Operating System : OS X 10.9.3

//Clean the environment
clc;
```

```
clear all;
clf;

// Compensator model
R1=1000; R2=5e3;C1=1e-6;C2=1e-5;
kc=5;
s=poly(0,'s');
g=kc*(R1*C1*s+1)/(R2*C2*s+1);

// System transfer function
g1=0.2/(s^2+1.7*s+1);

// Overall transfer function
sys=tf2ss(g*g1);

// Unit step response
t=linspace(0,10,1000);
y=csim('step',t,sys);
plot(t,y);
title('Unit step response of the electrical system',
    'fontsize',4)
xlabel('Time t','fontsize',2)
ylabel('Response y(t)','fontsize',2)
//set(gca(),"grid",[0.3 0.3])
```

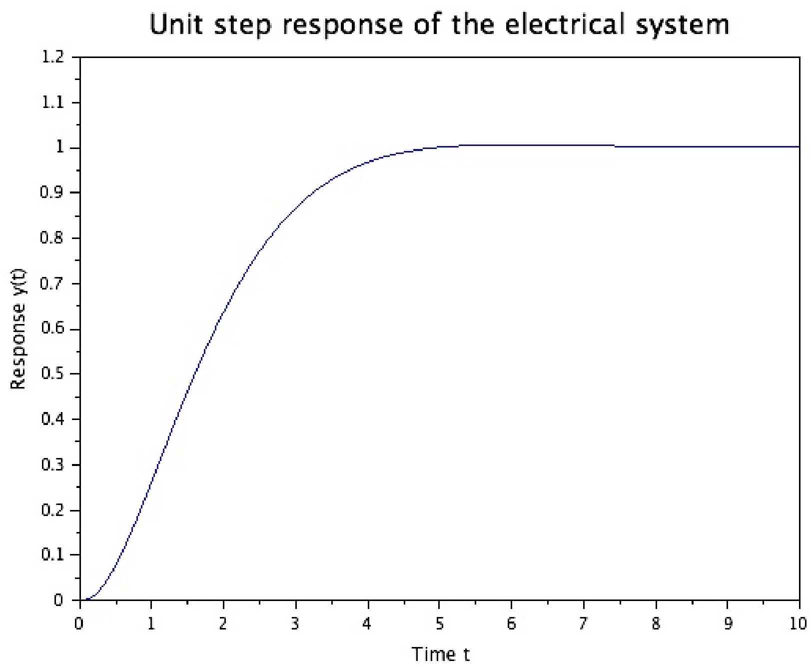

Figure 1.1: Lab 01

Experiment: 2

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

Scilab code Solution 2.02 Lab02

```

//
    ________________________________________________________________

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


//scilab -5.5.0
//Operating System : OS X 10.9.3

//Clean the environment
```

```
clc;
clear all;
//clf;

// State space representation
A=[0 1 0; 0 0 1; -5 -25 -5];
B=[0; 25;-120];
C=[1 0 0];
D=0;

sys1=syslin('c',A,B,C,D);
mprintf('State space representation of the given system is')
disp(sys1)

// Eigen values of system matrix
eig_val=spec(A)
mprintf('Eigen values of the system matrix are')
disp(eig_val)

// Transfer function of the given system
g1=ss2tf(sys1)
mprintf('Transfer function representation of the given system is')
disp(g1)
```

Experiment: 3

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

Scilab code Solution 3.03 Lab3

```

//
    ____________________________________________________________________

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


//scilab -5.5.0
//Operating System : OS X 10.9.3

//Clean the environment
```

```
clc;
clear all;
//clf;

// State space model
A=[-3 1; 1 -3];
B=[1;2];
C=[2 3];
D=0;

sys=syslin('c',A,B,C,D)
mprintf('State space representation of the given system is')
disp(sys)


// Eigen values of system matrix
eig_val=spec(A)
mprintf('Eigen values of the system matrix are')
disp(eig_val)

// Controllable canonical form
[Ac, Bc T]=canon(A,B)
T=flipdim(T,2);
Ac=T\A*T;
Bc=T\B;
Cc=C*T;
Dc=D;
sysc=syslin('c',Ac,Bc,Cc,Dc)
mprintf('State space representation of the given system in Controllable canonical form is')
disp(sysc)

// Diagonal form
[Ad M]=bdiag(A);
Bd=M\B;
Cd=C*M;
Dd=D;
```

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

Experiment: 4

Obtain the step and impulse response of the state model.

Scilab code Solution 4.04 Lab 04

```
//
   ____________________________________________________________________
// Lab. 04: Obtain the step and impulse response of
   the state model.
//
   ____________________________________________________________________

//scilab -5.5.0
//Operating System : OS X 10.9.3

//Clean the environment
clc;
clear all;
clf;

// State space representation
A=[-2 -1; -1 -1];
```

```
B=[1;1];
C=[0 2];
D=0;
x0=[0;5]; // Initial condition
sys=syslin('c',A,B,C,D)

// Response to a given input
figure(0)
t=linspace(0,20,1001);
temp=size(t);
u=ones(temp(1),temp(2)); // Exogenous signal(step)
y=csim(u,t,sys,x0)
plot(t,y)
title('Unit step response of the system','fontsize',4)
xlabel('Time t','fontsize',2)
ylabel('Response y(t)','fontsize',2)

// Response to a given input
figure(1)
t=linspace(0,10,1001);
y=csim('impuls',t,sys)
plot(t,y)
title('Impulse response of the system','fontsize',4)
xlabel('Time t','fontsize',2)
ylabel('Response y(t)','fontsize',2)
```

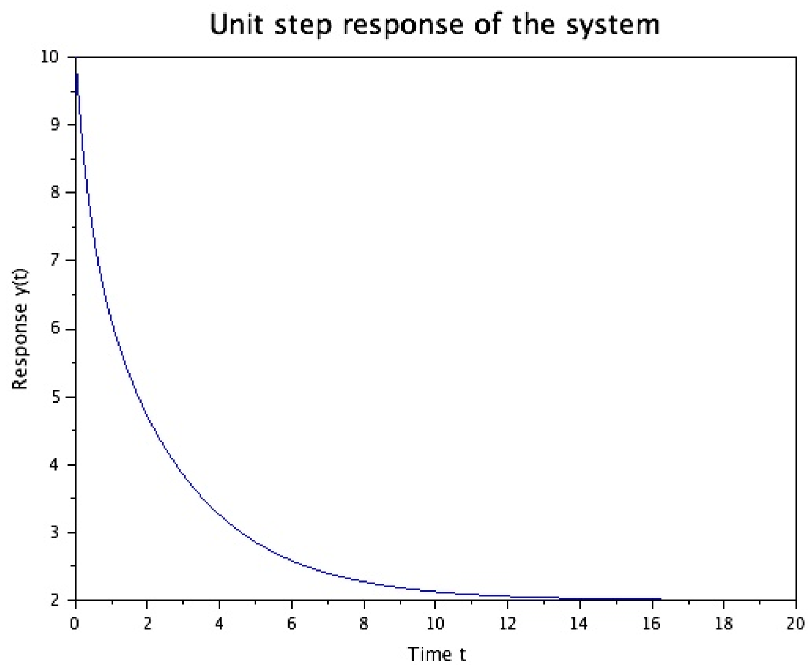

Figure 4.1: Lab 04

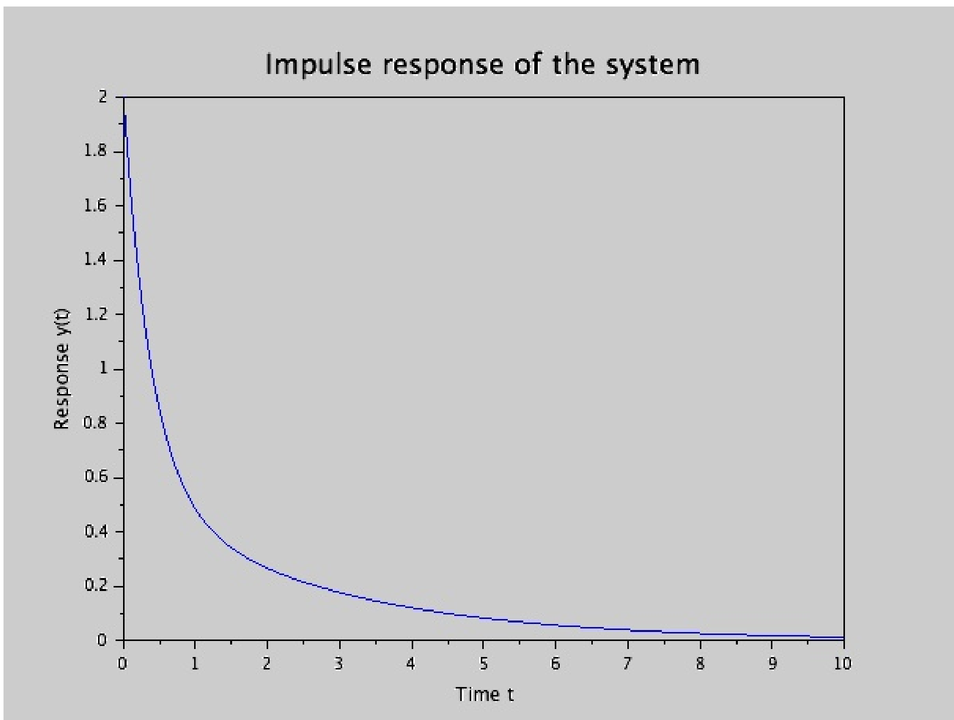

Figure 4.2: Lab 04

Experiment: 5

Check for the controllability and observability of a given system.

Scilab code Solution 5.05 Lab05

```

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

//scilab-5.5.0
//Operating System : OS X 10.9.3

//Clean the environment
clc;
clear all;
//clf;
```

```
// State space representation
A=[-5 1 0; 0 -2 1; 0 0 -1];
B=[6 0 1]';
C=[1 0 0];
D=0;
sys=syslin('c',A,B,C,D)

// Controllability test
n=cont_mat(sys)
mprintf('Controllability matrix is')
disp(n)

if rank(n)==3 then
    disp('System is controllable')
else
    disp('System is uncontrollable')
end

// Observability test
m=obsv_mat(sys)
mprintf('Observability matrix is')
disp(m)

if rank(m)==3 then
    disp('System is observable')
else
    disp('System is unobservable')
end
```

Experiment: 6

Obtain state feedback gain matrix for the given system.

Scilab code Solution 6.06 Lab 06

```
//
    ____________________________________________________________
// Lab. 06: Obtain state feedback gain matrix for
   the given system.
//
    ____________________________________________________________

//scilab -5.5.0
//Operating System : OS X 10.9.3

//Clean the environment
clc;
clear all;
clf;

// State space representation
A=[0 1 0; 0 0 1; -1 -5 -6];
```

```
B=[0 0 1]';
C=[0 0 1];
D=0;

// Desired poles
Pd=[-1+2*%i -1-2*%i -10];

// State feedback gain matrix
K=ppol(A,B,Pd)

//Closed loop system
sys=syslin('c',A-B*K,B,C,D)

//Response of closed loop system
t=linspace(0,20,1001);
y=csim('step',t,sys)
plot(t,y)
title('Response of the closed loop system','fontsize',4)
xlabel('Time t','fontsize',2)
ylabel('Response y(t)','fontsize',2)
```

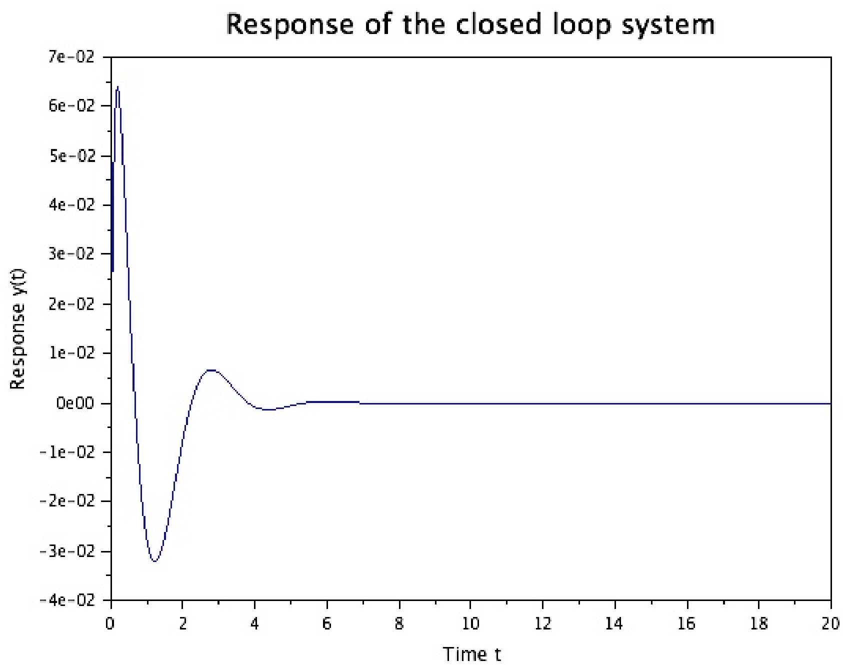

Figure 6.1: Lab 06

Experiment: 7

Design a full state observer for the system.

Scilab code Solution 7.07 Lab 07

```
//
    ____________________________________________________________________

// Lab. 07: Design a full state observer for the
    system.
//
    ____________________________________________________________________



//scilab-5.5.0
//Operating System : OS X 10.9.3

//Clean the environment
clc;
clear all;
clf;

//State space model
```

```
A=[1 -1 2; 2 -1 3; -1 -2 4];
B=[1 1 0]';
C=[1 1 0];
D=0;

//Stabilizer design
// Desired poles
Pd=[-7 -5 -10];

// State feedback gain matrix
K=ppol(A,B,Pd)

//Computation of observer gain
obsr_pol=[-20+0.5*%i -20-0.5*%i -60];
L=ppol(A',C',obsr_pol)'

// Augmented system
temp=size(A);
Aa=[A-B*K    B*K; zeros(temp(1),temp(2))     A-L*C ];
temp=size(Aa);
Ba=zeros(temp(1),1);
Ca=eye(6,6);
sys=syslin('c',Aa,Ba,Ca,zeros(6,1))

//Observer error
figure(0)
t=linspace(0,0.6,1001);
x0=[0 0 0 1 1 1 ]';
temp=size(t);
u=zeros(temp(1),temp(2)); // Exogenous signal(step)
y=csim(u,t,sys,x0)
plot(t,y(4:6,:))
title('Observer error','fontsize',4)
xlabel('$t$','fontsize',2)
ylabel('$x(t)-\hat x(t)$','fontsize',2)
legend('$x_1$','$x_2$','$x_3$')
```

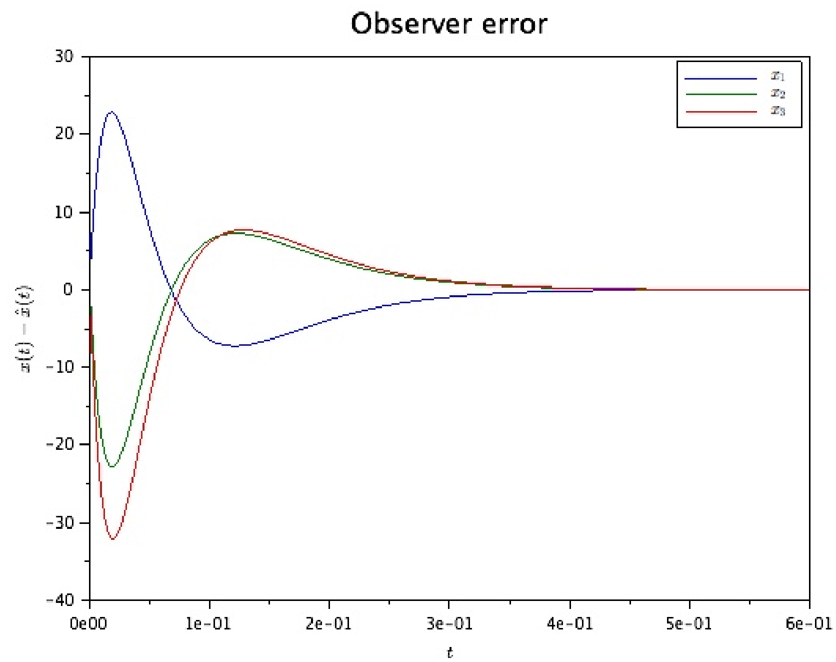

Figure 7.1: Lab 07

Experiment: 8

Determine steady state error of the given system.

Scilab code Solution 8.08 Lab 08

```
//______________________________________________________________________
// Lab. 08: Determine steady state error of the given system.
//______________________________________________________________________

//scilab-5.5.0
//Operating System : OS X 10.9.3

//Clean the environment
clc;
clear all;
clf;

//State space model
a=[0 1;-7 -9];
```

```
b=[0 1]';
c=[4 1];
d=0;
sys=syslin('c',a,b,c,d)

//Error in response of the system
t=linspace(0,20,1001);
y=csim('step',t,sys)
plot(t,1-y)
title('Error in response','fontsize',4)
xlabel('Time t','fontsize',2)
ylabel('Response y(t)','fontsize',2)

// Steady state error computation
ess=1+c*inv(a)*b
```

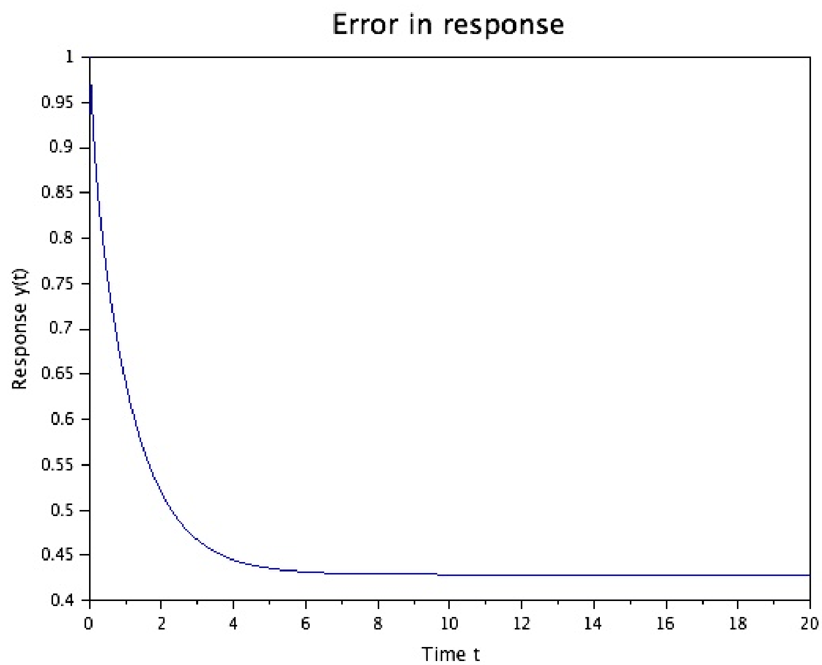

Figure 8.1: Lab 08

Experiment: 9

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

Scilab code Solution 9.09 Lab 09

```
//
    ____________________________________________________________________

// Lab. 09: Compensation of system using lead
    compensator designed via root
//locus technique.
//
    ____________________________________________________________________


//scilab -5.5.0
//Operating System : OS X 10.9.3

//Clean the environment
clc;
clear all;
clf;
```

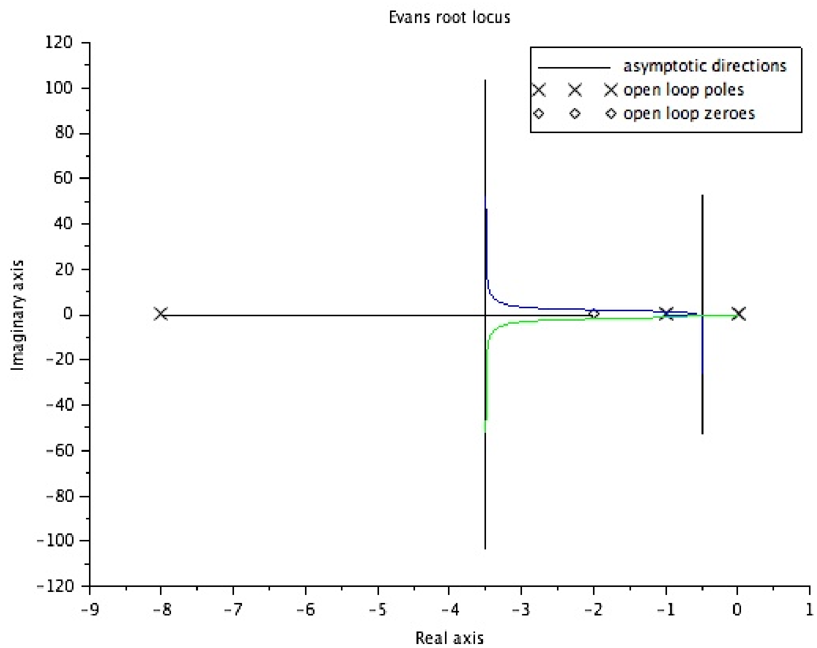

Figure 9.1: Lab 09

```

//System transfer function and its root locus
s=poly(0,'s');
g=1/(s*(s+1));
evans(g)

//Designed compensator
gc=(s+2)/(s+8);

//Root locus of compensated system
evans(g*gc)
```

Experiment: 10

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

Scilab code Solution 10.10 Lab 10

```
//
    ____________________________________________________________________________

// Lab.10: Design a lead compensator for the given
    system using bode plot.
// System is g=K/s(s+2). Design specifications: Kv
    =20 sec^-1 and PM=45 deg.
//
    ____________________________________________________________________________


//scilab-5.5.0
//Operating System : OS X 10.9.3

//Clean the environment
clc;
clear all;
```

```
clf;

//Desired specifications
Phi_s=45;
K=40;

//Uncompenstated system
s=poly(0,'s');
g=syslin('c',40/(s*(s+2)));

//Bode plot of the uncompenmsated system
bode(g,0.001,1000)
title('uncompensated system')
gm=g_margin(g)
pm=p_margin(g)
eps1=10;
Phi_m=(Phi_s-pm+eps1)*%pi/180
alpha=(1-sin(Phi_m))/(1+sin(Phi_m))
gain_phi_m=-10*log10(1/alpha)

// Observed frequency at gain_phi_m
wc2=9.3

// Corner frequency
w1=wc2*sqrt(alpha)
w2=wc2/sqrt(alpha)
Gc=(s+w1)/(s+w2)

//The bode plot of compensated system
figure(1);
bode(Gc*g,0.001,1000),
title('Compensated system')
```

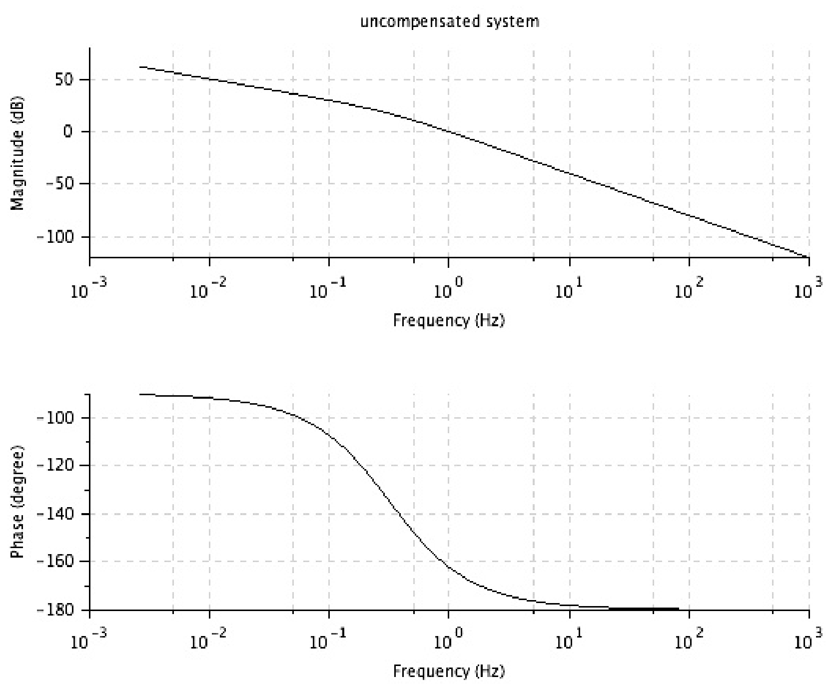

Figure 10.1: Lab 10

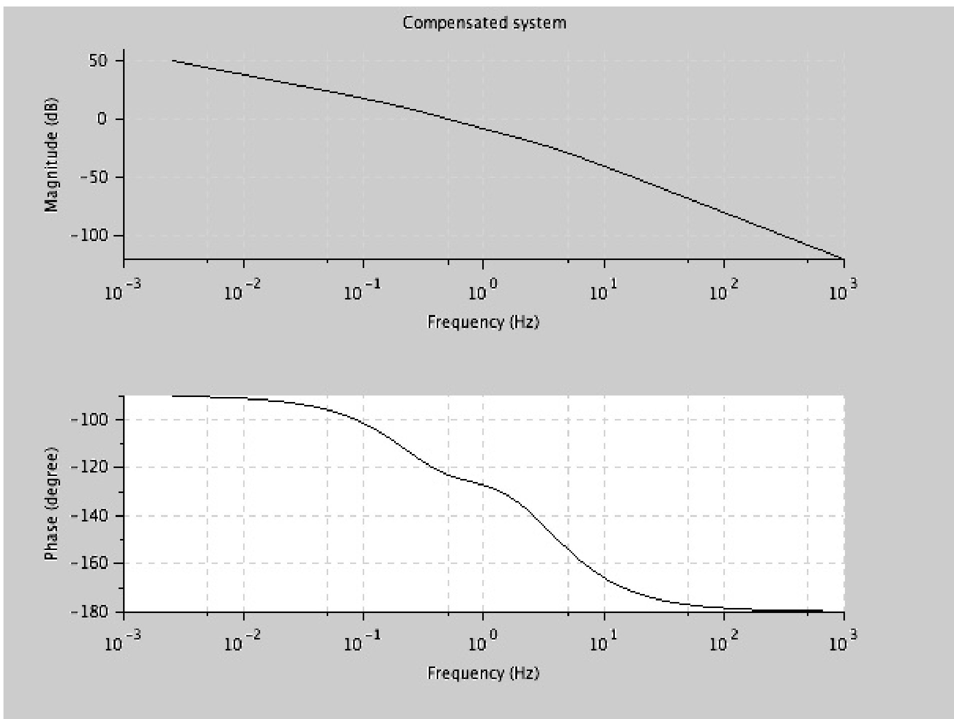

Figure 10.2: Lab 10