# Scilab Manual for Control System Design by Prof Dilip Pawar Instrumentation Engineering Bharati Vidyapeeth College Of Engineering, Navi Mumbai[1]

Solutions provided by
Mr Dilip N. Pawar
Instrumentation Engineering
Bharati Vidyapeeth College Of Engineering, Navi Mumbai

April 24, 2024

[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

# State model for SISO and MIMO Systems

**Scilab code Solution 1.1** a

```
// Scilab : 6.0.0
// OS: Windows 7, 64 bit
// Lab 1 a): To obtain State - Space Model of given
   system

clc ;
clear all;


// Define the Numerator and Denominator Polynomials
   of Transfer Function, G(s)
num=poly([30 10],"s","coeff");    //Defines the
   numerator of G(s)
den=poly([0 8 6 1],"s","coeff"); //Defines the
   denominator of G(s)

// Obtain the Contollable Phase Variable Form
sys=cont_frm(num,den);
[A,B,C,D]=abcd(sys)
```

```
disp(D,"D",C,"C",B,"B",A,"A")

// Result

// A

//     0.     1.     0.
//     0.     0.     1.
//     0.   -8.   -6.

// B

//   0.
//   0.
//   1.

// C

//     30.     10.     0.

// D

//   0.
```

# Experiment: 2

# Similarity Transformation

**Scilab code Solution 2.2** a

```
// Scilab : 6.0.0
// OS: Windows 7, 64 bit
// Lab 2 a): To obtain Similar State - Space Model of given system

clc ;
clear all;

// A system having the form xdot=Ax+Bu and y=Cx+du, where
A=[-2,1;-3,0]; B=[4;5]; C=[1,0]; D=0;//and
//The transformation Matrix P is taken as -
P = [2,1;4,3];
//the given system can be transformed to a similar system as -
AA=inv(P)*A*P
BB=inv(P)*B
CC=C*P
disp(C,"CC",B,"BB", A,"AA")
// Check the eigenvalues of both systems
EV1=spec(A); EV2=spec(AA);
```

```
disp(EV2,"Eigenvalues of System2", EV1,"Eigenvalues
    of System1")
// If both eigenvalues are same then systems are
    similar

// Result

// AA
//
//   -2.    1.
//   -3.    0.
//
// BB
//
//    4.
//    5.
//
// CC
//
//    1.    0.
//
// Eigenvalues of System1
//
//   -1. + 1.4142136i
//   -1. - 1.4142136i
//
// Eigenvalues of System2
//
//   -1. + 1.4142136i
//   -1. - 1.4142136i
//
```

# Experiment: 3

# Controllability and Observability

**Scilab code Solution 3.3** a

```
// Scilab : 6.0.0
// OS: Windows 7, 64 bit
// Lab 3 a): To Check the Controllability of given
   system by Gilbert's Test

clc ;
clear all;

// Let the system is described by state equations
A=[0,1,0;0,0,1;-6,-11,-6];
B=[0;0;1];
// Solution: -
// Gilbert's Test is required to find canonical
   state variable form as -
// Given system is in Phase Variable form, therefore
    Vander Monde Matrix is
// required as an Modal Matrix
// Find the eigenvalues and eigenvectors of matrix A
[V,D]=spec(A);
```

```
// Forming the Vander Monde Matrix from the
   eigenvalues as -
d = diag(D); dd=(d.^2);
M=[1,1,1;d';dd'];
Minv=inv(M);
// Finding the Bcap
Bcap=Minv*B;
disp(Bcap,"Bcap")
// Check the contents of B vector, whether any
   element is zero,
// if not then system is controllable
// Result

// Bcap
//
//    0.5
//   -1.
//    0.5
```

# Experiment: 4

# Pole-Placement with State Feedback

**Scilab code Solution 4.4** a

```
// Scilab : 6.0.0
// OS: Windows 7, 64 bit
// Lab 4 a): To find a gain matrix K, by usinf Pole-placement by state feedback

clc ;
clear all;

// Problem: Consider the system as
//   .
//  X = AX + Bu
// where A = [0,1,0;0,0,1;-1,-5,-6]; B = [0;0;1];
// By using state feedback control u= - Kx, it is desired to have the closed
// - loop poles at s1 = -2+j*4; s2 = -2-j*4; and s3 = -10;
// Determine the state feedback-gain matrix K with SCILAB

```

```
// First define the given A, B matrices as -
A = [0,1,0;0,0,1;-1,-5,-6];
B = [0;0;1];
// Then define the closed - loop poles as -
s1=-2+%i*4; s2=-2-%i*4; s3=-10;
S=[s1,s2,s3];
// Then invoking commands 'ppol' as -
K=ppol(A,B,S)
disp(K,"K")

// Result

// K
//
//    199.    55.    8.
```

# Experiment: 5

# Lead Compensator Design by Root – Locus

**Scilab code Solution 5.5** To design a Lead Compensator by Root Locus technique

```
// Scilab : 6.0.0
// OS: Windows 7, 64 bit
// Lab 5: To design a Lead Compensator by Root-Locus
    technique

clc ;
clear all;



s=%s;
G=syslin('c',1/s^2)      // Transfer Function of Un-
   compensated System
Gc=syslin('c',(s+1.15)/((s^2)*(s+4.2))) // Transfer
   Function of Compensated System
// Root - Locus of Un-Compensated System
figure(0)
clf, evans(G), xgrid(5, 1, 7)
```

```

// Root - Locus of Compensated System
figure(1)
clf, evans(Gc), xgrid(5, 1, 7)
replot([-4.5,-10,0.5,10])

// Step Response of Un-Compensated System
figure(2)
t= 0:0.1:100;
x=[csim('step',t,G/(1+G),[0;0;0])]';
plot2d(t',x),
xlabel("t", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
xgrid(5, 1, 7)
xtitle( 'Step Response of Un-Compensated System');//, 'X axis', 'Y axis');
// Step Response of Compensated System
figure(3)
t= 0:0.1:100;
x=[csim('step',t,Gc/(1+Gc),[0;0;0])]';
plot2d(t',x),
xlabel("t", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
xtitle( 'Step Response of Compensated System');//, 'X axis', 'Y axis');
xgrid(5, 1, 7)
```

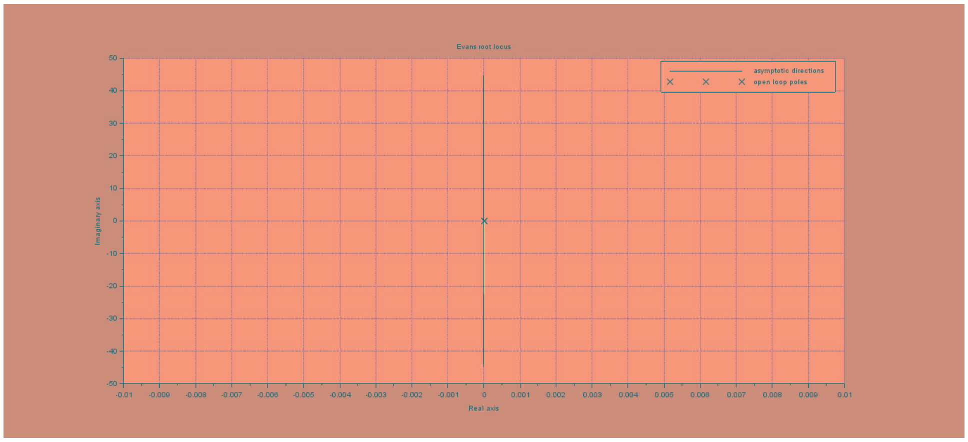


Figure 5.1: To design a Lead Compensator by Root Locus technique

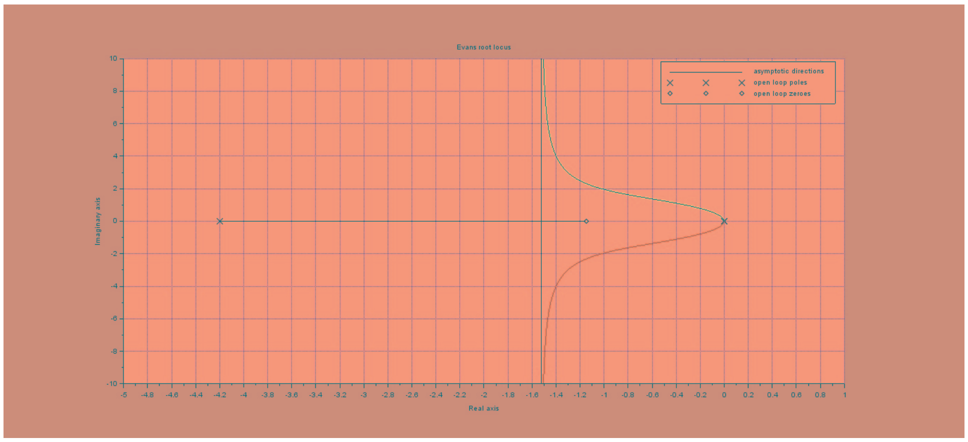


Figure 5.2: To design a Lead Compensator by Root Locus technique

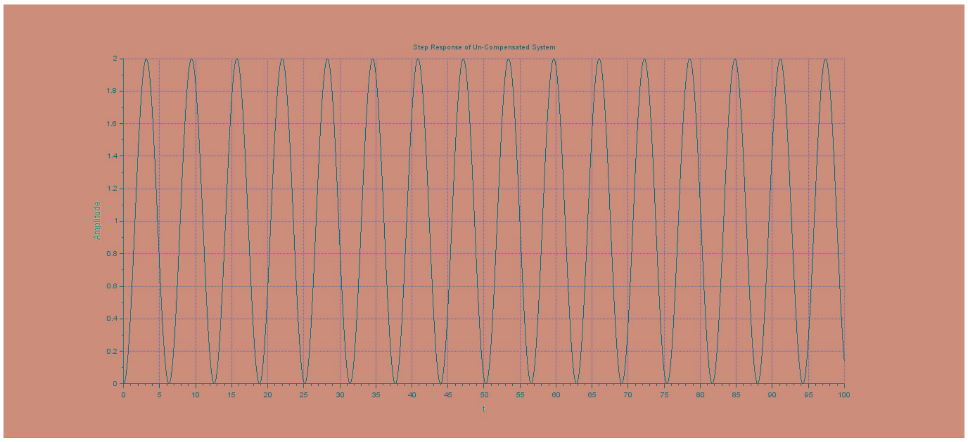


Figure 5.3: To design a Lead Compensator by Root Locus technique

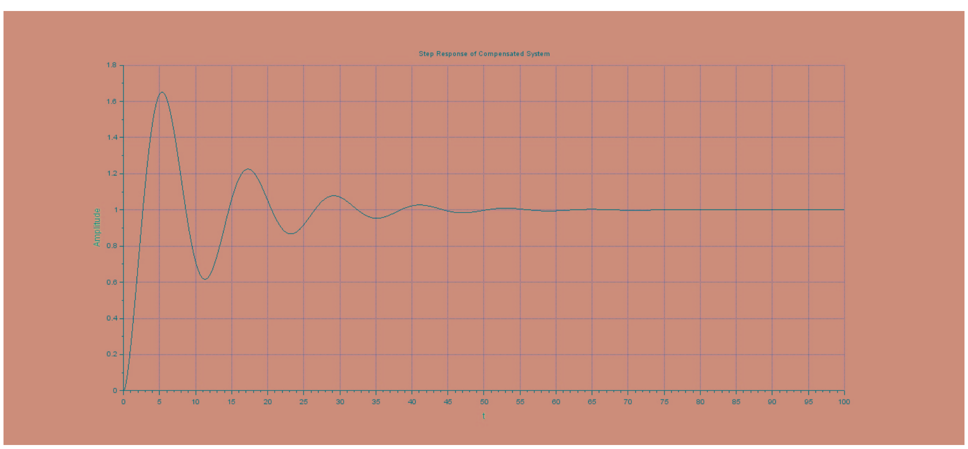


Figure 5.4: To design a Lead Compensator by Root Locus technique

# Experiment: 6

# Lag Compensator Design by Root – Locus

**Scilab code Solution 6.6** To design a Lag Compensator by Root Locus technique

```
// Scilab : 6.0.0
// OS: Windows 7, 64 bit
// Lab 6: To design a Lag Compensator by Root−Locus
   technique

clc ;
clear all;

s=%s;
G=syslin('c',1/(s*(s+1)*(s+4))) // Transfer Function
    of Un−Compensated System
Gc=syslin('c',(s+0.2)/((s+0.02)*s*(s+1)*(s+4))) //
   Transfer Function of Compensated System
// Root − Locus of Un−Compensated System
figure(0)
clf, evans(G), xgrid//(5, 1, 7)
replot([-4.5,-10,0.5,10])
// Step Response of Un−Compensated System
```

```
figure(1)
t= 0:0.1:100;
x=[csim('step',t,G/(1+G),[0;0;0])]';
plot2d(t',x),xgrid//(5, 1, 7)
xlabel("t", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
xtitle( 'Step Response of Un-Compensated System');

// Root - Locus of Compensated System
figure(2)
clf, evans(Gc), xgrid//(5, 1, 7)

replot([-4.5,-6,0.5,6])
// Step Response of Compensated System
figure(3)
t= 0:0.1:100;
x=[csim('step',t,Gc/(1+Gc))]';
plot2d(t',x),xgrid//(5, 1, 7)
xlabel("t", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
xtitle( 'Step Response of Compensated System');
```

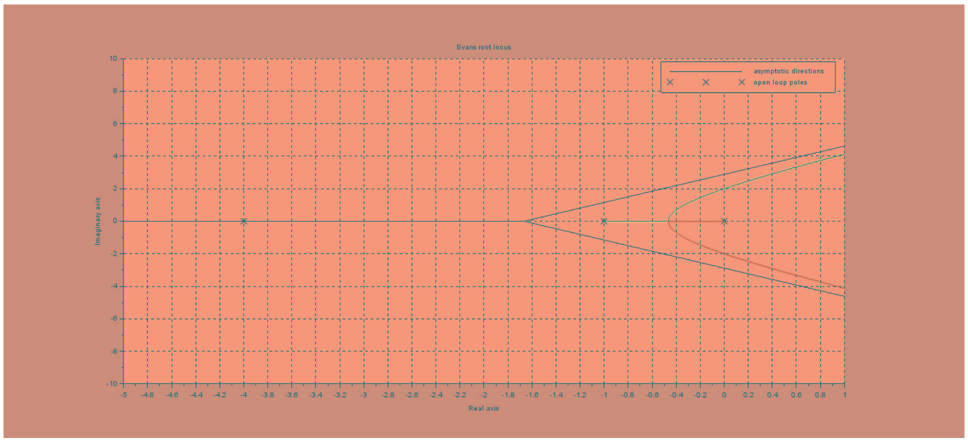


Figure 6.1: To design a Lag Compensator by Root Locus technique

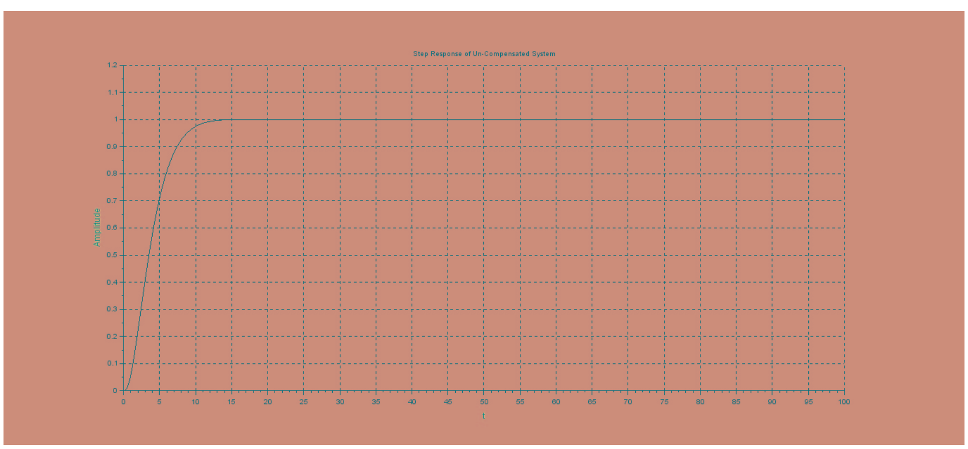


Figure 6.2: To design a Lag Compensator by Root Locus technique

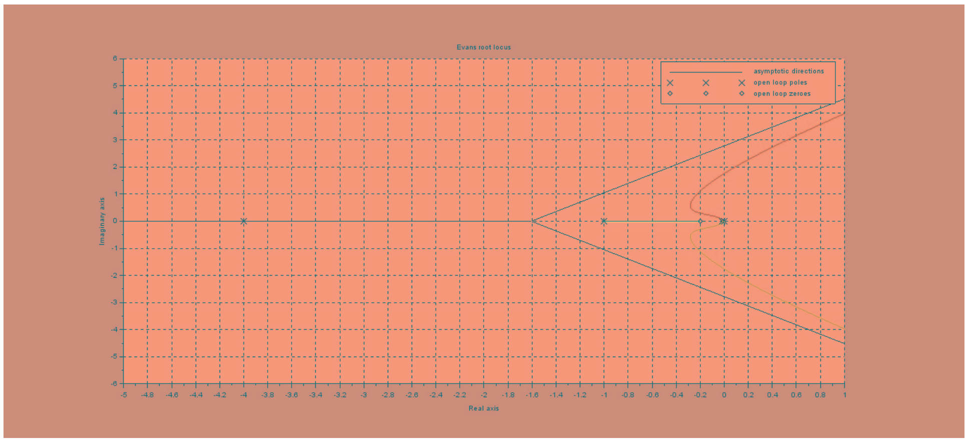


Figure 6.3: To design a Lag Compensator by Root Locus technique

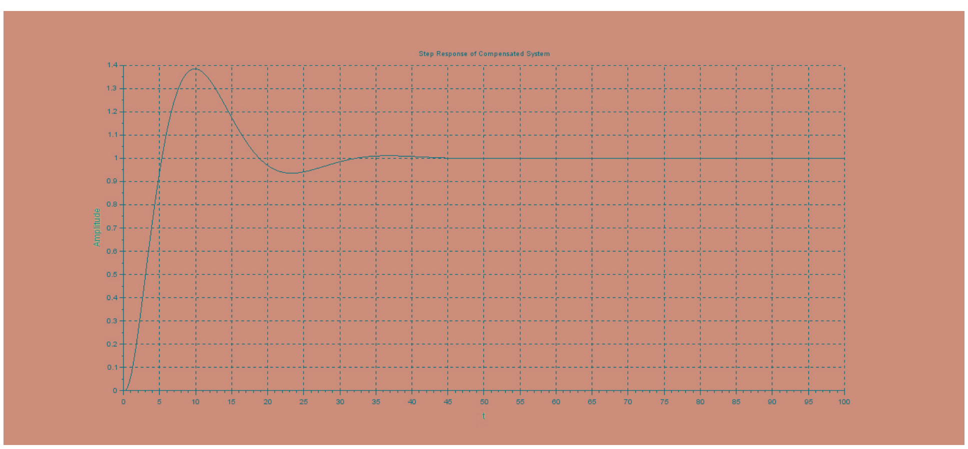


Figure 6.4: To design a Lag Compensator by Root Locus technique

# Experiment: 7

# Time Delay or Transportation Lag

**Scilab code Solution 7.7** To Study the Effect of Time Delay on System Response

```
// Scilab : 6.0.0
// OS: Windows 7, 64 bit
// Lab 7: To Study the Effect of Time Delay on
   System Response
clc ;
clear all;

// For T = 0, i.e. No Delay
s=poly(0,'s')
// Transfer Function of Open-Loop System with T=0
G1=syslin('c',exp(0)*10/((s)*(s+1)));
figure(0)
bode(G1,0.1,100)
[gm1,frg1]=g_margin(G1);
disp(gm1,"Gain Margin,T=0",frg1,"Phase Cut-off
   Frequency, T=0")
show_margins(G1)
[phm1,frp1]=p_margin(G1);
```

```
disp(phm1,"Phase Margin, T=0",frp1,"Gain Cut-off Frequency, T=0")
show_margins(G1)

// For Time Delay T = 0.1
T=0.1;
nTd=[1-T*s/2]; dTd=[1+T*s/2];// Num and Den of Delay System by Pade
sysTd=syslin('c',nTd/dTd);
G2=syslin('c',(sysTd*10)/((s)*(s+1)));
figure(1)
bode(G2,0.1,100)
[gm2,frg2]=g_margin(G2);
disp(gm2,"Gain Margin, T=0.1",frg2,"Phase Cut-off Frequency, T=0.1")
show_margins(G2)
[phm2,frp2]=p_margin(G2);
disp(phm2,"Phase Margin, T=0.1",frp2,"Gain Cut-off Frequency, T=0.1")
show_margins(G2)

// Results
// Frequency Response Specifications of Un-Compensated System
//Phase Cut-off Frequency, T=0
//
//     []
//
// Gain Margin,T=0
//
//    Inf
//
// Gain Cut-off Frequency, T=0
//
//    0.4908709
//
// Phase Margin, T=0
```

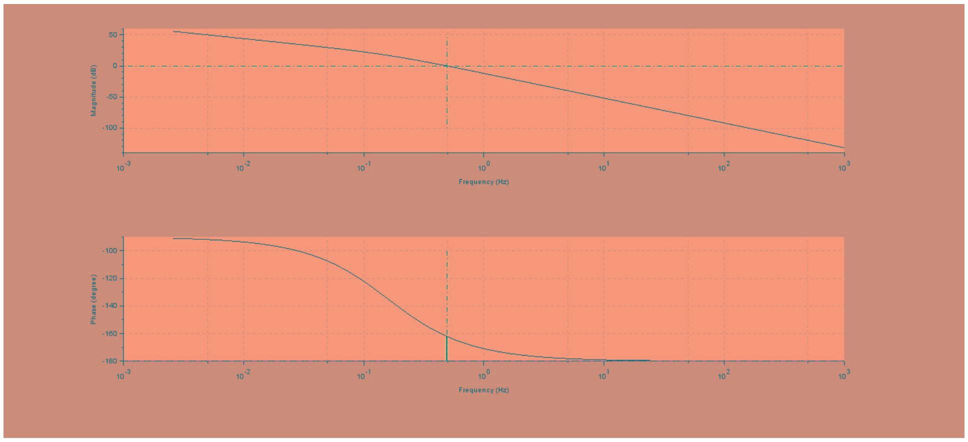

Figure 7.1: To Study the Effect of Time Delay on System Response

```
//    17.964236
// Frequency Response Specifications of Compensated
    System

// Phase Cut-off Frequency , T=0.1
//
//    0.4971165
//
// Gain Margin , T=0.1
//
//    0.2093087
//
// Gain Cut-off Frequency , T=0.1
//
//    0.4908709
//
// Phase Margin , T=0.1

//    0.4310002
```

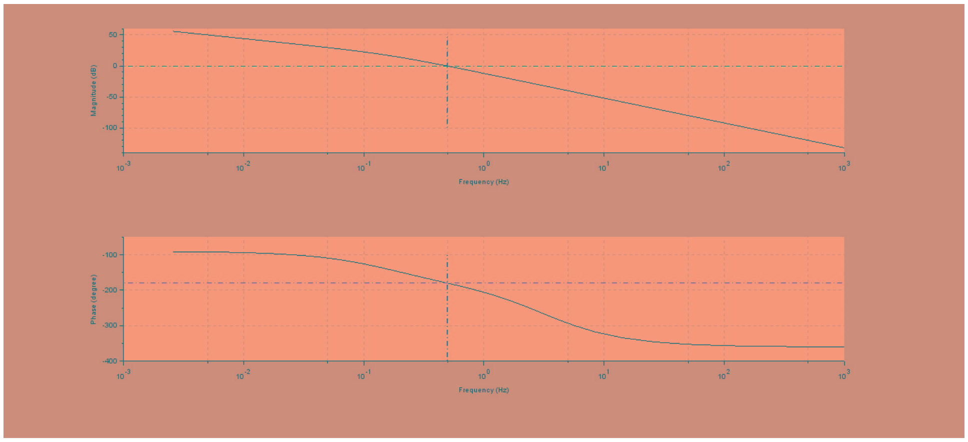


Figure 7.2: To Study the Effect of Time Delay on System Response

# Experiment: 8

# Lead Compensator Design by Bode Plot Technique

**Scilab code Solution 8.8** To design a Lead Comensator by Bode plot technique

```
// Scilab : 6.0.0
// OS: Windows 7, 64 bit
// Expt. No. 8: To design a Lead Comensator by Bode-plot technique

clc
close
clf();

s=poly(0,'s')
// Transfer Function of Open-Loop System
G=syslin('c',12/((s)*(s+1)))
figure(0)
bode(G,0.1,100)
[gm,frg]=g_margin(G);
disp(gm,"Gain Margin",frg,"Phase Cut-off Frequency")
show_margins(G)
[phm,frp]=p_margin(G);
```

```
disp(phm,"Phase Margin",frp,"Gain Cut-off Frequency")
show_margins(G)

// Transfer Function of Open-Loop Compensated System , GGc
GGc=syslin('c',12*(1+0.385*s)/((s)*(s+1)*(1+0.125*s)))

// Bode - Plot of Compensated System
figure(1)
bode(GGc,0.1,100)
[gm,frg]=g_margin(GGc);
disp(gm,"Gain Margin",frg,"Phase Cut-off Frequency")
show_margins(GGc)
[phm,frp]=p_margin(GGc);
disp(phm,"Phase Margin",frp,"Gain Cut-off Frequency")
show_margins(GGc)

// Step Response of Coles-Loop Un-Compensated System
figure(2)
t= 0:0.1:10;
x=[csim('step',t,G/(1+G))]';
plot2d(t',x),xgrid(5, 1, 7)
xlabel("t", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
xtitle( 'Step Response of Un-Compensated System');

// Step Response of Close-Loop Compensated System
figure(3)
t= 0:0.1:10;
x=[csim('step',t,GGc/(1+GGc))]';
plot2d(t',x),xgrid(5, 1, 7)
xlabel("t", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
xtitle( 'Step Response of Compensated System');
// Results
```

```
// Frequency Response Specifications of Un-Compensated System
//Phase Cut-off Frequency
//
//     []
//
// Gain Margin
//
//    Inf
//
// Gain Cut-off Frequency
//
//    0.5399649
//
// Phase Margin
//
//    16.422908

// Frequency Response Specifications of Compensated System

// Phase Cut-off Frequency
//
//     []
//
// Gain Margin
//
//    Inf
//
// Gain Cut-off Frequency
//
//    0.720447
//
// Phase Margin
//
//    43.107374
```

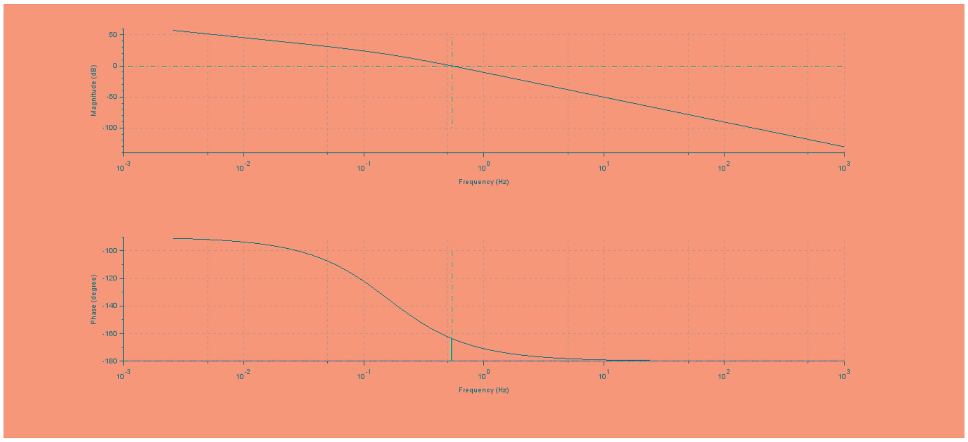


Figure 8.1: To design a Lead Comensator by Bode plot technique

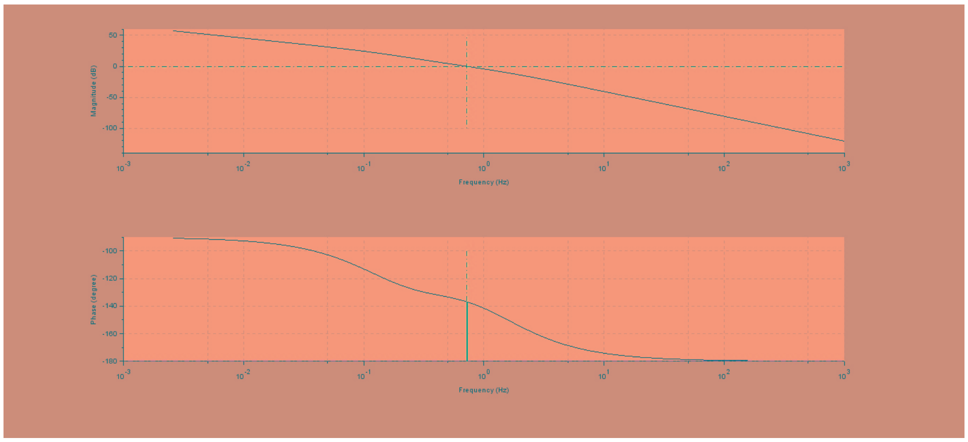


Figure 8.2: To design a Lead Comensator by Bode plot technique

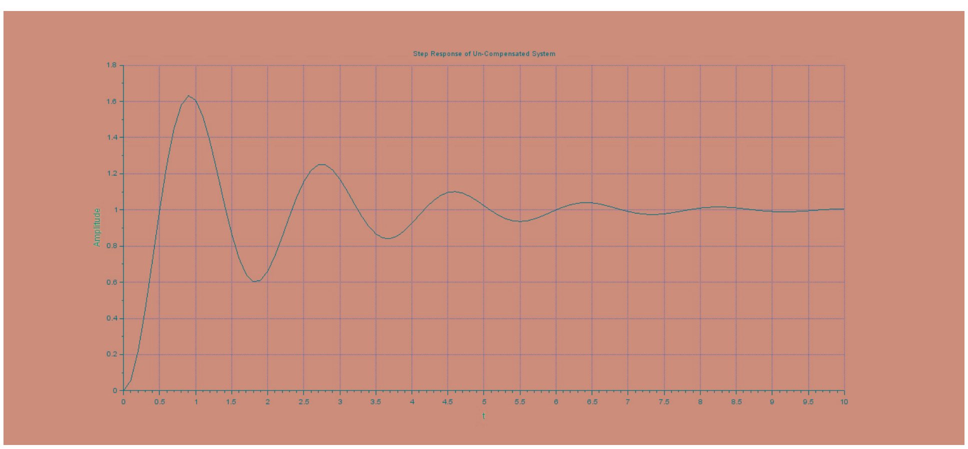


Figure 8.3: To design a Lead Comensator by Bode plot technique

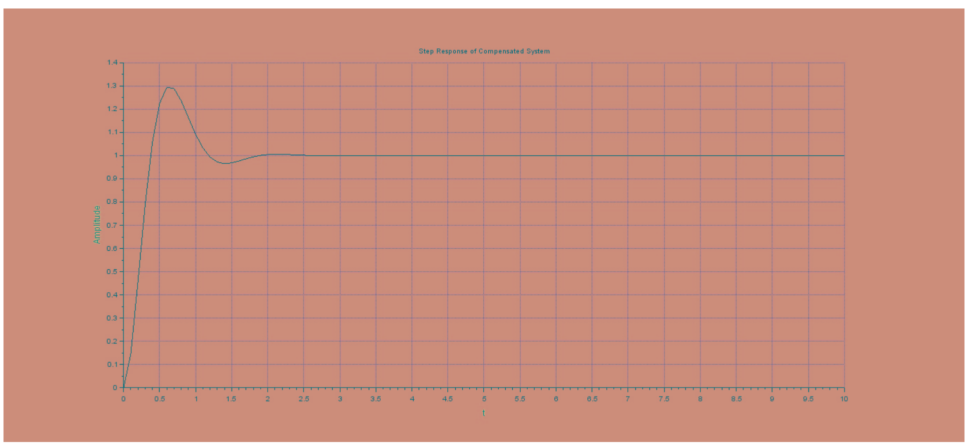


Figure 8.4: To design a Lead Comensator by Bode plot technique

# Experiment: 9

# Lag Compensator Design by Bode Plot Technique

**Scilab code Solution 9.9** To design a Lag Compensator by Bode plot technique

```
// Scilab : 6.0.0
// OS: Windows 7, 64 bit
// Expt. No. 9: To design a Lag Compensator by Bode–
   plot technique

clc
close
clf();

s=poly(0,'s')
// Transfer Function of Open–Loop System
G=syslin('c',30/((s)*(0.1*s+1)*(0.2*s+1)))
figure(0)
// Bode – Plot of Un–Compensated System
bode(G,0.1,100)

[gm,frg]=g_margin(G);
disp(gm,"Gain Margin",frg,"Phase Cut–off Frequency")
```

```
show_margins(G)
[phm,frp]=p_margin(G);
disp(phm,"Phase Margin",frp,"Gain Cut-off Frequency")
show_margins(G)

// Transfer Function of Open-Loop Compensated System, GGc
GGc=syslin('c',30*(1+3.33*s)/((s)*(0.1*s+1)*(0.2*s+1)*(1+33.3*s)))

// Bode - Plot of Compensated System
figure(1)
bode(GGc,0.1,100)

[gm,frg]=g_margin(GGc);
disp(gm,"Gain Margin",frg,"Phase Cut-off Frequency")
show_margins(GGc)
[phm,frp]=p_margin(GGc);
disp(phm,"Phase Margin",frp,"Gain Cut-off Frequency")
show_margins(GGc)

// Step Response of Closed-Loop Un-Compensated System
figure(2)
t= 0:0.1:10;
x=[csim('step',t,G/(1+G))]';
plot2d(t',x),xgrid(5, 1, 7)
xlabel("t", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
xtitle( 'Step Response of Un-Compensated System');

// Step Response of Closed-Loop Compensated System
figure(3)
t= 0:0.1:10;
x=[csim('step',t,GGc/(1+GGc))]';
plot2d(t',x),xgrid(5, 1, 7)
```

```
xlabel("t", "fontsize", 2,"color", "blue");
ylabel("Amplitude", "fontsize", 2, "color", "blue");
xtitle( 'Step Response of Compensated System');

// Result
// Frequency Response Specifications of Un-
   Compensated System
//Phase Cut-off Frequency
//
//    1.1253954
//
// Gain Margin
//
//  -6.0205999
//
// Gain Cut-off Frequency
//
//    1.555363
//
// Phase Margin
//
//  -17.245408
// Frequency Response Specifications of Compensated
   System

// Phase Cut-off Frequency
//
//    1.0788163
//
// Gain Margin
//
//    13.24356
//
// Gain Cut-off Frequency
//
//    0.4129329
//
// Phase Margin
```

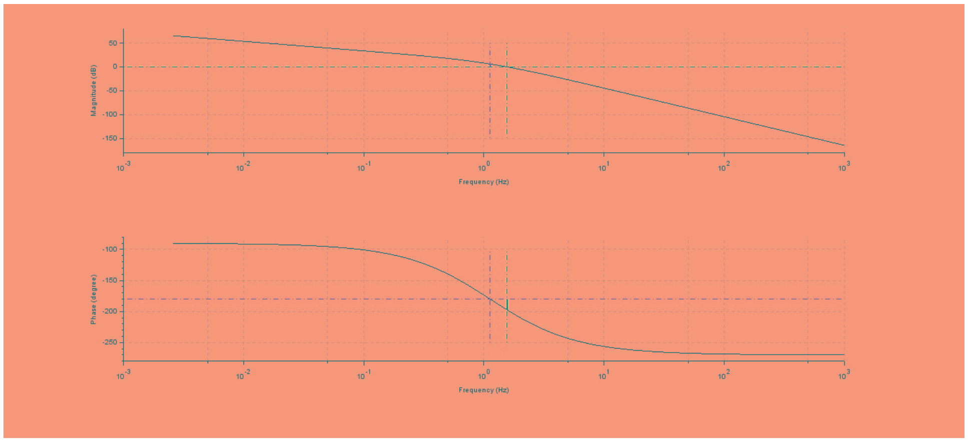


Figure 9.1: To design a Lag Compensator by Bode plot technique

```
//
//     42.090918
//
```

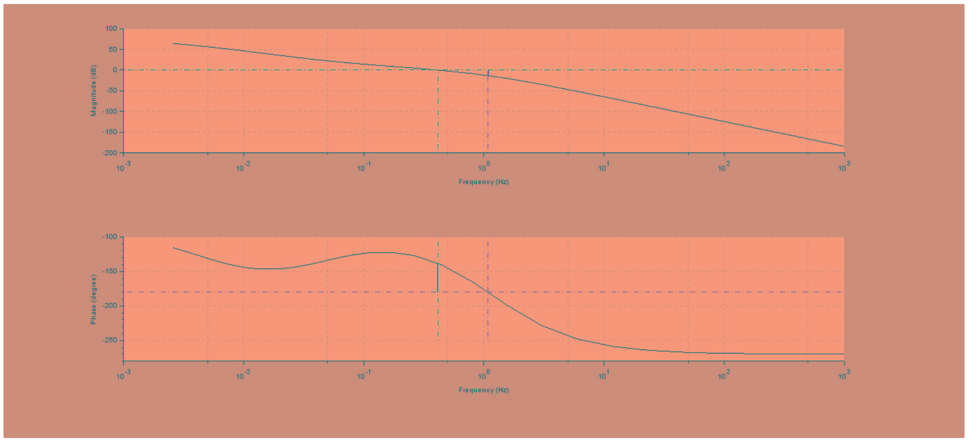


Figure 9.2: To design a Lag Compensator by Bode plot technique

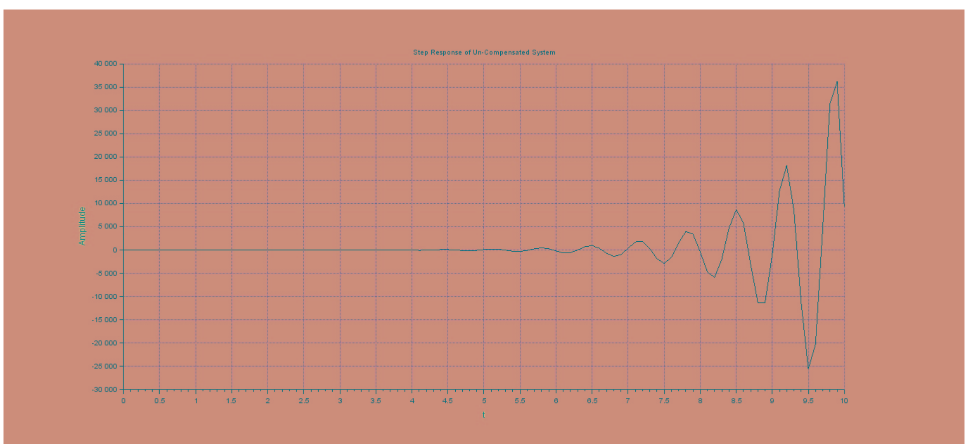


Figure 9.3: To design a Lag Compensator by Bode plot technique

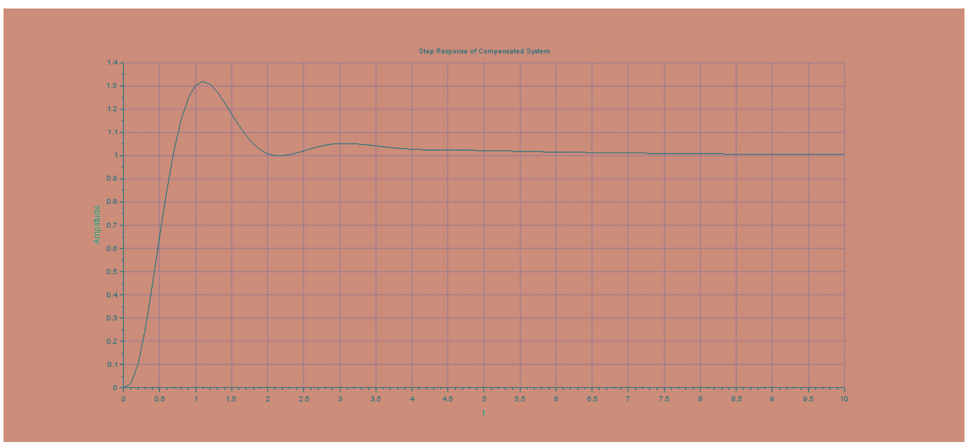


Figure 9.4: To design a Lag Compensator by Bode plot technique

# Experiment: 10

# PID Controller Design

This code can be downloaded from the website wwww.scilab.in

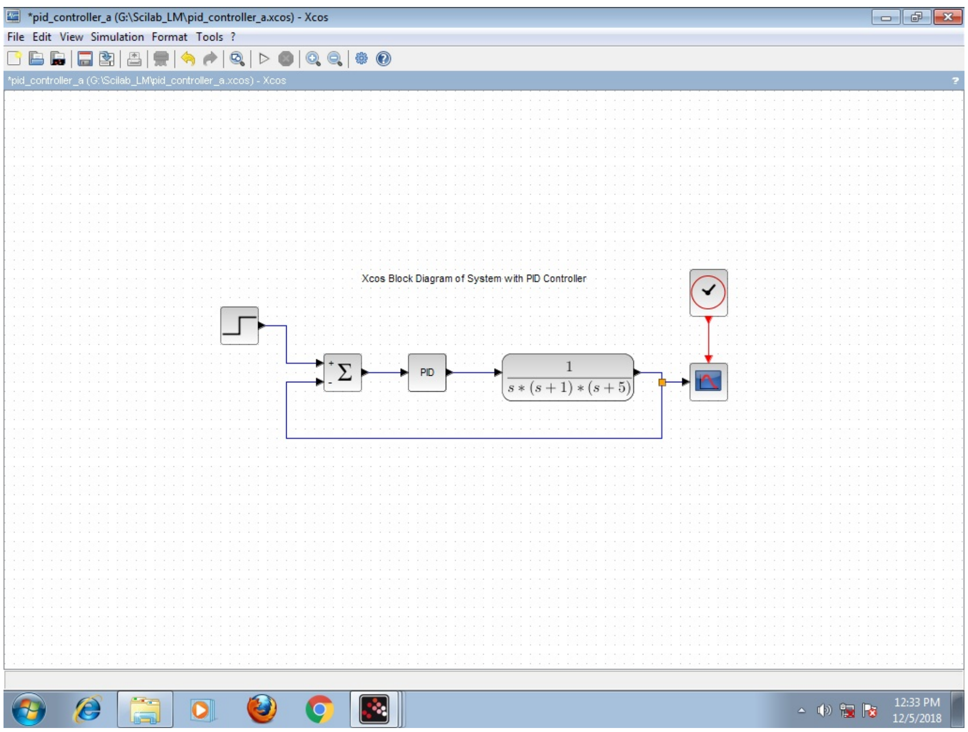


Figure 10.1: To design a PID Controller by using Ziegler Nichols tuning rules

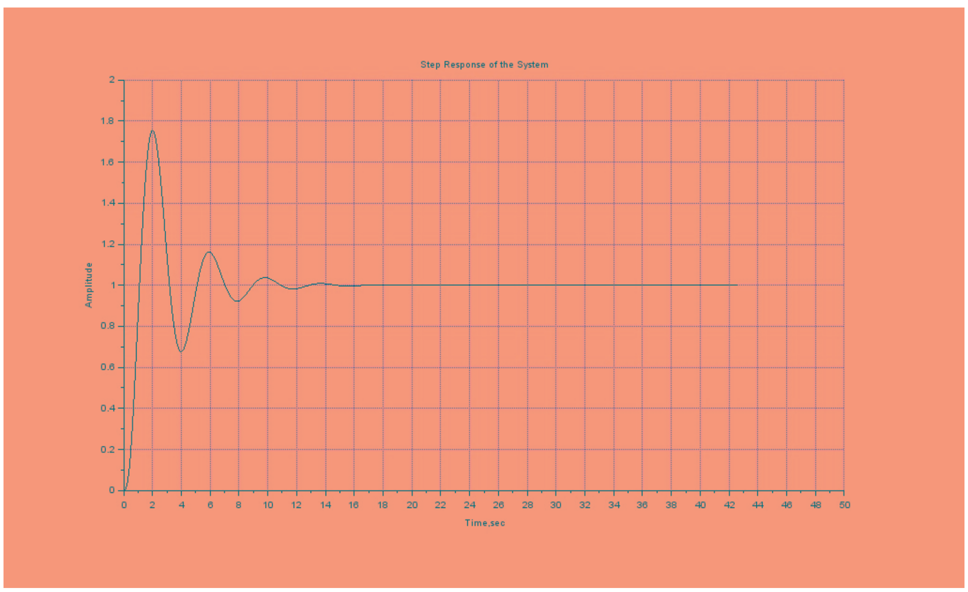


Figure 10.2: To design a PID Controller by using Ziegler Nichols tuning rules

# Experiment: 11

# Servomechanism or Tracking Problem

This code can be downloaded from the website wwww.scilab.in

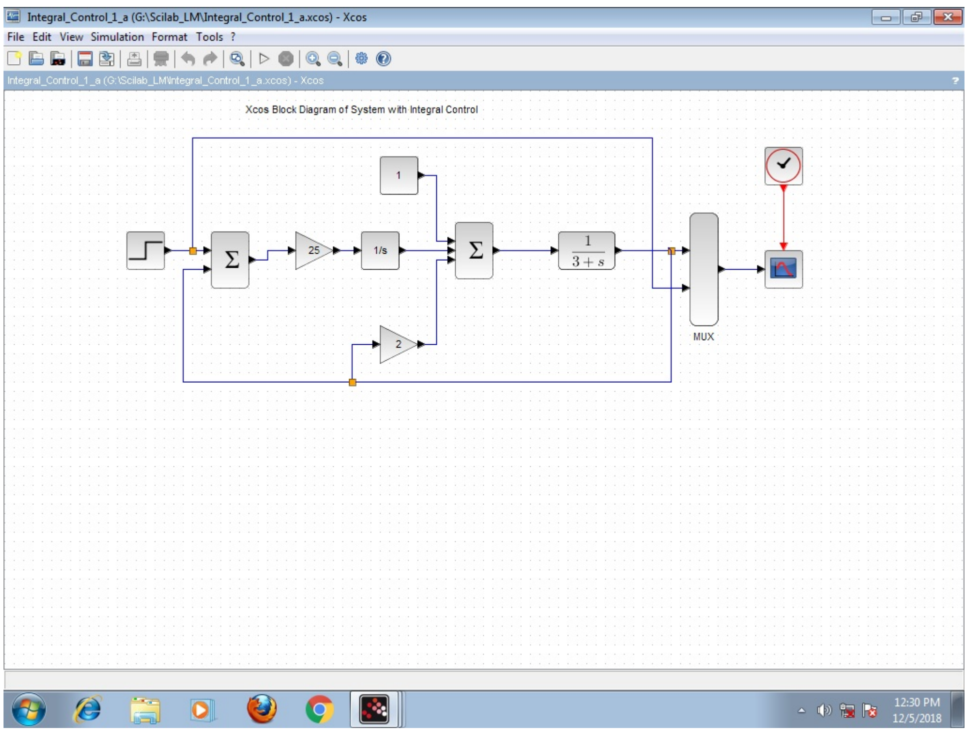


Figure 11.1: To apply integral control to servo problem to minimize the error

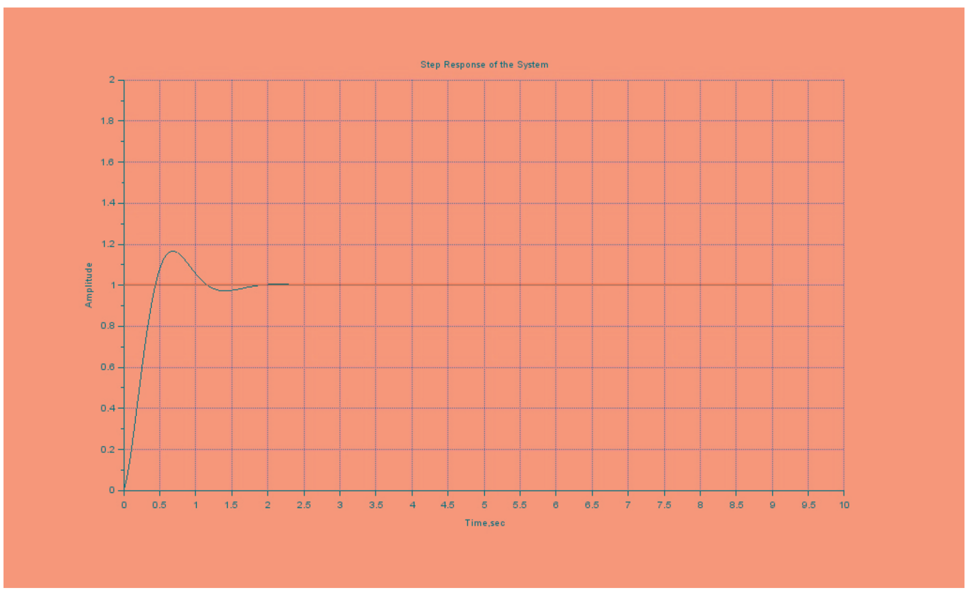


Figure 11.2: To apply integral control to servo problem to minimize the error