Scilab Manual for Lab Practices II (M.E. Instrumentation & Control)

by Prof Deepti Khimani
Instrumentation Engineering
VESIT[1]

Solutions provided by
Prof Mrs. Deepti Khimani
Instrumentation Engineering
Mumbai University

March 29, 2026

[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

To study various built in functions for finding determinant, inverse, eigen values and eigen vectors.

Scilab code Solution 1.1 Lab01

```
1 // Lab01: To study various built in functions for
    finding determinant,
2 // inverse, eigen values and eigen vectors.
3
4 //                scilab - 5.5.2
5 //                Operating System : Windows 7, 32-bit
6
7 //
    ------------------------------------------------------------------------------
8
9 //Clean the environment
10 close;
11 clear;
12 clc;
```

```
13
14 //   ---------------------------------------------------------------------------
15
16 // Ex1.
17 //Declaration of given matrix
18 A=[3 2 0 ; 2 -1 7; 5 4 9];
19 disp(A,"Matirx A=","Ex1.:")
20
21 //Built-in fuctions to compute the determinant and
      inverse of a given matrix.
22 detA=det(A);
23 disp(detA, 'Determinant of A is=')
24 invA=inv(A);
25 disp(invA, 'Inverse of A is=')
26
27 //Built-in fuction to find eigen values of a given
      matrix.
28 eig_val=spec(A)
29 disp(eig_val, 'Eigen values of matrix A are=')
30
31 [eig_vec eig_val]=spec(A)
32 disp(eig_vec,'Eigen vector form of matrix A is')
33 disp(eig_val,'Diagonal form of eigen values of
      matrix A')
34
35 //   ---------------------------------------------------------------------------
36
37 // Ex.2
38 //Declaration of given matrix
39 B=[-4 2 5; 7 -1 6; 2 3 7];
40 disp(B,"Matirx B=","Ex2.:")
41
42 //Built-in fuction to compute the determinant and
      inverse of a given matrix.
```

```
43 detB=det(B);
44 disp(detB, 'Determinant of B is=')
45 invB=inv(B);
46 disp(invB, 'Inverse of B is=')
47
48 //Built-in fuction to find eigen values of a given
     matrix.
49 eig_val=spec(B)
50 disp(eig_val, 'Eigen values of matrix B are=')
51
52 [eig_vec eig_val]=spec(B)
53 disp(eig_vec,'Eigen vector form of matrix B is')
54 disp(eig_val,'Diagonal form of eigen values of
     matrix B')
55
56 //
```

Experiment: 2

Obtain 1, 2, infinity norm and reduced row echeolon form of a given matrix.

Scilab code Solution 2.1 Lab02

```
1 // Lab02: Obtain 1, 2, infinity norm and reduced row
      echelon form
2 // of a given matrix.
3
4 //                scilab - 5.5.2
5 //                Operating System : Windows 7, 32-bit
6 //
```

```
7
8 //Clean the environment
9 close;
10 clear;
11 clc;
12
13 //
```

```
14
15 // Ex1.
16 //Declaration for given matrix
17 A=[0 1 0; 3 0 2; -12 7 -6];
18 disp(A,"Matirx A=","Ex1.:")
19
20 //1-norm of matrix
21 Nrm_1=norm(A,1);
22 disp(Nrm_1,'1-norm of a given matrix is:')
23
24 //2-norm of matrix
25 Nrm_2=norm(A,2);
26 disp(Nrm_2,'2-norm of a given matrix is:')
27
28 //inf. norm of matrix
29 Nrm_inf=norm(A,'inf');
30 disp(Nrm_inf,'Infinity norm of a given matrix is')
31
32 // Reduced row Echeolon Form of a matrix
33 Re=rref(A);
34 disp(Re,'Reduced row Echeolon Form of A is =')
35
36 //
   ---------------------------------------------------------------------------
37
38 // Ex2.
39 //Declaration for given matrix
40 B=[2 3 4 5; -7 4 3 -2; 0 0 1 3; 1 0 -5 -4];
41 disp(B,"Matirx B=","Ex2.:")
42
43 //1-norm of matrix
44 Nrm_1=norm(B,1);
45 disp(Nrm_1,,'Norm 1 of a given matrix is')
46
47 //2-norm of matrix
48 Nrm_2=norm(B,2);
```

```
49 disp(Nrm_2,,'Norm 2 of a given matrix is')
50
51 //inf. norm of matrix
52 Nrm_inf=norm(B,'inf');
53 disp(Nrm_inf,,'Infinity norm of a given matrix is')
54
55 // Reduced row Echeolon Form of a matrix
56 Re=rref(B);
57 disp(Re,'Reduced row Echeolon Form of B is =')
58
59 //
```

Experiment: 3

To perform LU Decomposition on a given matrix.

Scilab code Solution 3.1 Lab03

```
1 // Lab03: To perform LU Decomposition on a given
    matrix.
2
3 //                scilab - 5.5.2
4 //                Operating System : Windows 7, 32-bit
5 //
    ------------------------------------------------------------------------

6 //Clean the environment
7
8 close;
9 clear;
10 clc;
11
12 //
    ------------------------------------------------------------------------

13
14 // Ex1.
```

```
15 //Declaration for given matrix
16 A=[1 2 3; 2 3 4; 3 4 6];
17 disp(A,"Matirx A=","Ex1.:")
18
19 //LU decomposition
20 [La Ua]=lu(A);
21 disp(La,'Lower tringular matrix La=')
22 disp(Ua,'Upper tringular matrix Ua=')
23
24 //
   ---------------------------------------------------------------------------


25
26 // Ex2.
27 //Declaration for given matrix
28 B=[3 5 7 7; 98 7 42 122; 100 25 90 160; 234 67 91
      97];
29 disp(B,"Matirx B=","Ex2.:")
30
31 //LU decomposition
32 [Lb Ub]=lu(B);
33 disp(Lb,'Lower tringular matrix Lb=')
34 disp(Ub,'Upper tringular matrix Ub=')
35
36 //
   ---------------------------------------------------------------------------
```

Experiment: 4

To perform QR Decomposition on a given matrix.

Scilab code Solution 4.1 Lab04

```
1 // Lab04: To perform QR Decomposition on a given
    matrix.
2
3 //                scilab - 5.5.2
4 //                Operating System : Windows 7, 32-bit
5 //
    ------------------------------------------------------------------------
6
7 //Clean the environment
8 close;
9 clear;
10 clc;
11 //
    ------------------------------------------------------------------------
12
13 //Ex1.
14 // Declaration for given matrix
```

```
15 A=[3 6; -2 5];
16 disp(A,"Matirx A=","Ex1.:")
17
18 //QR decomposition
19 [Qa Ra]=qr(A);
20 disp(Qa,'Orthogonal matrix Qa=')
21 disp(Ra,'Upper tringular matrix Ra=')
22
23 //
```

```
24
25 //Ex2.
26 // Declaration for given matrix
27 B=[1 2 3 4; 1 3 3 5; 1 2 3 7];
28 disp(B,"Matirx B=","Ex2.:")
29
30 //QR decomposition
31 [Qb Rb]=qr(B);
32 disp(Qb,'Orthogonal matrix Qb=')
33 disp(Rb,'Upper tringular matrix Rb=')
34
35 //
```

Experiment: 5

To perform Single Value Decomposition on a given matrix.

Scilab code Solution 5.1 Lab05

```
1 // Lab05: To perform Singular Value Decomposition on
      a given matrix.
2
3 //                scilab - 5.5.2
4 //                Operating System : Windows 7, 32-bit
5 //
  ------------------------------------------------------------------------
6
7 //Clean the environment
8 close;
9 clear;
10 clc;
11
12 //
  ------------------------------------------------------------------------
```

```
13
14 //Ex1.
15 // Declaration for given matrix
16 A=[3 6; -2 5];
17 disp(A,"Matirx A=","Ex1.:")
18
19 //Singular value decomposition
20 [Ua Sa Va]=svd(A);
21 disp(Ua,'Orthogonal matrix Ua=')
22 disp(Va,'Orthogonal matrix Va=')
23 disp(Sa,'Singular value matrix Sa=')
24
25 //
```

```
26
27 //Ex2.
28 // Declaration of another matrix
29 B=[1 2 3 4; 1 3 3 5; 1 2 3 7];
30 disp(B,"Matirx B=","Ex2.:")
31
32 //Singular value decomposition
33 [Ub Sb Vb]=svd(B);
34 disp(Ub,'Orthogonal matrix Ub=')
35 disp(Vb,'Orthogonal matrix Vb=')
36 disp(Sb,'Singular value matrix Sb=')
37
38 //
```

Experiment: 6

To study Gram Schmidt Orthogonalization.

Scilab code Solution 6.1 Lab06

```
1 // Lab06: To study Gram Schmidt Orthogonalization.
2
3 //                scilab - 5.5.2
4 //                Operating System : Windows 7, 32-bit
5
6 //
    ------------------------------------------------------------------------
7
8 //Clean the environment
9 close;
10 clear;
11 clc;
12
13 //
    ------------------------------------------------------------------------
14
15 // Declaration of given matrix
```

```
16 a=[1 2 1; 0 2 1; 2 3 0; 1 1 1];
17 [m n]=size(a);
18 q=zeros(m,n);
19 r=zeros(n,n);
20
21 for j=1:n
22     v=a(:,j)
23     for i=1:j-1
24         r(i,j)=q(:,i)'*a(:,j)
25         v=v-r(i,j)*q(:,i)
26     end
27     r(j,j)=norm(v)
28     q(:,j)=v/r(j,j)
29 end
30
31 disp(v,'v=')
32 disp(r,'r=')
33 disp(q,'q=')
34
35 //
```

Experiment: 7

To compare closed loop response of a given system using P, PI and PID controller.

Scilab code Solution 7.1 Lab07

```
1 // Lab07: To compare closed loop response of a given
      system using
2 // P, PI and PID controller.
3
4 //                scilab - 5.5.2
5 //                Operating System : Windows 7, 32-bit
6 //
```

```
7
8 //Clean the environment
9 xdel(winsid());//close all graphics Windows clear ;
10 clear;
11 clc;
12
13 //
```

```
14 //// Transfer function
15 s=%s; // or
16 s=poly(0,'s');
17 sys=syslin('c',(1)/(s^2+10*s+20))
18
19 //step response
20 t=0:0.05:2.5;
21 v=csim('step',t,sys);
22 plot2d(t,v,2)
23
24 //Title, labels and grid to the figure
25 //figure handel settings
26 f=get("current_figure"); //Current figure handle
27 f.background=8; //make the figure window background
      white
28 l=f.children(1);
29 l.background=8 ;//make the text background white
30 id=color('grey');
31 xgrid(id);
32 //custom script for setting figure properties
33 title('Response of given system to a step','fontsize
      ',3)
34 xlabel('Time t (sec.)','fontsize',2)
35 ylabel('Amplitude','fontsize',2)
36
37 //
   ------------------------------------------------------------------------


38
39 // Response of the system with proportional
      controller
40 Kp=10;
41 sys_P=syslin('c',(Kp)/(s^2+10*s+20+Kp));
42
43 //step response
44 figure;
45 t=0:0.05:2.5;
```

```
46 v=csim('step',t,sys_P);
47 plot2d(t,v,2)
48
49 //Title, labels and grid to the figure
50 //figure handel settings
51 f=get("current_figure"); //Current figure handle
52 f.background=8; //make the figure window background
      white
53 l=f.children(1);
54 l.background=8 ;//make the text background white
55 id=color('grey');
56 xgrid(id);
57 title('Response of given system with Proportional
      controller','fontsize',3)
58 xlabel('Time t (sec.)','fontsize',2)
59 ylabel('Amplitude','fontsize',2)
60 xstring(1,0.31,"Steady state error=(1-0.335)
      *100=66.5%")
61
62 //
     ------------------------------------------------------------------------

63 // Response of the system with proportional plus
      integral (PI) controller
64 Kp=25;
65 Ki=60;
66 sys_PI=syslin('c',(Kp*s+Ki)/(s^3+10*s^2+(20+Kp)*s+Ki
      ));
67
68 //step response
69 figure;
70 t=0:0.05:2.5;
71 v=csim('step',t,sys_PI);
72 plot2d(t,v,2)
73
74 //Title, labels and grid to the figure
75 //figure handel settings
76 f=get("current_figure"); //Current figure handle
```

```
77 f.background=8; //make the figure window background
      white
78 l=f.children(1);
79 l.background=8 ;//make the text background white
80 id=color('grey');
81 xgrid(id);
82 title('Response of given system with Proportional
      plus integral...
83  controller','fontsize',3)
84 xlabel('Time t (sec.)','fontsize',2)
85 ylabel('Amplitude','fontsize',2)
86 xstring(1,0.93,"Steady state error=(1-1)*100=0%")
87
88 //
   ------------------------------------------------------------------------


89 // Response of the system with proportional plus
      integral plus
90 //derivative (PID) controller
91 Kp=25;
92 Ki=60;
93 Kd=1;
94 sys_PID=syslin('c',(Kd*s^2+Kp*s+Ki)/(s^3+(10+Kd)*s
      ^2+(20+Kp)*s+Ki));
95
96 //step response
97 figure;
98 t=0:0.05:2.5;
99 v=csim('step',t,sys_PID);
100 plot2d(t,v,2)
101
102 //Title, labels and grid to the figure
103 //figure handel settings
104 f=get("current_figure"); //Current figure handle
105 f.background=8; //make the figure window background
      white
106 l=f.children(1);
107 l.background=8 ;//make the text background white
```

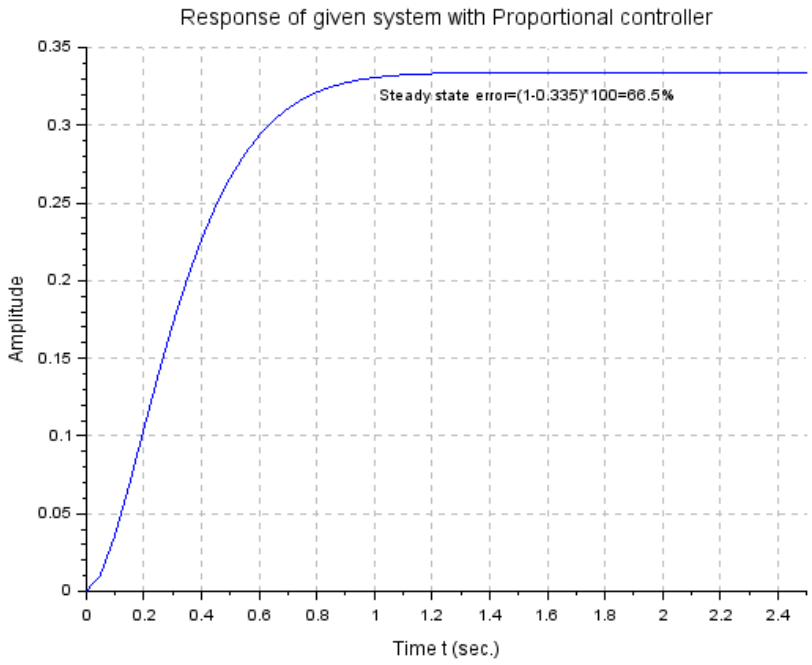

Figure 7.1: Lab07

```
108 id=color('grey');
109 xgrid(id);
110
111 title('Response of given system with Proportional
      plus integral...
112  plus derivative controller','fontsize',3)
113 xlabel('Time t (sec.)','fontsize',2)
114 ylabel('Amplitude','fontsize',2)
115 xstring(1,0.93,"Settling time improved to 1.6 sec.")
116 //
```

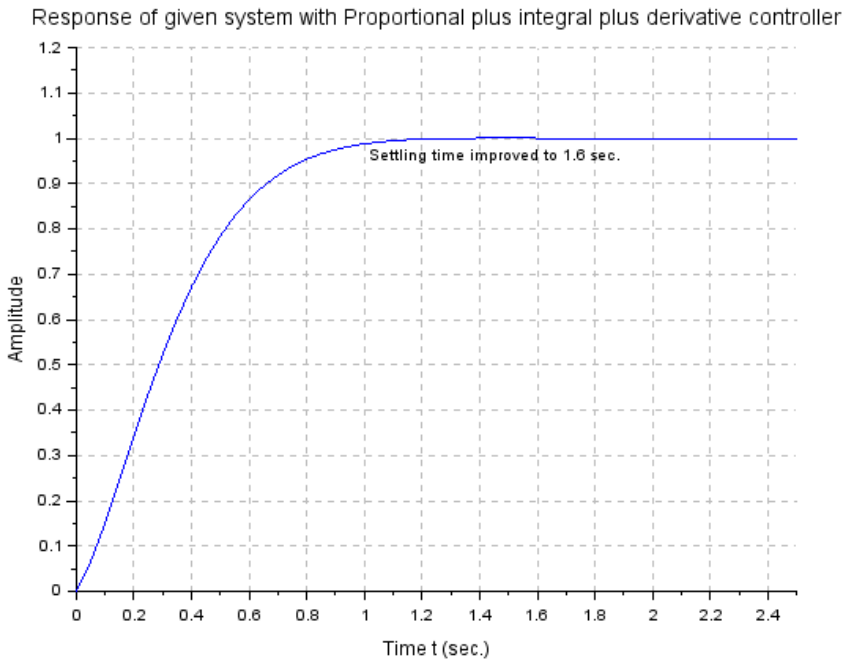

Figure 7.2: Lab07

Experiment: 8

Design the controller for a given system using Model Reference Adaptive Control approach.

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

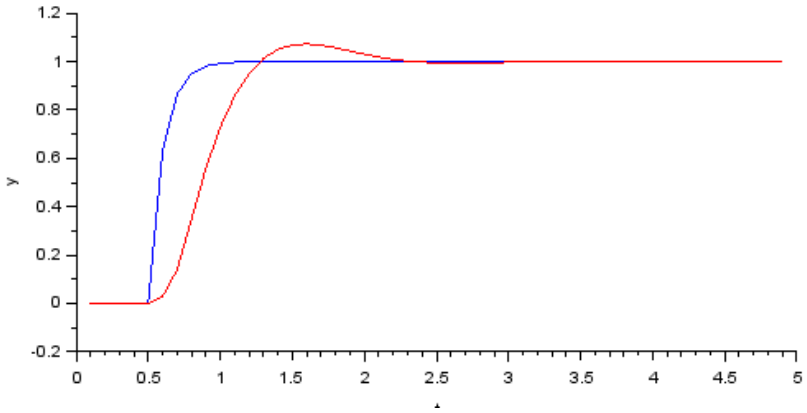

Figure 8.1: Lab08

Experiment: 9

Design the controller for a given system using Lyapunov design approach.

Scilab code Solution 9.1 Lab09

```
1 // Lab09: Design the controller for a given system
     using
2 // Lyapunov design approach.
3
4
5 //                scilab - 5.5.2
6 //                Operating System : Windows 7, 32-bit
7 //
  ------------------------------------------------------------------------------


8
9 //Clean the environment
10 xdel(winsid());//close all graphics Windows clear ;
11 clear;
12 clc;
13
```

```
14 //
   ------------------------------------------------------------------------

15 //system dicription
16 A=[0 1;0.5 -0.5]
17 B=[0 1]';
18 C=[1 0];
19 D=0;
20
21 //system model
22
23 sys=syslin ('c',A,B,C,D);
24 //
   ------------------------------------------------------------------------

25
26 //controller design
27 //here alpha1=1.5, alpha2=2;
28 alpha1=1.5;
29 alpha2=2;
30 b2=2;
31 //by design
32 k1=1+alpha1;
33 k2=abs(alpha2)+3 // condition is k2>|alpha2|
34 K=inv(b2)*[k1 k2];
35
36 //
   ------------------------------------------------------------------------

37 //closed loop system model
38
39 Ac=A-B*K
40 sysc=syslin ('c',A-B*K,B,C,D);
41
42 //
   ------------------------------------------------------------------------

43
```

```
44 //Simulation
45
46 x0=[1 0.5]'; //initial Condition
47 t=0:0.1:5; //simulation time
48 u=zeros(1,length(t)); //no input
49 [y x]=csim(u,t,sys,x0); //open loop system
50
51 tc=0:0.1:20; //simulation time
52 uc=zeros(1,length(tc)); //no input
53 [yc xc]=csim(uc,tc,sysc,x0); //closed loop system
54
55 //
   ------------------------------------------------------------------------


56
57 //Open and closed loop state responses
58 plot(t',x'); //open loop
59
60 //Title, labels and grid to the figure
61
62 //figure handel settings
63 f=get("current_figure"); //Current figure handle
64 f.background=8; //make the figure window background
      white
65 l=f.children(1);
66 l.background=8 ;//make the text background white
67 id=color('grey');
68 xgrid(id);
69
70 title('Open loop state responses','fontsize',3)
71 xlabel('$t (sec.)$','fontsize',3)
72 ylabel (["$x_1(t)$","$x_2(t)$"] , 'fontsize',3)
73 h=legend("$x_1(t)$","$x_2(t)$",4);
74 h.font_size=3;
75 h.fill_mode='off'
76
77 figure,
78 plot(tc',xc'); //closed loop
```

```
79 //Title, labels and grid to the figure
80 exec .\fig_settings.sci; // custom script for
      setting figure properties
81 title('Closed loop state responses','fontsize',3)
82 xlabel('$t (sec.)$','fontsize',3)
83 ylabel (["$x_1(t)$","$x_2(t)$"] , 'fontsize',3)
84 h=legend("$x_1(t)$","$x_2(t)$");
85 h.font_size=3;
86 h.fill_mode='off'
87
88 //
   ------------------------------------------------------------------------


89
90 //Open and closed loop state trajectories
91 figure,
92 plot(x(1,:),x(2,:),'r--');
93 plot(xc(1,:),xc(2,:));
94 zoom_rect([-1 -1 2 3])
95
96 //Title, labels and grid to the figure
97
98 //figure handel settings
99 f=get("current_figure"); //Current figure handle
100 f.background=8; //make the figure window background
      white
101 l=f.children(1);
102 l.background=8 ;//make the text background white
103 id=color('grey');
104 xgrid(id);
105 title('Open and Closed loop state trajectories','
      fontsize',3)
106 xlabel("$x_1(t)$",'fontsize',3);
107 ylabel ("$x_2(t)$", 'fontsize',3);
108 f=gca();
109 f.x_location = "origin"
110 f.y_location = "origin"
111 h=legend("Oen loop","Closed loop",4);
```

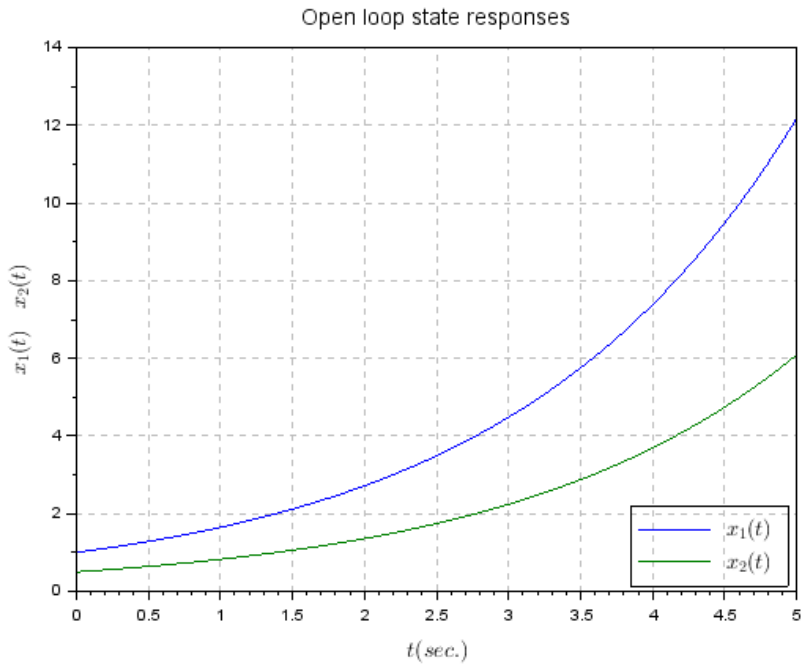

Figure 9.1: Lab09

```
112 h.font_size=2;
113 h.fill_mode='off'
114
115 //
```

---

check Appendix AP 1 for dependency:

`fig_settings.sci`

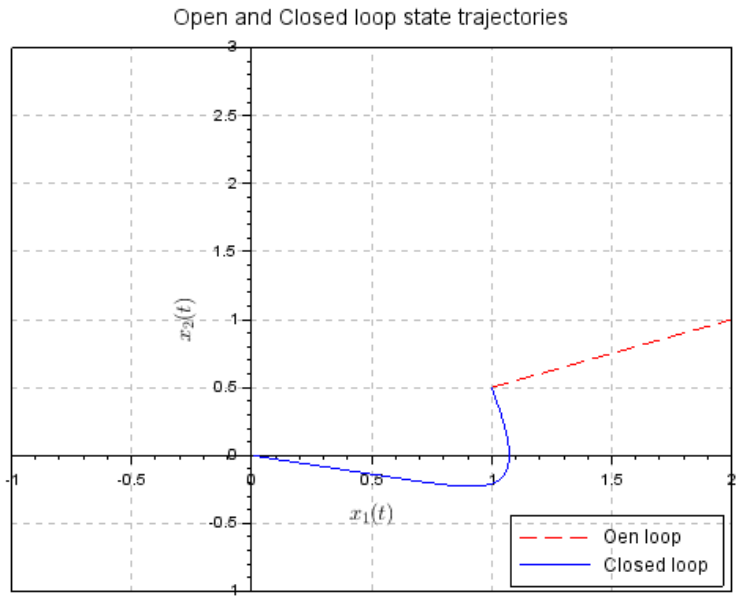

Figure 9.2: Lab09

Experiment: 10

Design the controller for a given system using variable structure control scheme.

Scilab code Solution 10.1 Lab10

```
1 // Lab10: Design the controller for a given system
     using
2 // variable structure control scheme.
3
4
5 //                scilab - 5.5.1
6 //                Operating System : Windows 7, 32-bit
7 //
  ------------------------------------------------------------------------
8
9 //Clean the environment
10 xdel(winsid());//close all graphics Windows clear ;
11 clear;
12 clc ;
13
14 //
```

```
15
16 //system dicription
17 xi=0.7;
18 alpha=1;
19 c=-xi/2+sqrt(xi^2/4+alpha)
20
21 //
   ---------------------------------------------------------------------------

22
23 //Simulation
24 importXcosDiagram("Lab10.xcos")
25 xcos_simulate(scs_m,4);
26 scs_m.props.context
27
28 //
   ---------------------------------------------------------------------------

29
30 //State responses
31 figure,
32 plot(x1.time,x1.values,2);
33 plot(x2.time,x2.values,'r--');
34 //Title, labels and grid to the figure
35 exec .\fig_settings.sci; // custom script for
      setting figure properties
36 title('State responses of the system with VSC','
      fontsize',3)
37 xlabel("$t sec.$",'fontsize',3);
38 ylabel ("$x_1(t),\,x_2(t)$", 'fontsize',3);
39 h=legend("Oen loop","Closed loop",1);
40 h.font_size=2;
41 h.fill_mode='off'
42
43 //
   ---------------------------------------------------------------------------
```

```
44
45 //State trajectory
46 figure,
47 plot(x1.values,x2.values);
48 //Title, labels and grid to the figure
49 exec .\fig_settings.sci; // custom script for
      setting figure properties
50 title('State trajectory of the system with VSC','
      fontsize',3)
51 xlabel("$x_1$",'fontsize',3);
52 ylabel ("$x_2$", 'fontsize',3);
53 f=gca();
54 f.x_location = "origin"
55 f.y_location = "origin"
56
57 //
```

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

check Appendix AP 1 for dependency:

`fig_settings.sci`

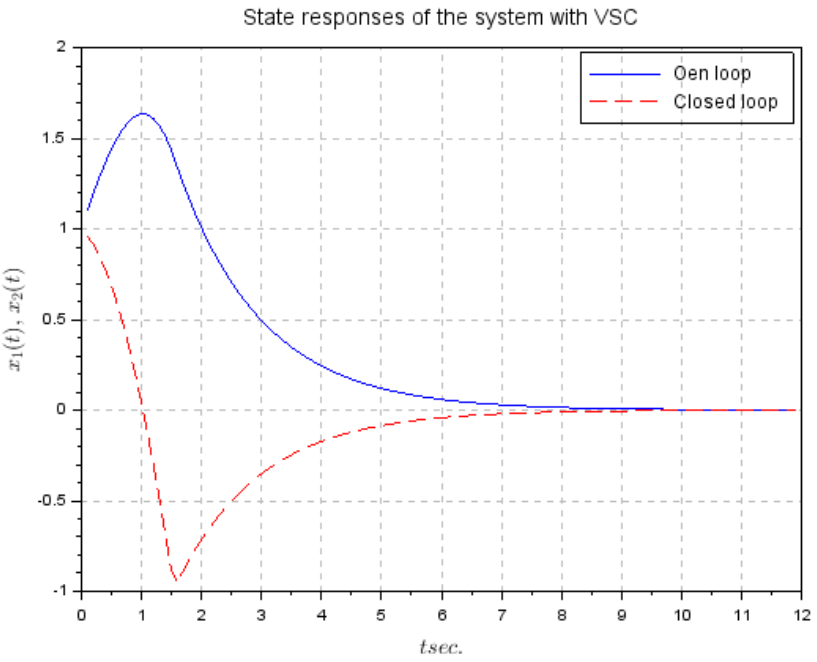

Figure 10.1: Lab10

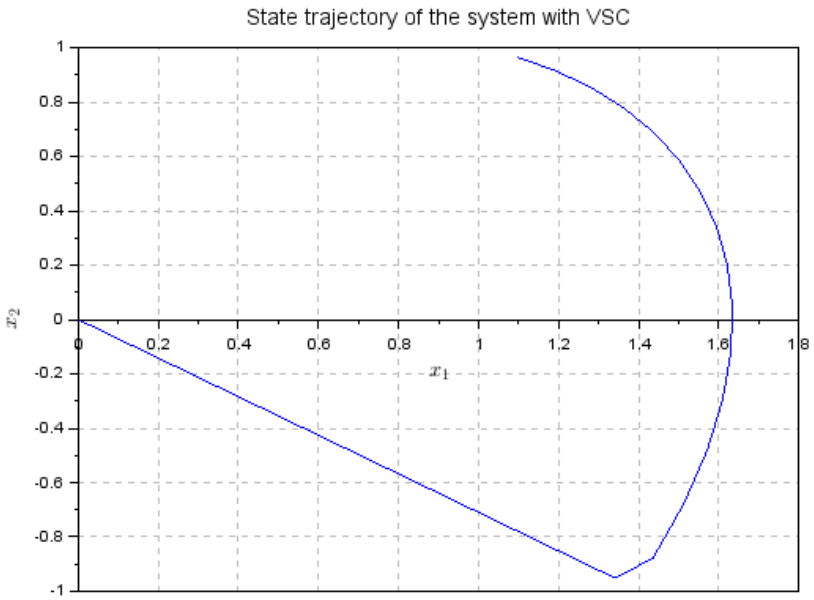

Figure 10.2: Lab10

Experiment: 11

To identify the parameters of a discrete time system using non recursive LS method.

Scilab code Solution 11.1 Lab11

```
1 // Lab.11: Identify the parameters of a discrete
     time system using
2 // non-recursive LS method.
3
4 xdel(winsid()); //close all graphics Windows clear ;
5 clear;
6 clc ;
7 //
```

```
8 //system model
9 s=poly(0,'s');
10 num=s+1;
11 den=s^2+1.6*s+1;
12 sys=syslin('c', num/den);
13 //
```

```
14 //Discretization of a given system
15 Ts=0.2;    //Sampling time
16 sysd=dscr(sys,Ts);
17 sysd_tf=ss2tf(sysd);
18 //
```

```
19 //System Identification
20 t=0:Ts:2;
21 //ramp inputis used as with the step input psi
      becomes singular,
22 //because input columns in psi are repeated
23 u=t;
24 y=flts(u,sysd_tf);
25
26 for i=1:1:5
27 Y(i)=y(i+3);
28 psi(i,1:4)=[-y(i+2) -y(i+1) u(i+2) u(i+1)];
29 end
30 theta=inv(psi'*psi)*psi'*Y;
31 //identified system
32 num=[theta([4,3])];
33 den=[theta([2,1]); 1];
34 num_z=poly(num,'z','coeff')
35 den_z=poly(den,'z','coeff')
36 sysd_id=num_z/den_z;
37 disp(sysd_tf,'sysd_tf=','The discrete model of the
      system is');
38 disp(sysd_tf,'sysd_id=','The identified system is');
39 //
```

Appendix

Scilab code AP 1 1 //figure handel settings
2 f=get("current_figure"); //Current figure handle
3 f.background=8; //make the figure window background
white
4 l=f.children(1);
5 l.background=8 ;//make the text background white
6 id=color('grey');
7 xgrid(id);

fig settings