Scilab Textbook Companion for Control Systems by A Nagoor Kani[1]

Created by
Vibin Bharath M B
B.E.
Electrical Engineering
St.Xavier's Catholic College of Engineering
College Teacher
None
Cross-Checked by
None

July 31, 2019

[1]Funded by a grant from the National Mission on Education through ICT, http://spoken-tutorial.org/NMEICT-Intro. This Textbook Companion and Scilab codes written in it can be downloaded from the "Textbook Companion Project" section at the website http://scilab.in

Book Description

Title: Control Systems

Author: A Nagoor Kani

Publisher: Rba Publications, Chennai

Edition: 3

Year: 2015

ISBN: 978-8900121230

Scilab numbering policy used in this document and the relation to the above book.

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

AP Appendix to Example(Scilab Code that is an Appednix to a particular Example of the above book)

For example, Exa 3.51 means solved example 3.51 of this book. Sec 2.3 means a scilab code whose theory is explained in Section 2.3 of the book.

Contents

List of Scilab Codes

Chapter 2

COMPONENTS OF CONTROL SYSTEM

Scilab code Exa 2.1 POTENTIOMETER

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 2.1
clc;
clear;
n=300;// number of turns
li=0.01//linearity =1%
v=30//voltage=30V
kp=v/n
disp(kp,'Potentiometer constant in (volts/turn)')
nmid=n/2
vmid=kp*nmid
disp(vmid,'Voltage at mid point is;')
disp('Range of voltage at mid point with 1%
    linearity is 14.7volts to 15.3 volts')//vmid(+-)
    0.3
```

```
//assuming potentiometer is perfectly linear so the resistance at midpoint setting from reference is 50kiloohm
//load resistance of 500Kiloohm  is connected in parallel with potentiometer
req=(500*50)/(500+50)
v0=(v*req)/(50+req)//using voltage divison rule
disp(v0,'Voltage at midpoint with load resistance 500Kiloohm (in volts)')
```

Scilab code Exa 2.2 `GEAR TRAIN`

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 2.2
clc;
clear;
n1=20//no of teeth in first gear
n2=10//no of teeth in second gear
diratio=n1/n2
disp(diratio,'the ratio of diameters is')
theta1=40//gear1 is rotated by an angle of 40 degree
theta2=(n1/n2)*theta1
disp(theta2,'displace ment of gear2 in (degrees)')
as1=30//angular speed of gear1 is 30 rad/sec
as2=(n1/n2)*as1
disp(as2,' angular speed of gear2 in(rad/sec)')
aa2=4//angular accleration of gear 2 is 4 rad/sec^2
aa1=(n2/n1)*aa2
disp(aa1,'angular accleration of gear 1 in (rad/sec^2)')
t1=5//torque on gear1 5N-m
```

```
t2=(n2/n1)*t1
disp(t2,'torque on gear2 in (N-m)')
```

Scilab code Exa 2.3 GEAR TRAIN

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 2.3
clc;
clear;
n1=200//no of teeth in  gear1
n2=50//no of teeth in  gear2
n3=100//no of teeth in  gear3
n4=50//no of teeth in  gear4
n5=40//no of teeth in  gear5
n6=20//no of teeth in  gear6
n7=150//no of teeth in  gear7
//if gear 1 rotates in clockwise then all odd no
   gears rotate in clockwisw and even no gear rotate
    in anticlockwise
ad1=2//angular displacement in gear 1 is 2rad
ad4=(n1/n4)*ad1
disp(ad4,'angular displacement in gear 4 in rad (
   anticlockwise)')
ad7=(n1/n7)*ad1
disp(ad7,'angular displacement in gear 7 in rad  (
   clockwise)')
av6=20//angular velocity  of gear 6 is 20 rad/sec
av1=(n6/n1)*av6
disp(av1,'angular velocity of gear 1 in rad/sec; (
   clockwise)')
av3=(n6/n3)*av6
```

```
disp(av3,'angular velocity of gear 3 in rad/sec; (
    clockwise)')
t1=10//torque on gear 1 is 10 N-m
t3=(n3/n1)*t1
disp(t3,'torque in gear 3 in N-m')
t7=(n7/n1)*t1
disp(t7,'torque in gear 7 in N-m')
```

Chapter 3

TIME RESPONSE ANALYSIS

Scilab code Exa 3.1 RESPONSE OF THE SYSTEM

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 3.1

clc;
clear;
s=%s
p=poly([4],'s','coeff')
q=poly([0 5 1],'s','coeff')
g=p./q
disp(g,'The given transfer function is')
c=g/(1+g)
disp(c,'The closed loop transfer function is')
u=c/s
disp(u,'The input is unit step signal')
```

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

Scilab code Exa 3.2 RESPONSE OF THE SYSTEM

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 3.2

clc;
clear;
s=%s
p=poly([100],'s','coeff')
q=poly([0 2 1],'s','coeff')
h=poly([1 0.1 0 ],'s','coeff')
g=p./q
disp(g,'the given transfer function is')
c=g/(1+(g*h))
disp(c,'the closed loop transfer function is')
u=c/s
disp(u,'the in put is unit step signal')
```

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

Scilab code Exa 3.3 RESPONSE OF THE SYSTEM

```
//control systems by Nagoor Kani A
//Edition 3
```

```
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 3.3

clc;
clear;
s=poly(0,'s')
// the input is unit step signal
h=syslin('c', 600/(s^2+70*s+600))//the closed loop transfer function
disp(h,'the closed loop transfer function')
//standard form od second order system is w^2/s^2+2*zeta*w*s+w^2
//compaing h with the standard form
w=sqrt(600)//natural frequency of oscillation
disp(w,'natural frequency of oscillation in rad/sec')
zeta=70/(2*w)//damping ratio
disp(zeta,'damping ratio')
```

Scilab code Exa 3.4 RESPONSE OF THE SYSTEM

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 3.4

clc;
clear;
s=poly(0,'s')
// the input is unit step signal
h=syslin('c',100/(s^2+10*s+100))//the value of k is
```

```
    100
k=100
zeta=0.5//given damping ratio
disp(k,'the value of k is')
disp(h,'the closed loop transfer function')
//standard form od second order system is w^2/s^2+2*
    zeta*w*s+w^2
//compaing h with the standard form
w=sqrt(k)//natural frequency of oscillation
disp(w,'natural frequency of oscillation in rad/sec'
    )
mp=exp((-zeta*%pi)/sqrt(1-(zeta)^2))*100//percentage
     peak overshoot
disp(mp,'percentage peak overshoot in percentage')
tp=%pi/(w*sqrt(1-(zeta)^2))
disp(tp,'peak time in seconds')
```

Scilab code Exa 3.6 RESPONSE OF THE SYSTEM

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 3.6

clc;
clear;
s=poly(0,'s')
// the input is unit step signal
h=syslin('c',16/(s^2+4*s+16))//the value of k is 0.2
zeta=0.5//given damping ratio
disp(h,'the closed loop transfer function')
//standard form od second order system is w^2/s^2+2*
    zeta*w*s+w^2
```

```
//compaing h with the standard form
w=4//natural frequency of oscillation
disp(w,'natural frequency of oscillation in rad/sec'
   )
k=(2*zeta*w-(0.8))/16
disp(k,'the value of k is')
mp=exp((-zeta*%pi)/sqrt(1-(zeta)^2))*100//percentage
    peak overshoot
disp(mp,'percentage peak overshoot in percentage')
tp=%pi/(w*sqrt(1-(zeta)^2))
disp(tp,'peak time in seconds')
//constructing a right angle triangle with zeta and
   sqrt(1-zeta^2)
theta=atan(0.866/0.5)//(1-zeta^2)/zeta
disp(theta,'the value of theta is')
tr=(%pi- theta)/(w*sqrt(1-(zeta)^2))
disp(tr,'the rise time in seconds')
t=1/(zeta*w)//time constant
ts1=3*t//settling time for 5% error
disp(ts1,'settling time for 5% error in seconds')
ts2=4*t//settling time for 2% error
disp(ts2,'settling time for 2% error in seconds')
```

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

Scilab code Exa 3.7 RESPONSE OF THE SYSTEM

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 3.7

```

```
clc;
clear;
s=%s
p=poly([1 0.4 0 ],'s','coeff')
q=poly([0 0.6 1],'s','coeff')
g=p./q
disp(g,'the given transfer function is')
c=g/(1+g)
disp(c,'the closed loop transfer function is')
u=c/s
disp(u,'the in put is unit step signal')
//standard form od second order system is w^2/s^2+2*
    zeta*w*s+w^2
//compaing h with the standard form
w=1//natural frequency of oscillation
disp(w,'natural frequency of oscillation in rad/sec'
    )
zeta=1/(2*w)
disp(zeta,'the damping ratio is')
mp=exp((-zeta*%pi)/sqrt(1-(zeta)^2))*100//percentage
     peak overshoot
disp(mp,'percentage peak overshoot in percentage')
tp=%pi/(w*sqrt(1-(zeta)^2))
disp(tp,'peak time in seconds')
```

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

Scilab code Exa 3.9 RESPONSE OF THE SYSTEM

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
```

```
// Example 3.9

clc;
clear;
s=poly(0,'s')
// the input is 12 unit step signal
h=syslin('c',10/(s^2+2*s+10))
disp(h,'the closed loop transfer function')
//standard form od second order system is w^2/s^2+2*zeta*w*s+w^2
//compaing h with the standard form
w=3.162//natural frequency of oscillation
disp(w,'natural frequency of oscillation in rad/sec')
zeta=2/(2*w)
disp(zeta,' damping ratio is')
mp=exp((-zeta*%pi)/sqrt(1-(zeta)^2))*100//percentage peak overshoot
disp(mp,'percentage peak overshoot in percentage')
po=(mp/100)*12//peak over shoot for 12 units
disp(po,'peak over shoot for 12 units')
tp=%pi/(w*sqrt(1-(zeta)^2))
disp(tp,'peak time in seconds')
//constructing a right angle triangle with zeta and sqrt(1-zeta^2)
theta=atan(0.866/0.5)//(1-zeta^2)/zeta
disp(theta,'the value of theta is')
tr=(%pi- theta)/(w*sqrt(1-(zeta)^2))
disp(tr,'the rise time in seconds')
t=1/(zeta*w)//time constant
ts1=3*t//settling time for 5% error
disp(ts1,'settling time for 5% error in seconds')
ts2=4*t//settling time for 2% error
disp(ts2,'settling time for 2% error in seconds')
```

Chapter 4

FREQUENCY RESPONSE ANALYSIS

Scilab code Exa 4.1 BODE PLOT

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 4.1
clc;
clear;
s=poly(0,'s')//defines s as  poly nomial variable
h=syslin('c',(s^2)/((1+0.2*s)*(1+0.02*s)))//the
    given transfer function assigned to variable h .
    Assume the value of K as 1
scf(1)
bode(h,0.1,100)//frequency range
show_margins(h)
//calculation of system gain K
K=10^(-28/20)//value of K is calculated by equating
    20logK to -28db
disp(K,'the value of gain')
```

Scilab code Exa 4.2 BODE PLOT

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 4.2
clc;
clear;
s=poly(0,'s')//defines s as  poly nomial variable
h=syslin('c',(75*(1+0.2*s))/(s*(s^2+16*s+100)))//the
    given transfer function assigned to variable h
scf(1)
bode(h,0.1,100)//frequency range
show_margins(h)
a=g_margin(h)
b=p_margin(h)
disp(a,b,'the gain margin and phase margin  are')
```

Scilab code Exa 4.3 BODE PLOT

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 4.3
clc;
clear;
s=poly(0,'s')//defines s as  poly nomial variable
```

```
h=syslin('c',(%e*(0.2*s)/(s*(s+2)*(s+8))))//the
   given transfer function assiganed to variable h
   assume K=1
scf(1)
bode(h,0.1,100)//frequency range
show_margins(h)
//calculation of K
K1=10^(30/20)
disp(K1 ,'when gain margin =2db ' )
K2=10^(24/20)
disp(K2,'the value of K when phase margin is 45')
```

Scilab code Exa 4.4 BODE PLOT

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 4.4
clc;
clear;
s=poly(0,'s')//defines s as  poly nomial variable
h=syslin('c',(10)/(s*(1+0.4*s)*(1+0.1*s)))//the
   given transfer function assigned to variable h
scf(1)
bode(h,0.1,100)//frequency range
show_margins(h)
```

Scilab code Exa 4.5 BODE PLOT

```
//control systems by Nagoor Kani A
//Edition 3
```

```
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 4.5
clc;
clear;
s=poly(0,'s')////defines s as poly nomial variable
h=syslin('c',(20)/(s*(1+3*s)*(1+4*s)))//the given transfer function assigned to variable h
scf()
bode(h,0.1,100)
show_margins(h)
//calculation of gain cross over frequency
disp('from the plot the value of gain cross over frequency is :1.1rad/sec')
```

Scilab code Exa 4.6 BODE PLOT

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 4.6
clc;
clear;
s=poly(0,'s')//defines s as polynomial variable
h=syslin('c',(5*(1+2*s))/(1+4*s)*(4+0.25*s))//the given transfer function assigned to variable h
scf(1)
bode(h,0.1,100)//frequency range
show_margins(h)
```

Chapter 5

CONCEPTS OF STABILITY AND ROOTLOCUS

Scilab code Exa 5.1 ROUTH CRITERION

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.1

clc;
clear;
s=poly(0,'s')
a=(s^4)+(8*s^3)+(18*s^2)+(16*s)+5
b=coeff(a)
n=length(b)
R=routh_t(a)
disp(R,'the routh array is;')
disp('since there is no sign change  the system is
    stable')
disp('all the four roots lie left half of the s
    plane')
```

Scilab code Exa 5.2 ROUTH CRITERION

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.2

clc;
clear;
s=poly(0,'s')
a=(s^6)+(2*s^5)+(8*s^4)+(12*s^3)+(20*s^2)+(16*s)+16
b=coeff(a)
n=length(b)
R=routh_t(a)
disp(R,'the routh array is;')
ap=s^4+6*s^2+8
r=roots(ap)
disp(r,'the roots of auxilary polynomial;')
disp('the system is marginally stable;')
disp('four roots lying in imaginary axis ;')
```

Scilab code Exa 5.3 ROUTH CRITERION

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.3
```

```

clc;
clear;
s=poly(0,'s')
a=(s^5)+(s^4)+(2*s^3)+(2*s^2)+(3*s)+5
b=coeff(a)
n=length(b)
R=routh_t(a)
disp(R,'the routh array is;')
disp('the system is unstable')
```

Scilab code Exa 5.4 ROUTH CRITERION

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.4

clc;
clear;
s=poly(0,'s')
a=(9*s^5)-(20*s^4)+(10*s^3)-(s^2)-(9*s)-10
b=coeff(a)
n=length(b)
R=routh_t(a)
disp(R,'the routh array is;')
disp('there are 3 sign changes in first column of
    routh array ')
disp('three roots lie on right side of s plane so
    the system is unstable')
```

Scilab code Exa 5.5 ROUTH CRITERION

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.5

clc;
clear;
s=poly(0,'s')
a=(s^7)+(9*s^6)+(24*s^5)+(24*s^4)+(24*s^3)+(24*s^2)
    +(23*s)+15
b=coeff(a)
n=length(b)
R=routh_t(a)
disp(R,'the routh array is;')
disp('there is sign change in first column of routh
    array so ths system is unstable;')
ae=s^4+s^2+1
r=roots(ae)
disp(r,'the roots of auxilary equation are')
disp('two roots lie on right half of splane five
    roots lie on left half of s plane')
```

Scilab code Exa 5.6 ROUTH CRITERION

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.6

```

```
clc;
clear;
s=poly(0,'s')
a=(s^7)+(5*s^6)+(9*s^5)+(9*s^4)+(4*s^3)+(20*s^2)
   +(36*s)+36
b=coeff(a)
n=length(b)
R=routh_t(a)
disp(R,'the routh array is;')
//divide characterstic equation by auxilary
   polynomial to get quotient polynomial
//routh table for quotient poly nomial
a1=(s^3)+(5*s^2)+(9*s)+9 //quotient poly nomial
b1=coeff(a1)
n1=length(b1)
R1=routh_t(a1)
disp(R1,'the routh array for quotient poly nomial is
   ;')
ap=s^4+4 //auxillary polynomial
r=roots(ap)
disp(r,'the roots are')
disp('the system is unstable')
disp('two roots on right half of s plane and five
   roots lie in left half of s plane')
```

Scilab code Exa 5.7 ROUTH CRITERION

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.7

clc;
```

```
clear;
s=poly(0,'s')
a=(s^5)+(4*s^4)+(8*s^3)+(8*s^2)+(7*s)+4
b=coeff(a)
n=length(b)
R=routh_t(a)
disp(R,'the routh array is;')
ap=s^2+1
r=roots(ap)
disp(r,'the roots are')
disp ('the roots of auxillary equation are in
    imagianry axis so the system is marginally stable
    ')
disp('three roots lie in left half of s plane')
```

Scilab code Exa 5.8 ROUTH CRITERION

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.8

clc;
clear;
s=poly(0,'s')
a=(s^6)+(s^5)+(3*s^4)+(3*s^3)+(3*s^2)+(2*s)+1
b=coeff(a)
n=length(b)
R=routh_t(a)
disp(R,'the routh array is;')
//characterstic polynomial can be expressed as
    product of auxillary polynomial and quotient
    polynomial
```

```
//divide characterstic equation by auxilary
   polynomial to get quotient polynomial
//routh table for quotient poly nomial
a1=(s^4)+(s^3)+(2*s^2)+(2*s)+1 //quotient poly
   nomial
b1=coeff(a1)
n1=length(b1)
R1=routh_t(a1)
disp(R1,'the routh array for quotient poly nomial is
   ;')
ap=s^2+1
r=roots(ap)
disp(r,'the roots are')
disp('the system is unstable')
disp('two roota on imaginary axis ,two roots on
   right half of s plane and two roots lie in left
   half of s plane')
```

Scilab code Exa 5.9 ROUTH CRITERION

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.9

clc;
clear;
s=poly(0,'s')
//the close loop tranfer function is K/s*(s+1)*(s+2)
   +K
a=(s^3)+(3*s^2)+(2*s)+1//the characterstic equation
   assuming K=1
b=coeff(a)
```

```
n=length(b)
R=routh_t(a)
disp(R,'the routh array is;')
disp('the valus of K lies between 0 to6 for the
   system to be stable')
```

Scilab code Exa 5.12 ROUTH CRITERION

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.12

clc;
clear;
s=poly(0,'s')
//the close loop tranfer function is K(1-s)/s*(s^2+5
   s+9)+K(1-s)
a=(s^3)+(5*s^2)+(8*s)+1//the charater stic equation
   is assuming K=1
b=coeff(a)
n=length(b)
R=routh_t(a)
disp(R,'the routh array is;')
disp('the valus of K lies between 0 to7.5 for the
   system to be stable')
```

Scilab code Exa 5.13 NYQUIST PLOT

```
//control systems by Nagoor Kani A
//Edition 3
```

```
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.13

clc;
clear;
s=poly(0,'s')//defines s as  poly nomial variable
h=syslin('c',(230/(s*(s+2)*(s+10))))//the value of K
     lies between 0 to 240 .the given transfer
    function assigned to variable h
nyquist(h)
show_margins(h,'nyquist')
//calculation of K
//when -0.00417K=-1the contour passes through -1+j0
    and the corrospondig value of K is the limiting
    value of k for stabilty
K=1/0.00417
disp(K,'the value of K is')
```

Scilab code Exa 5.14 NYQUIST PLOT

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.14

clc;
clear;
s=poly(0,'s')//defines s as  poly nomial variable
h=syslin('c',(0.6*(1+s)^2/(s^3)))//the system is
    stable for K vlaue greater than 0.5. the given
    transfer function assigned to variable h
```

```
scf()
nyquist(h)
show_margins(h,'nyquist')
//when K>0.5 −1+j0 is encircled in both clockwise
   and anticlockwise direction one time. so the
   system is stable
disp('the system is stable for K vlaue greater than
   0.5.')
```

Scilab code Exa 5.15 NYQUIST PLOT

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.15

clc;
clear;
s=poly(0,'s')//defines s as  poly nomial variable
h=syslin('c',(1+4*s)/(s*(1+s)*(1+2*s)))//the given
   transfer function assigned to variable h
scf()
nyquist(h)
show_margins(h,'nyquist')
disp('the closed loop system is unstable ')
disp('two poles of closed loop system are lying on
   right half of s plane')
```

Scilab code Exa 5.16 NYQUIST PLOT

```
//control systems by Nagoor Kani A
```

```
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.16

clc;
clear;
s=poly(0,'s')//defines s as  poly nomial variable
h=syslin('c',(7*(1+0.5*s)*(1+s)/(1+10*s)*(s-1)))//
   system is stable for K>6.....the given transfer
   function assigned to variable h
scf()
nyquist(h)
show_margins(h,'nyquist')
//calculation of K
//when -0.01667K=-1the contur passes through -1=
   j0and the corrospong value of K is the limiting
   value of k for stabilty
K=1/0.1667
disp(K,'the value of K is')
disp('the open loop system is unstable')
disp('for stability of closed loop system K>6')
```

Scilab code Exa 5.17 NYQUIST PLOT

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.17

clc;
clear;
```

```
s=poly(0,'s')//defines s as  polynomial variable
h=syslin('c',(5/(s*(1-s))))//the given transfer
   function assigned to variable h
scf()
nyquist(h)
disp('Both open loop and closed loop functions are
   unstable ')
```

Scilab code Exa 5.18 NYQUIST PLOT

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.28

clc;
clear;
s=poly(0,'s')//defines s as  poly nomial variable
h=syslin('c',(s+2)/(s+1)*(s-1))//the given transfer
   function assigned to variable h
scf()
nyquist(h)
show_margins(h,'nyquist')
disp('open loop system is unstable')
disp('closed loop system is stable;' )
```

Scilab code Exa 5.22 ROOT LOCUS

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
```

```
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.22

clc;
clear;
s=poly(0,'s')//defines s as  poly nomial variable
G=syslin('c',(1/(s*(s^2+4*s+13))))//the given transfer function assigned to variable G assume K =1
scf()
evans(G)
//calculation of K
disp('the characterstic equation is given by : s^2+4*s^2+13*s+K')
//put s=jw and equate real and imaginary parts
//K=4*w^2
K=4*13
disp(K,'the value of K is ')
```

Scilab code Exa 5.23 ROOT LOCUS

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.23

clc;
clear;
s=poly(0,'s')//defines s as  poly nomial variable
h=syslin('c',(48/(s*(s+2)*(s+4))))//the given transfer function assigned to variable h assume K =1
```

```
scf()
evans(h)
//calculation of K
disp('the characterstic equation is given by : s^3+6*s^2+8*s+K')
//put s=jw and equate real and imaginary parts
//K=4*w^2
K=6*8
disp(K,'the value of K is ')
```

Scilab code Exa 5.24 ROOT LOCUS

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.24

clc;
clear;
s=poly(0,'s')//defines s as  poly nomial variable
h=syslin('c',(s+9)/(s*(s^2+4*s+11)))//the given transfer function assigned to variable h assume K=1
scf()
evans(h)
//the characterstic equation is (s^3+4s^2+11s)+Ks+9K
//put s=jw and equating real and imaginary parts to calculate K
K=(4*(4.4)^2)/9//the value of w is 4.4
disp(K,'the value of K is;')
```

Scilab code Exa 5.25 ROOT LOCUS

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.25

clc;
clear;
s=poly(0,'s')//defines s as  poly nomial variable
h=syslin('c',(1/(s*(s+4)*(s^2+4*s+20))))// the given
    transfer function assigned to variable h assume
   K=1
scf()
evans(h)
//the characterstic equation is (s^4+8s^3+36s^2+80s)
   +K
//put s=jw and equating real and imaginary parts to
   calculate K
K=-(3.2)^4+36*(3.2)^2//the value of w is 3.2
disp(K,'yhe value of K is;')
```

Scilab code Exa 5.26 ROOT LOCUS

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.26

clc;
clear;
```

```
s=poly(0,'s')//defines s as  poly nomial variable
h=syslin('c',(s+15)/(s*(s+1)*(s+5)))//the given transfer function assigned to variable h assume K =1
scf()
evans(h)
//the characterstic equation is (s^3+6s^2+5s)+Ks+15K
//put s=jw and equating real and imaginary parts to calculate K
K=30/(-4.5)
disp(K,'yhe value of K is;')
```

Scilab code Exa 5.27 ROOT LOCUS

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.27

clc;
clear;
s=poly(0,'s')//defines s as  poly nomial variable
K=poly(0,'K')
h=syslin('c',(s^2+6*s+25)/(s*(s+1)*(s+2)))//the given transfer function assigned to variable h assume K=1
scf()
evans(h)
//the characterstic equation is (s^3+(3+K)s^2+(2+6K)s)25K
//put s=jw and equating real and imaginary parts to calculate K
a=roots((-6*K^2)+5*K-6)
```

```
disp(a,'the value of K is')
```

Scilab code Exa 5.28 ROOT LOCUS

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 5.28

clc;
clear;
s=poly(0,'s')//defines s as  poly nomial variable
h=syslin('c',(1/(s*(s^2+6*s+10))))//the given
    transfer function assigned to variable h assume K
    =1
scf()
evans(h)
//the characterstic equation is (s^3+6s^2+10s)+K
//put s=jw and equating real and imaginary parts to
    calculate K
K=6*(3.2)^2//the valuw of w is 3.2
disp(K,'the value of K is ')
```

Chapter 6

LINEAR SYSTEM DESIGN

Scilab code Exa 6.1 LAG COMPENSATOR

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 6.1

clc;
clear;
s=poly(0,'s')
//calculation of gain K
//for ramp input ess(steady state error ) is0.2
ess=0.2
kv=1/ess
// open loop transfer function G(s)=K/s*(1+2*s)
//by definition of velocity error constant applying
    limit s=0 in G(s)
disp('the value of K is 5;')
h=syslin('c',5/(s*(1+2*s)))
bode(h)
show_margins(h)
```

```
xtitle("uncompensated system")
//from the plot the phase margin of uncompensated system is 18
//but the system requires phase margin of 40 so lag compensation required
pm=45//choose PM of compensated system is 45 degree
phigcn=45-180// phase of G(s) at new gain cross over frequency
wgcn=0.5//the frequency corrosponding to  phase of −135 is 0.5 rad/sec
agcn=20//db magnitude at wgcn
//20log betaa=20
betaa=10^(agcn/20)
disp(betaa,'the value for betaa is')
zc=wgcn/10//zero of lag compensator
t=10/wgcn
disp(t,'the value for t is')
pc=1/(betaa*t)
disp(pc,'pole of lag compensator is')
//transfer function of lag compensation is (s+1/t)/(s+1/betaa*t))
hc=syslin('c', (10*(1+20*s))/(1+200*s))
disp(hc,'the transfer function of lag compensator is;')
//open loop transfer function of compensated system is h*hc
hcmp=syslin('c',h*hc)
disp(hcmp,'open loop transfer function of compensated system is')
figure()
bode(hcmp)
show_margins(hcmp)
xtitle("compensated system")
```

Scilab code Exa 6.2 LAG COMPENSATOR

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 6.2

clc;
clear;
s=poly(0,'s')
//calculation of gain K
kv=30//velocity error constant is 30sec^-1
// open loop transfer function G(s)=K/s*(s+4)*(s+80)
//by definition of velocity error constant applying limit s=0 in G(s)
disp('the value of K is 9600;')
h=syslin('c',9600/(s*(s+4)*(s+80)))
bode(h)
show_margins(h)
xtitle("uncompensated system")
//from the plot the phase margin of uncompensated system is 12
//but the system requires phase margin of 33 so lag compensation required
pm=38//choose PM of compensated system is 38 degree
phigcn=38-180// phase of G(s) at new gain cross over frequency
wgcn=4.7//the frequency corrosponding to  phase of -142 is 4.7 rad/sec
agcn=16//db magnitude at wgcn
//20log betaa=16
betaa=10^(agcn/20)
disp(betaa,'the value for betaa is')
zc=wgcn/10//zero of lag compensator
t=10/wgcn
disp(t,'the value for t is')
pc=1/(betaa*t)
disp(pc,'pole of lag compensator is')
```

```
//transfer function of lag compensation is (s+1/t)/(s+1/betaa*t))
hc=syslin('c', (6.3*(1+2.13*s))/(1+13.419*s))
disp(hc,'the transfer function of lag compensator is ;')
//open loop transfer function of compensated system is h*hc
hcmp=syslin('c',h*hc)
disp(hcmp,'open loop transfer function of compensated system is')
figure()
bode(hcmp)
show_margins(hcmp)
xtitle("compensated system")
```

Scilab code Exa 6.3 LAG COMPENSATOR

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 6.3

clc;
clear;
s=poly(0,'s')
K=20//the value of K
h=syslin('c',20/(s*(s+2)*(s+8)))
evans(h)
xtitle("uncompensated system")
//given  ramp input ess(steady state error ) is0.125
ess=0.125
kvd=1/ess//desired velocity error constant
// transfer function of un compessated system G(s)
```

```
=20/s*(s+2)*(s+8)
//by definition of velocity error constant applying limit s=0 in G(s)
kvu=1.25
disp(kvu,'The velocity error constant of un compensated system is')
disp(kvd,'desired velocity error constant')
A=kvd/kvu//A is the factor by which velocity error constant increases
betaa=A*1.2
disp(betaa,'the value of betaa is')
zc=0.1*(-2)//zero of lag compensator=0.1* second pole
t=(-1/zc)
disp(t,'the value for t is')
pc=(-1)/(betaa*t)//pole of lag compensator
//transfer function of lag compensation is (s+1/t)/(s+1/betaa*t))
hc=syslin('c',(s+0.2*s)/(s+0.026*s))
disp(hc,'transfer function of lag compensation is')
//open loop transfer function of compensated system is h*hc
hcmp=syslin('c', h*hc)
disp(hcmp,'open loop transfer function of compensated system is')
figure()
evans(hcmp)
xtitle("compensated system")
//by definition of velocity error constant applying limit s=0 in hcmp
kvc=9.165//velocity error constant of compensated system
essc=1/kvc//steady state error for compensated system
disp(essc,'steady state error for compensated system')
disp('since the steady state error of compensated system is less than 0.125 the design is
```

```
acceptable')
```

Scilab code Exa 6.4 LAG COMPENSATOR

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 6.4

clc;
clear;
s=poly(0,'s')
K=240//the value of K
h=syslin('c',240/(s*(s+10)^2))
evans(h)
xtitle("uncompensated system")
kvd=20//given desired velocity error constant
// transfer function of un compessated system G(s)=20/s*(s+2)*(s+8)
//by definition of velocity error constant applying limit s=0 in G(s)
kvu=2.4
disp(kvu,'The velocity error constant of un compensated system is')
disp(kvd,'desired velocity error constant')
A=kvd/kvu//A is the factor by which velocity error constant increases
betaa=A*1.2
disp(betaa,'the value of betaa is')
zc=0.1*(-10)//zero of lag compensator=0.1* second pole
t=(-1/zc)
disp(t,'the value for t is')
```

```
pc=(-1)/(betaa*t)//pole of lag compensator
//transfer function of lag compensation is (s+1/t)/(
    s+1/betaa*t))
hc=syslin('c',(s+1)/(s+0.1))
disp(hc,'transfer function of lag compensation is')
//open loop transfer function of compensated system
    is h*hc
hcmp=syslin('c', h*hc)
disp(hcmp,'open loop transfer function of
    compensated system is')
figure()
evans(hcmp)
xtitle("compensated system")
//by definition of velocity error constant applying
    limit s=0 in hcmp
kvc=24//velocity error constant of compensated
    system
disp('since the velocity error constant of
    compensated system is greater than specified
    value the design is acceptable')
```

Scilab code Exa 6.5 LEAD COMPENSATOR

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 6.5

clc;
clear;
s=poly(0,'s')
//calculation of gain K
//given for ramp input ess(steady state error ) is
```

```
1/15
ess=1/15
kv=1/ess
// open loop transfer function G(s)=K/s*(s+1)
//by definition of velocity error constant applying limit s=0 in G(s)
disp('the value of K is 15;')
h=syslin('c',15/(s*(s+1)))
bode(h)
show_margins(h)
xtitle("uncompensated system")
//from the plot the phase margin of uncompensated system is 13
//but the system requires phase margin of 45 so lead compensation required
pm=45//choose PM of compensated system is 45 degree
phim=37//maximum lead angle
alpha=(1-(sind(phim)))/(1+(sind(phim)))
disp(alpha,'the vale of alpha is')
wmdb=-20*log(1/sqrt(alpha))////db magnitude
wm=5.6//from the bode plot of uncompensated system the frequency wm corrosponding to db gain of -6db is 5.6rad/sec
t=1/(wm*sqrt(alpha))
disp(t,'the value of t is')
//transfer function of lead compensator is (s+1/t)/(s+1/alpha*t)
hc=syslin('c',(0.25*(1+0.36*s))/(1+0.09*s))
disp(hc,' transfer function of lead compensator is')
//open loop transfer function of compensated system is h*hc
hcmp=syslin('c',h*hc)
disp(hcmp,'open loop transfer function of compensated system is ')
figure()
bode(hcmp)
show_margins(hcmp)
xtitle("compensated system")
```

Scilab code Exa 6.6 LEAD COMPENSATOR

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 6.6

clc;
clear;
s=poly(0,'s')
//calculation of gain K
kv=50//given velocity error constant
// open loop transfer function G(s)=K/s*(s+1)(s+5)
//by definition of velocity error constant applying limit s=0 in G(s)
disp('the value of K is 250;')
h=syslin('c',250/(s*(s+1)*(s+5)))
bode(h)
show_margins(h)
xtitle("uncompensated system")
//from the plot the phase margin of uncompensated system is -44
//but the system requires phase margin of 20 so lead compensation required
pm=20//choose PM of compensated system is 20 degree
//since the lead angle required is greater than 60 we have to realise lead compensator as cascade of two compensators with each compensator providing half of required phase
phim=69/2//maximum lead angle
alpha=(1-sind(phim))/(1+sind(phim))
disp(alpha,'the vale of alpha is')
```

```
wmdb=-20*log(1/sqrt(alpha))////db magnitude
wm=7.8//from the bode plot of uncompensated system the frequency wm corrosponding to db gain of -6db is 5.6rad/sec
t=1/(wm*sqrt(alpha))
disp(t,'the value of t is')
//transfer function of lead compensator is (s+1/t)/(s+1/alpha*t)
hc=syslin('c',(0.0784*(1+0.024*s)^2)/(1+0.067*s)^2)
disp(hc,' transfer function of lead compensator is')
//open loop transfer function of compensated system is h*hc
hcmp=syslin('c',h*hc)
disp(hcmp,'open loop transfer function of compensated system is ')
figure()
bode(hcmp)
show_margins(hcmp)
xtitle("compensated system")
```

Scilab code Exa 6.9 LAG LEAD COMPENSATOR

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 6.9

clc;
clear;
s=poly(0,'s')
//calculation of gain K
kv=80//given velocity error constant
// open loop transfer function G(s)=K/s*(s+3)(s+6)
```

```
//by definition of velocity error constant applying limit s=0 in G(s)
disp('the value of K is 1440;')
h=syslin('c',1440/(s*(s+3)*(s+6)))
bode(h)
show_margins(h)
xtitle("uncompensated system")
//from the plot the phase margin of uncompensated system is -46
pm=40//choose PM of compensated system is 40 degree
phigcn=40-180// phase of G(s) at new gain cross over frequency
wgcn=1.8//the frequency corrosponding to  phase of -140 is 1.8 rad/sec
wgcl=4//choose gain cross over frequency of lag compensator as 4rad/sec
agcl=23//db magnitude at egcl is 23db
//agcl=20log*betaa
betaa=10^(agcl/20)
disp(betaa,'the value of betaa is')
zc1=wgcl/10//zero of lag compensator
t1=10/wgcl
disp(t1,'the value of t1 is ')
pc1=1/(betaa*t1)
disp(pc1,'pole of lag compensator is')
//transfer function of lag section is (betaa*1+st1)/(1+s*betaa*t1)
hc1=syslin('c',(14*(1+2.5*s))/(1+35*s))
disp(hc1,'transfer function of lag section')
alpha=1/betaa
disp(alpha,'the value of alpha is')
wmdb=-20*log(1/sqrt(alpha))////db magnitude
wm=17//from the bode plot of uncompensated system the frequency wm corrosponding to db gain of -12 db is 17rad/sec
t2=1/(wm*sqrt(alpha))
//transfer function of lead section is (alpha*1+st2)/(1+s*alpha*t2)
```

```
hc2=syslin('c',(0.07*(1+0.22*s))/(1+0.0154*s))
disp(hc2,'transfer function of lead section')
hc3=syslin('c',hc1*hc2)
disp(hc3,'the tansfer function of lag lead
   compensation system is')
//open loop transfer function of compensated system
   is h*hc3
hcmp=syslin('c',h*hc3)
disp(hcmp,'the overll transfer function of
   compensated system')
figure()
bode(hcmp)
show_margins(hcmp)
xtitle("compensated system")
```

Scilab code Exa 6.11 PD CONTROLLER

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 6.11

clc;
clear;
s=poly(0,'s')
//given tranfer function g(s)=10/(s*(1+2*s)*(1+s))
h=syslin('c',10/(s*(1+2*s)*(1+s)))
pm=30//given phase margin
w=6.2//given gain cross over frequency in rad/sec
//put s=jw in G(s) magnitude of G(jw) gives A1 and
   angle of G(jw) gives phi1 at w
A1=2.052
phi=-207.5//in degrees
```

```
theta=30-(-27.5)//desired pm -pm of uncompensated system
kd=sind(theta)/w*A1//derivative constant
kp=cosd(theta)/A1//proportional constant
disp(kd,kp,'the values of derivative constant and proportional constant are')
//transfer function of PD controller is (kp+kd*s)
hc=syslin('c', s*((0.343/s)+0.262))
disp(hc,'the transfer function of PD controller is')
hcmp=syslin('c', h*hc)
disp(hcmp,'the transfer function of compensated system is')
```

Scilab code Exa 6.12 PI CONTROLLER

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 6.12

clc;
clear;
s=poly(0,'s')
//given tranfer function g(s)=100/(s+1)*(s+2)*(s+5)
h=syslin('c',100/(s+1)*(s+2)*(s+5))
pm=60//given phase margin
w=0.5//given gain cross over frequency in rad/sec
//put s=jw in G(s) magnitude of G(jw) gives A1 and angle of G(jw) gives phi1 at w
A1=8.63
phi=-46//in degrees
theta=pm-134//desired pm -pm of uncompensated system
ki=(-w)*sind(theta)/A1//integral constant
```

```
kp=cosd(theta)/A1//proportional constant
disp(ki,kp,'the values of integral constant and
   proportional constant are')
//transfer function of PI controller is (kp+ki/s)
hc=syslin('c', 0.056*(1+0.57*s)/s)
disp(hc,'the transfer function of PD controller is')
hcmp=syslin('c', h*hc)
disp(hcmp,'the transfer function of compensated
   system is')
```

Scilab code Exa 6.13 PID CONTROLLER

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 6.13

clc;
clear;
s=poly(0,'s')
//given tranfer function g(s)=100/(s+1)*(s+2)*(s+10)
h=syslin('c',100/(s+1)*(s+2)*(s+10))
pm=45//given phase margin
w=4//given gain cross over frequency in rad/sec
//put s=jw in G(s) magnitude of G(jw) gives A1 and
   angle of G(jw) gives phi1 at w
A1=0.5
phi=-161//in degrees
theta=pm-19//desired pm -pm of uncompensated system
ess=0.1//steady state error for ramp input
kv=1/ess//velocity errpr constant
//the transfer function of PID controller is Gc(s)=
   kp+kd*s+ki/s
```

```
//by definition of velocity error constant applying s=0 in S*Gc(s)*G(s)
ki=2//integeral constant
disp(ki,'the value of integral constant')
kd=((sind(theta)/(w*A1))+(ki/w^2))//derivative constant
kp=cosd(theta)/A1//proportional constant
disp(kd,kp,'the values of proportional  constant and derivative  constant  are')
hc=syslin('c',0.344*(s^2+5.23*s+5.81)/s)
disp(hc,'the transfer function of PID controller is')
hcmp=syslin('c',h*hc)
disp(hcmp,'the transfer function of compensated system is')
```

Scilab code Exa 6.14 PD CONTROLLER

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 6.14

clc;
clear;
s=poly(0,'s')
//dominent pole sd=-zeta*w=%i*w*sqrt(1-zeta^2)
zeta=0.8//damping ratio
w=2//natural frequency of osciilation in rad/sec
sd=(-zeta*w)+((%i*w)*sqrt(1-zeta^2))
disp(sd,'the dominennt pole is')
d=abs(sd)
disp(d,'the value of d is ')
```

```
betaa=phasemag(sd)
disp(betaa,'the value of betaa is;')
h=syslin('c',20/s*(s+2)*(s+4))//given tranfer
   function G(s)
//find magnitude and phase of G(s) at s=sd
a=20/(sd*(2+sd)*(4+sd))
ad=abs(a)
disp(ad,'the value of ad is')
phid=phasemag(a)
disp(phid,'the value of phid is')
kd=sind(phid)/(d*ad*sind(betaa))//derivative
   constant
disp(kd,'the derivative  constant is')
kp=(-sind(betaa+phid))/(ad*sind(betaa))//
   proportional constant
disp(kp,'the integral  constant is')
hc=syslin('c', s*((0.243/s)+0.557))//transfer
   function of PD controller is kpof +kd*s
disp(hc,'transfer function of PD controller is')
hcmp=syslin('c',h*hc)//transfer function
   compensated system
disp(hcmp,'transfer function   compensated system ')
```

Scilab code Exa 6.15 PI CONTROLLER

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 6.15

clc;
clear;
s=poly(0,'s')
```

```
//dominent pole sd=-zeta*w=%i*w*sqrt(1-zeta^2)
zeta=0.9//damping ratio
w=2.5//natural frequency of osciilation in rad/sec
sd=(-zeta*w)+((%i*w)*sqrt(1-zeta^2))
disp(sd,'the dominennt pole is')
d=abs(sd)
disp(d,'the value of d is ')
betaa=phasemag(sd)
disp(betaa,'the value of betaa is;')
h=syslin('c',4/(s+1)*(s+5))//given tranfer function G(s)
//find magnitude and phase of G(s) at s=sd
a=4/((1+sd)*(5+sd))
ad=abs(a)
disp(ad,'the value of ad is')
phid=phasemag(a)
disp(phid,'the value of phid is')
ki=-(d*sind(phid))/(ad*sind(betaa))//integral constant
disp(ki,'the integral  constant is')
kp=(-sind(betaa+phid))/(ad*sind(betaa))-(2*ki*cosd(betaa))/d //proportional constant
disp(kp,'the proportional  constant is')
hc=syslin('c', 2.02*(s+1.19)/s)//transfer function of PD controller is kpof +kd*s
disp(hc,'transfer function of PI controller is')
hcmp=syslin('c',h*hc)//transfer function compensated system
disp(hcmp,'transfer function  compensated system ')
```

Scilab code Exa 6.16 PID CONTROLLER

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
```

```
//Scilab version 6.0.0
//operating systems windows 10
// Example 6.15

clc;
clear;
s=poly(0,'s')
//dominent pole sd=-zeta*w=%i*w*sqrt(1-zeta^2)
zeta=0.8//damping ratio
w=2.5//natural frequency of osciilation in rad/sec
sd=(-zeta*w)+((%i*w)*sqrt(1-zeta^2))
disp(sd,'the dominennt pole is')
d=abs(sd)
disp(d,'the value of d is ')
betaa=phasemag(sd)
disp(betaa,'the value of betaa is;')
h=syslin('c',75/(s+1)*(s+3)*(s+8))//given tranfer function G(s)
//find magnitude and phase of G(s) at s=sd
a=75/((1+sd)*(3+sd)*(8+sd))
ad=abs(a)
disp(ad,'the value of ad is')
phid=phasemag(a)
disp(phid,'the value of phid is')
ess=0.08//steady state error
kv=1/ess//velocity error constant
//the transfer function of PID controller is Gc(s)=kp+kd*s+ki/s
//by definition of velocity error constant applying s=0 in S*Gc(s)*G(s)
ki=12.5/3.125//integeral constant
kd=sind(phid)/(d*ad*sind(betaa))+(ki/(d^2)) //derivative constant
disp(kd,'the derivative  constant is')
kp=(-sind(betaa+phid))/(ad*sind(betaa))-(2*ki*cosd(betaa))/d //proportional constant
disp(kp,'the proportional  constant is')
hc=syslin('c', 0.68*(s^2+4.26*s+5.88)/s)//transfer
```

```
    function of PID controller is kpof +kd*s
disp(hc,'transfer function of PI controller is')
hcmp=syslin('c',h*hc)//transfer function
    compensated system
disp(hcmp,'transfer function  compensated system ')
```

Chapter 7

STATE SPACE ANALYSIS

Scilab code Exa 7.10 STATE MODEL

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 7.10

clc;
clear;
s=%s
p=poly([10],'s','coeff')
q=poly([1 2 4 1],'s','coeff')
sm=cont_frm(p,q)
disp(sm,'the state model in matrix form is')
```

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

Scilab code Exa 7.11 STATE MODEL

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 7.11

clc;
clear;
s=%s
p=poly([40 10],'s','coeff')
q=poly([0 3 4 1],'s','coeff')
sm=cont_frm(p,q)
disp(sm,'the state model in matrix form is')
```

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

Scilab code Exa 7.12 STATE MODEL

```
//control systems by Nagoor Kani A
//Edition 3
//Year of publication 2015
//Scilab version 6.0.0
//operating systems windows 10
// Example 7.12

clc;
clear;
s=%s
h=syslin('c',(2*(s+5))/((s+2)*(s+3)*(s+4)))
disp(h,'thr transfer function is')
ss=tf2ss(h)
disp(ss,'the state space model is')
[Ac,Bc,U,ind]=canon(ss(2),ss(3))
```

```
disp(Ac,Bc,U,ind)
```

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