Scilab Textbook Companion for Signals And Systems by A. Nagoorkani[1]

Created by
Sridhatta Jayaram Aithal
BACHELOR IN ENGINEERING(E&TC)
Others
A.C.PATIL COLLEGE OF ENGINEERING
College Teacher
None
Cross-Checked by
None

January 25, 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: Signals And Systems

Author: A. Nagoorkani

Publisher: Mcgraw Hill Education (india) Private Limited

Edition: 11

Year: 2015

ISBN: 978-0-07-015139-0

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

CONTINOUS TIME SIGNALS AND SYSTEMS

Scilab code Exa 2.1.C To check the signals are periodic or not

```
1 //example 2.1.c
2 //check the signal is periodic or not
3 clc ;
4 t =-15:0.01:15;
5 y =exp(-(2*%i*%pi*t)/7);
6 plot(t,y);
7 disp ( 'Plot  shows that given signal is  periodic
    with periodicity=7' ) ;
```

Scilab code Exa 2.1.D To check the signals are periodic or not

```
1 //example 2.1.d
```

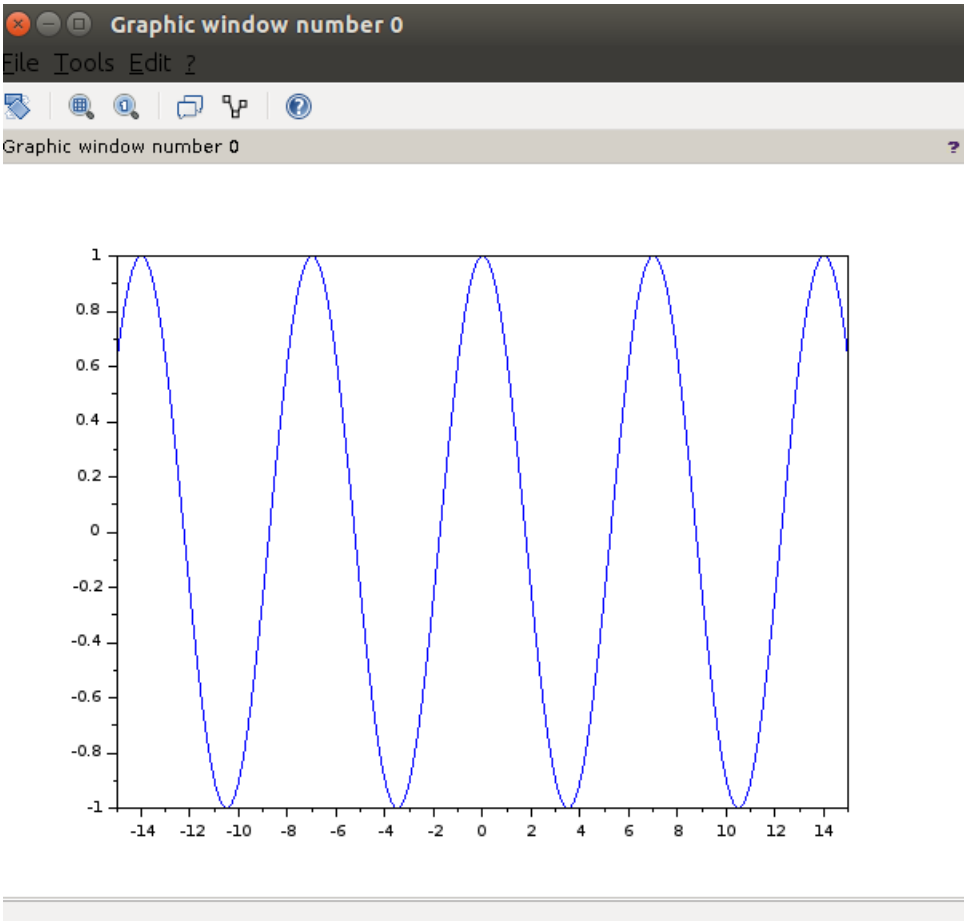

Figure 2.1: To check the signals are periodic or not

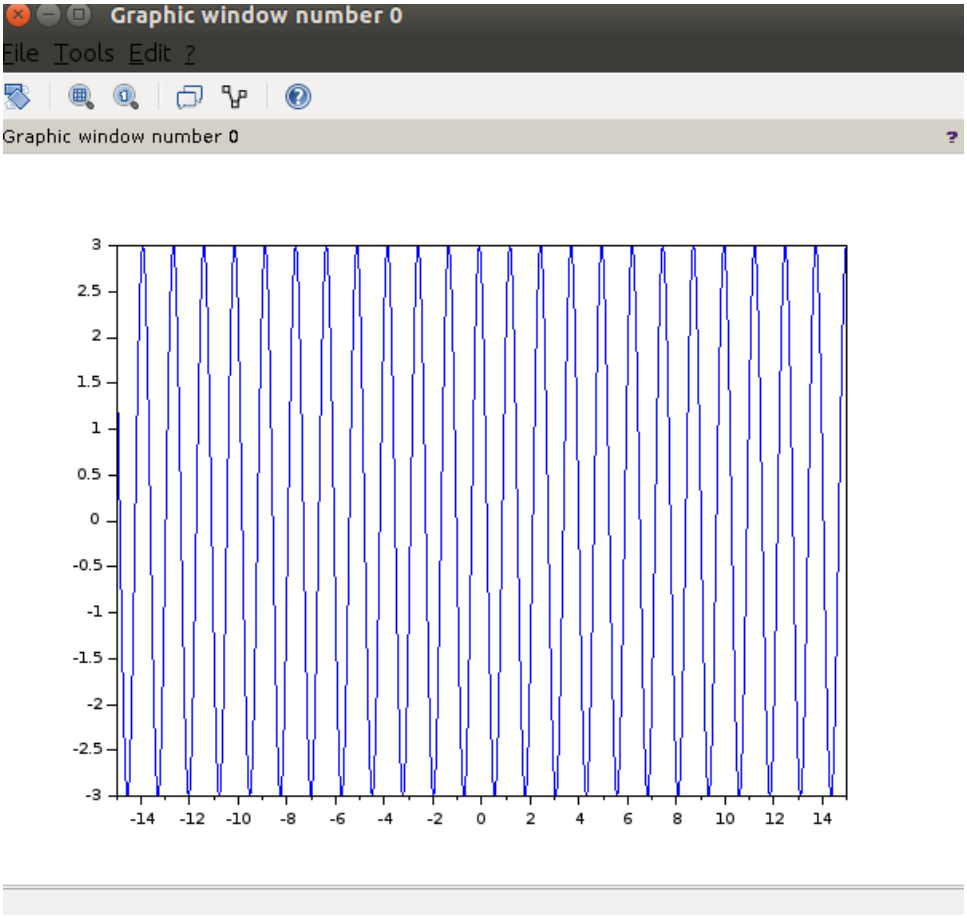

Figure 2.2: To check the signals are periodic or not

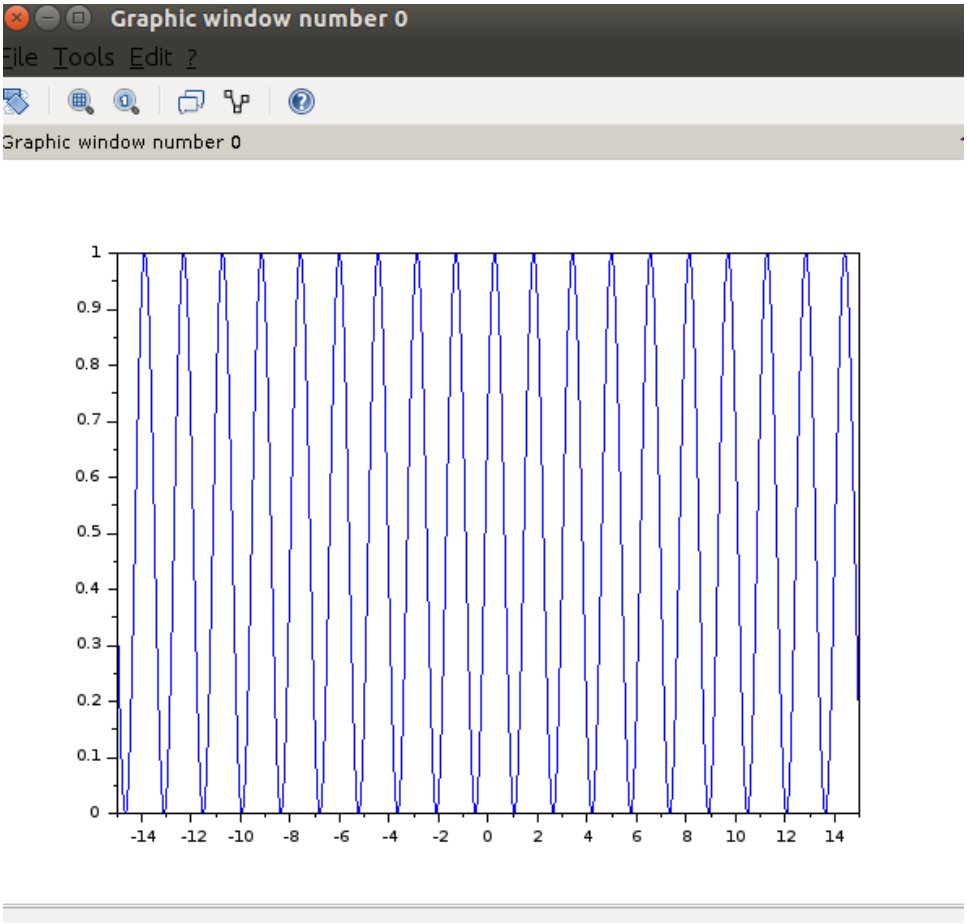

Figure 2.3: To check the signals are periodic or not

```
2 //check the signal is periodic or not
3 clc ;
4 t =-15:0.01:15;
5 y =3*(cos((5*t)+(%pi/6)));
6 plot(t,y);
7 disp ( 'Plot shows that given signal is periodic
    with periodicity=2%pi/5' ) ;
```

Scilab code Exa 2.1.E To check the signals are periodic or not

```
1 //example 2.1.e
2 //check the signal is periodic or not
3 clc ;
4 t =-15:0.01:15;
5 y =(1+cos(2*(2*t)-(%pi/3)))/(2);
6 plot(t,y);
7 disp ( 'Plot  shows that given signal is  periodic
    with periodicity=%pi/2' ) ;
```

Scilab code Exa 2.1.a To check the signals are periodic or not

```
1 //example 2.1.a
2 //check the signal is periodic or not
3 clc ;
4 t = -15:0.01:15;
5 y =2*(cos( t/4 ));
6 plot (t ,y ) ;
7 xtitle('plot of function 2*cos(t/4)')
8 xlabel('time-->')
9 disp ( 'Plot  shows that given signal is periodic
    with period T=8%pi' ) ;
```

Scilab code Exa 2.1.b To check the signals are periodic or not

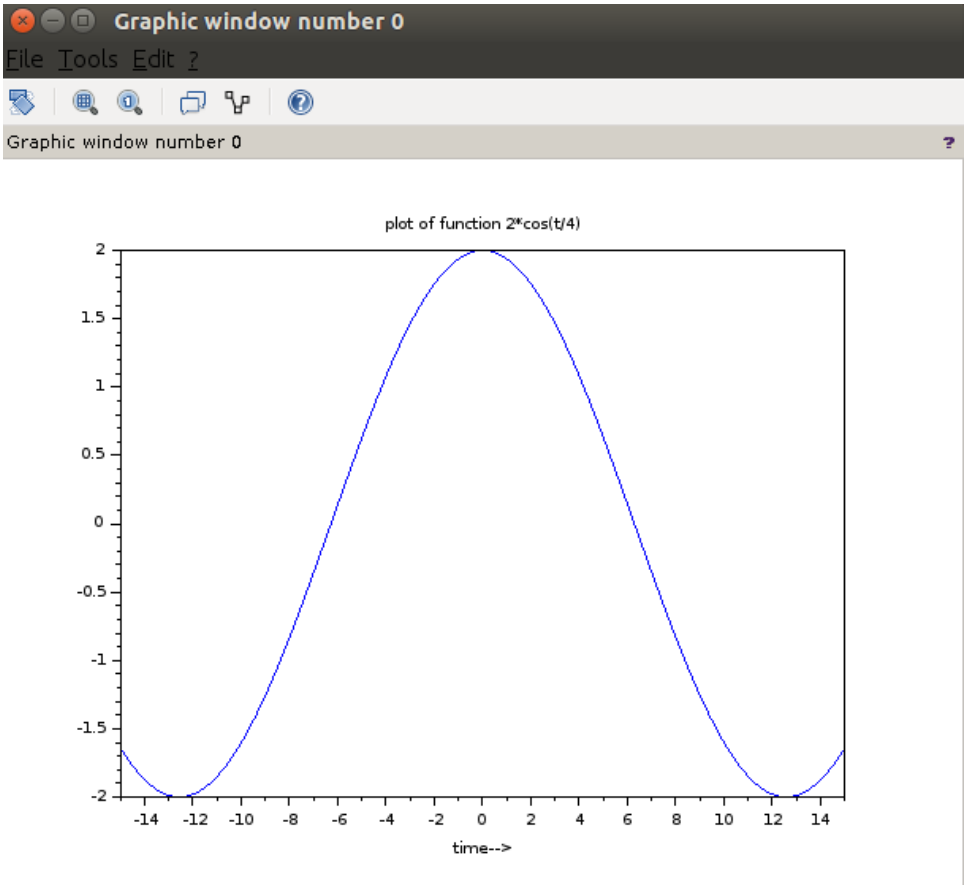

Figure 2.4: To check the signals are periodic or not

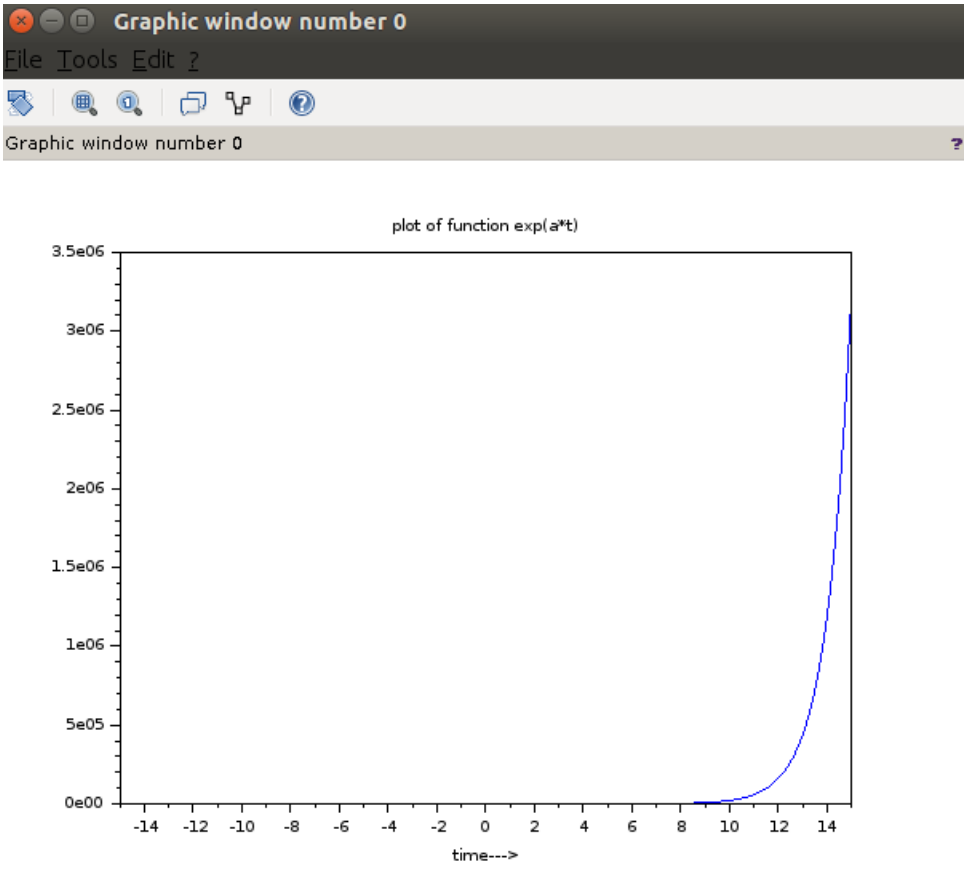

Figure 2.5: To check the signals are periodic or not

```
1 //example 2.1.b
2 //check the signal is periodic or not
3 clc ;
4 t =-15:0.01:15;
5 a=1;//assumed the value of a to be equal to 1
6 y =exp(a*t);
7 plot(t,y);
8 xtitle('plot of function exp(a*t)')
9 xlabel('time---->')
10 disp ( 'Plot  shows that given signal is not
     periodic' ) ;
```

Scilab code Exa 2.2.C To check the signals are periodic or not

```
1 //example 2.2.c
2 //check the signal is periodic or not
3 clc ;
4 t =-6:0.01:6;
5 y =(5*cos(4*%pi*t))+(3*sin(8*%pi*t));
6 plot(t,y);
7 disp ( 'Plot  shows that given signal is  periodic
     with periodicity=1/2' ) ;
```

Scilab code Exa 2.2.a To check the signals are periodic or not

```
1 //example 2.2.a
2 //check the signal is periodic or not
3 clc ;
4 t =-15:0.01:15;
```

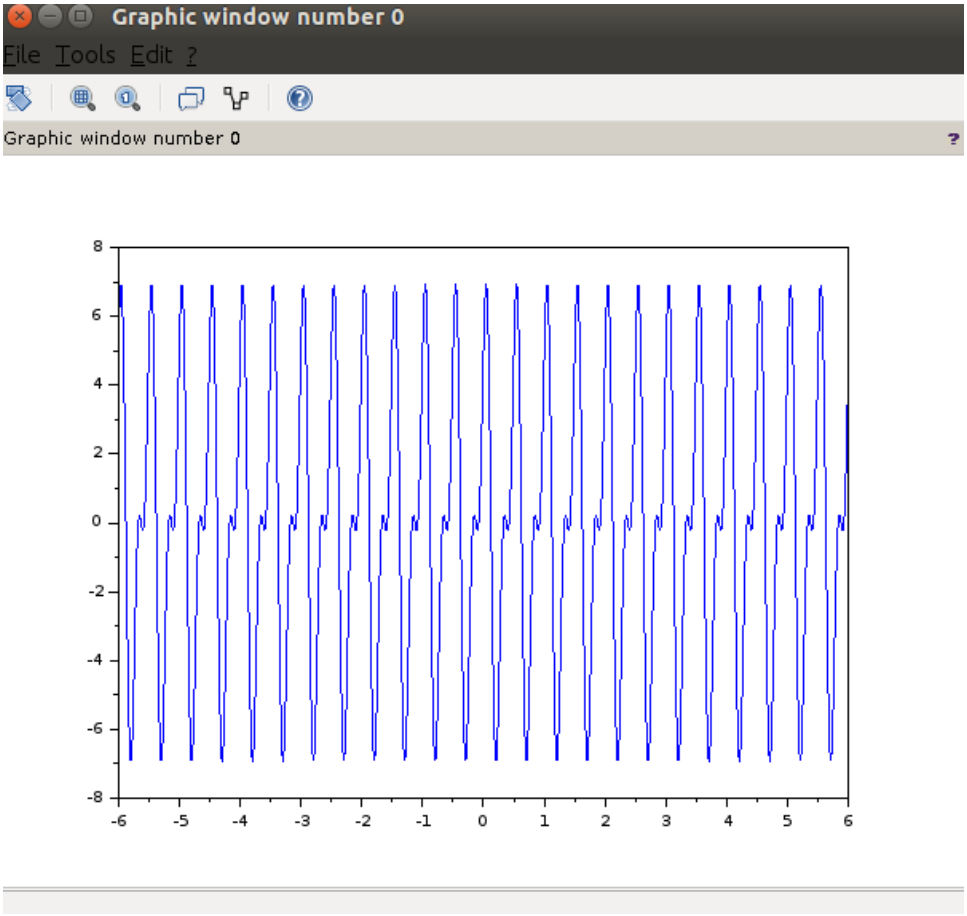

Figure 2.6: To check the signals are periodic or not

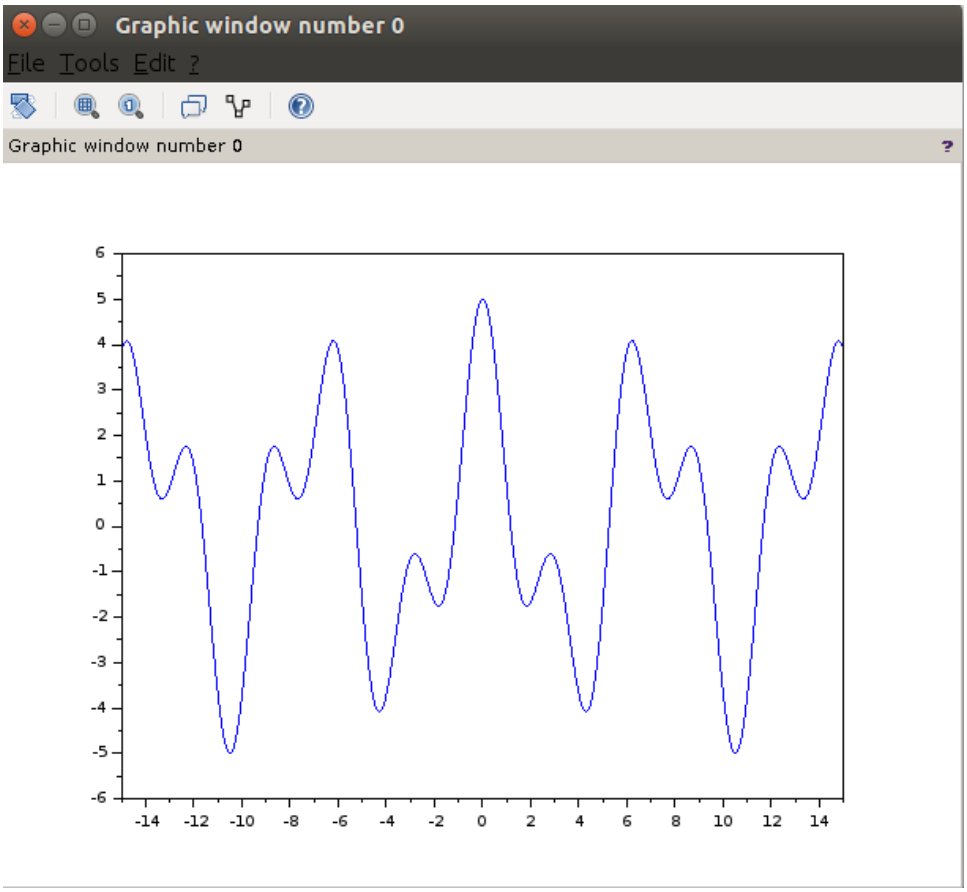

Figure 2.7: To check the signals are periodic or not

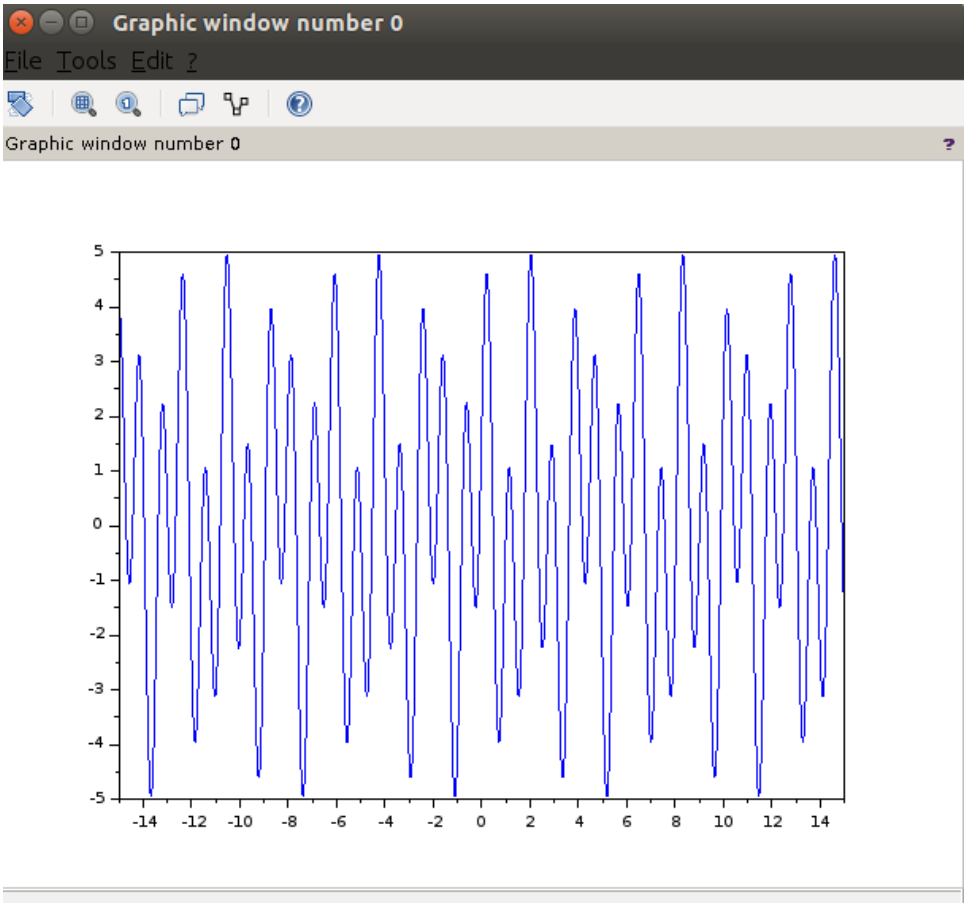

Figure 2.8: To check the signals are periodic or not

```
5 y =(2*cos((2)*(%pi)*t/3))+(3*cos((2)*(%pi)*t/7));
6 plot(t,y);
7 disp ( 'Plot shows that given signal is periodic
    with periodicity=21' ) ;
```

Scilab code Exa 2.2.b To check the signals are periodic or not

```
1 //example 2.2.b
2 //check the signal is periodic or not
3 clc ;
4 t =-15:0.01:15;
5 y =(2*cos(3*t))+(3*sin(7*t))
6 plot(t,y);
7 disp ( 'Plot  shows that given signal is  periodic
     with periodicity=2%pi' ) ;
```

Scilab code Exa 2.3.C To determine the even and odd part of the continous signal

```
1 clc;
2 t=-12:0.01:12
3 x=sin(2*t)+cos(t)+0.5*(sin(3*t)-sin(t))
4 h=-sin(2*t)+cos(t)-0.5*(sin(3*t)-sin(t))
5 e=cos(t)//(x+h)/2
6 o=(x-h)/2//sin(t)+0.5*(sin(3*t)-sin(t))
7 subplot(3,1,1)
8 plot(t,e)
9 xtitle('even signal')
10 subplot(3,1,2)
11 plot(t,o)
12 xtitle('odd signal')
```

Scilab code Exa 2.3.a To determin the even and odd part of the continous signal

```
1 //ex_2.3.a even and odd signals of x(t)
2 clear;
3 clc;
4 close;
5 t = 0:0.01:5;
6 x=exp(t)
7 figure
```

```
a=gca();
xtitle('x(t)')
plot2d(t,x)
figure
a=gca();
xtitle('even signal')
plot2d(t,x/2)
t1=-5:1/100:0;
plot2d(t1,x($:-1:1)/2)
a.y_location='origin'
figure
a=gca();
xtitle('odd signal')
plot2d(t,x/2)
t1=-5:1/100:0;
plot2d(t1,-x($:-1:1)/2)
a.y_location='origin'
a.x_location='origin'
```

Scilab code Exa 2.3.b To determine the even and odd part of the continous signal

```
//ex_2.3.b even and odd signals of x(t)
clear;
clc;
close;
t = 0:0.01:5;
x=3+(2*t)+5*((t)^2)
figure
a=gca();
xtitle('x(t)')
plot2d(t,x)
figure
a=gca();
xtitle('even signal')
plot2d(t,x/2)
```

```
15 t1=-5:1/100:0;
16 plot2d(t1,x($:-1:1)/2)
17 a.y_location='origin'
18 figure
19 a=gca();
20 xtitle('odd signal')
21 plot2d(t,x/2)
22 t1=-5:1/100:0;
23 plot2d(t1,-x($:-1:1)/2)
24 a.y_location='origin'
25 a.x_location='origin'
```

Scilab code Exa 2.4.A To determint the power and energy of the given signal

```
1 //Example 2.4.a
2 //Energy of the signal x(t)=(exp(-2*a*t)).u(t)
3 clc;
4 a=2;
5 E=integrate('exp(-a*t)^(2)','t',0,100)//Energy of
    the given signal
6 disp(E)
7 disp('AS ENERGY OF THE GIVEN SIGNAL IS FINITE HENCE
    THE GIVEN SIGNAL IS ENERGY SIGNAL');
```

Scilab code Exa 2.4.b To determine the even and odd part of the continous signal

```
1 //Example 2.4.a
2 //Energy of the signal x(t)=(exp(-2*a*t)).u(t)
3 clc;
4 P=integrate('1^(2)','t',(-100),100)/(2*100)//power
    of given signal t=100
5 disp(P)
```

```
6 disp('AS POWER OF THE GIVEN SIGNAL IS FINITE HENCE
    THE GIVEN SIGNAL IS POWER SIGNAL');
```

Scilab code Exa 2.4.c To determint the power and energy of the given signal

```
1 //Example 2.4.c
2 clc;
3 a=2;
4 P=(integrate('(3*cos(0.1*(%pi)*t))^(2)','t'
    ,-100,100)/(2*100))//power of given signal t=100
5 disp(P)
6 disp('AS POWER OF THE GIVEN SIGNAL IS FINITE HENCE
    THE GIVEN SIGNAL IS POWER SIGNAL');
```

Scilab code Exa 2.11.a to determine the system is whether time invariant or not

```
1 //Example 2.11.a
2 //to check the system is time invariant or not
3 clc ;
4 t0 =1;
5 T =10;
6 for t =1: T
7 x ( t ) =t;
8 y ( t ) =(2)*(t)*x(t) ;
9 end
10 inputshift = 2*(T)*x (T - t0 );
11 outputshift = y (T - t0 ) ;
12 if( inputshift == outputshift )
13 disp ( 'THE GIVEN SYSTEM I S TIME INVARIANT ' )
14 else
15 disp ( 'THE GIVEN SYSTEM I S TIME VARIANT ' ) ;
16 end
```

Scilab code Exa 2.11.b to determine the system is whether time invariant or not

```
1  //Example 2 . 2 11 b
2  clc ;
3  t0 =1;
4  T =10;
5  for t =1: T
6  x ( t ) =t;
7  y ( t ) =x(t)*sin(20*%pi*t) ;
8  end
9  inputshift = x(T-t0)*sin (20*%pi*(T) ) ;
10 outputshift = y (T - t0 ) ;
11 if( inputshift == outputshift )
12 disp ( 'THE GIVEN SYSTEM I S TIME INVARIANT ' )
13 else
14 disp ( 'THE GIVEN SYSTEM I S TIME VARIANT ' ) ;
15 end
```

Scilab code Exa 2.12.a to determine the system is whether time invariant or not

```
1  //Example 2    2.12.a
2  clc ;
3  t0 =1;
4  T =10;
5  for t =1: T
6  x ( t ) =t;
7  y ( t ) =(2)*exp(x(t)) ;
8  end
9  inputshift = 2*exp(x (T - t0))
10 outputshift = y (T - t0 )
11 if( inputshift == outputshift )
12 disp ( 'THE GIVEN SYSTEM I S TIME INVARIANT ' );
```

```
13 else
14 disp ( 'THE GIVEN SYSTEM I S TIME VARIANT ' ) ;
15 end
```

Scilab code Exa 2.12.b to determine the system is whether time invariant or not

```
1 //Example 2  2.12.b
2 clc ;
3 t0 =1;
4 T =10;
5 c=2;
6 for t =1: T
7 x ( t ) =t;
8 y ( t ) =x(t)+c ;
9 end
10 inputshift = x (T - t0)+c
11 outputshift = y (T - t0 )
12 if( inputshift == outputshift )
13 disp ( 'THE GIVEN SYSTEM I S TIME INVARIANT ' );
14 else
15 disp ( 'THE GIVEN SYSTEM I S TIME VARIANT ' ) ;
16 end
```

Scilab code Exa 2.12.c to determine the system is whether time invariant or not

```
1 //Example 2-12-c
2 clc ;
3 t0 =1;
4 T =10;
5 c=2;
6 for t =1: T
7 x ( t ) =t;
8 y ( t ) =3*(x(t))^(2);
```

```
9  end
10 inputshift = 3*(x(T - t0))^(2)
11 outputshift = y (T - t0 )
12 if( inputshift == outputshift )
13 disp ( 'THE GIVEN SYSTEM I S TIME INVARIANT ' );
14 else
15 disp ( 'THE GIVEN SYSTEM I S TIME VARIANT ' ) ;
16 end
```

Scilab code Exa 2.13.a to check whether the system is linear or non linear

```
1  //example 2_13_a
2  clear;
3  clc;
4  x1 = [1,1,1,1];
5  x2 = [2,2,2,2];
6  a = 1;
7  b = 1;
8  for t = 1:length(x1)
9    x3(t) = a*x1(t)+b*x2(t);
10 end
11 for t = 1:length(x1)
12   y1(t) = t*x1(t);
13   y2(t) = t*x2(t);
14   y3(t) = t*x3(t);
15 end
16 for t = 1:length(y1)
17   z(t) = a*y1(t)+b*y2(t);
18 end
19 count = 0;
20 for n =1:length(y1)
21   if(y3(t)== z(t))
22     count = count+1;
23   end
24 end
```

```
25 if(count == length(y3))
26    disp('Since It satisifies the superposition
         principle')
27    disp('The given system is a Linear system')
28    y3
29    z
30   else
31     disp('Since It does not satisify the
          superposition principle')
32     disp('The given system is a Non-Linear system')
33 end
```

Scilab code Exa 2.13.c to check whether the system is linear or non linear

```
1  //EXAMPLE 2.13.C
2  clear;
3  clc;
4  x1 = [1,1,1,1];
5  x2 = [2,2,2,2];
6  a = 1;
7  b = 1;
8  for t = 1:length(x1)
9    x3(t) = a*x1(t)+b*x2(t);
10 end
11 for t = 1:length(x1)
12   y1(t) = (x1(t)^2);
13   y2(t) = (x2(t)^2);
14   y3(t) = (x3(t)^2);
15 end
16 for t = 1:length(y1)
17   z(t) = a*y1(t)+b*y2(t);
18 end
19 count = 0;
20 for n =1:length(y1)
21   if(y3(t)== z(t))
```

```
22      count = count+1;
23    end
24  end
25  if(count == length(y3))
26     disp('Since It satisifies the superposition
          principle')
27     disp('The given system is a Linear system')
28     y3
29     z
30    else
31      disp('Since It does not satisify the
           superposition principle')
32      disp('The given system is a Non-Linear system')
33  end
```

Scilab code Exa 2.13.d to check whether the system is linear or non linear

```
1  //EXAMPLE 2.13.D
2  clear;
3  clc;
4  x1 = [1,1,1,1];
5  x2 = [2,2,2,2];
6  a = 1;
7  b = 1;
8  A=2
9  B=3;
10 for n = 1:length(x1)
11   x3(n) = a*x1(n)+b*x2(n);
12 end
13 for n = 1:length(x1)
14   y1(n) = A*x1(n)+B;
15   y2(n) = A*x2(n)+B;
16   y3(n) = A*x3(n)+B;
17 end
18 for n = 1:length(y1)
```

```
19    z(n) = a*y1(n)+b*y2(n);
20  end
21  count = 0;
22  for n =1:length(y1)
23    if(y3(n)== z(n))
24      count = count+1;
25    end
26  end
27  if(count == length(y3))
28     disp('Since It satisifies the superposition
          principle')
29     disp('The given system is a Linear system')
30     y3
31     z
32    else
33      disp('Since It does not satisify the
           superposition principle')
34      disp('The given system is a Non-Linear system')
35  end
```

Scilab code Exa 2.13.e to check whether the system is linear or non linear

```
1  //EXAMPLE 2.13.e
2  clear;
3  clc;
4  x1 = [1,1,1,1];
5  x2 = [2,2,2,2];
6  a = 1;
7  b = 1;
8  for t = 1:length(x1)
9    x3(t) = a*x1(t)+b*x2(t);
10 end
11 for t = 1:length(x1)
12   y1(t) = exp(x1(t));
13   y2(t) = exp(x2(t));
```

```
14    y3(t) = exp(x3(t));
15  end
16  for t = 1:length(y1)
17    z(t) = a*y1(t)+b*y2(t);
18  end
19  count = 0;
20  for n =1:length(y1)
21    if(y3(t)== z(t))
22      count = count+1;
23    end
24  end
25  if(count == length(y3))
26     disp('Since It satisifies the superposition
          principle')
27     disp('The given system is a Linear system')
28     y3
29     z
30    else
31      disp('Since It does not satisify the
           superposition principle')
32      disp('The given system is a Non-Linear system')
33  end
```

Scilab code Exa 2.17.a to test the stability of the system

```
1  //EXAMPLE 2.17.A
2  clc;
3  x=[1,1,1,1]
4  t=-1:0:1;
5  y(t)=cos(x(t));
6  disp('the max val of cos function is');
7  disp(cos(0));
8  disp('the min val of cos function is');
9  disp(cos(%pi));
10 disp('HENCE THE GIVEN SYSTEM IS BOUNDED IN -1 TO 1
```

```
   HENCE THE GIVEN SYSTEM IS STABLE');
```

Scilab code Exa 2.18.a to test the stability of the system

```
1 //Example 2.18.a
2 clc;
3 P=integrate('(exp(-5*t))','t',0,100)
4 E=integrate('(exp(5*t))','t',-100,0)
5 disp(P+E)
6 disp('AS THE INTEGRATION PRODUCT IS CONSTANT HENCE
   THE SYSTEM IS STABLE');
```

Scilab code Exa 2.18.c to test the stability of the system

```
1 //Example 2.18.C
2 clc;
3 P=integrate('(exp(-4*t))','t',0,100)
4 disp(P)
5 disp('AS THE INTEGRATION PRODUCT IS CONSTANT HENCE
   THE SYSTEM IS STABLE');
```

Scilab code Exa 2.18.d to test the stability of the system

```
1 //Example 2.18.D
2 clc;
3 P=integrate('(t*exp(-3*t))','t',0,100)
4 disp(P)
5 disp('AS THE INTEGRATION PRODUCT IS CONSTANT HENCE
   THE SYSTEM IS STABLE');
```

Scilab code Exa 2.18.f to test the stability of the system

```
1 //Example 2.18.F
2 clc;
3 P=integrate('(exp(-t)*sin(t))','t',0,100)
4 disp(P)
5 disp('AS THE INTEGRATION PRODUCT IS CONSTANT HENCE
    THE SYSTEM IS STABLE');
```

Scilab code Exa 2.20.B to perform convolution of the causal signal

```
1 //EXAMPLE 2.20.B
2 clc;
3 x1=100;
4 x=integrate('exp(-2*t)*exp(-5*t)','t',0,x1);
5 disp(x);
6 disp('valid for t>=0');
```

Scilab code Exa 2.20.C to perform convolution of the causal signal

```
1 //EXAMPLE 2.20.C
2 clc;
3 x1=100;
4 x=integrate('t*exp(-5*t)','t',0,x1);
5 disp(x);
6 disp('valid for t>=0');
```

Scilab code Exa 2.20.D to perform convolution of the causal signal

```
1 //EXAMPLE 2.20.D
2 clc;
3 x1=100;
4 x=integrate('t*cos(t)','t',0,x1);
5 disp(x);
6 disp('valid for t>=0');
```

Scilab code Exa 2.20.a to perform convolution of the causal signal

```
1 //EXAMPLE 2.20.A
2 clc;
3 x1=100;
4 x=integrate('2*1','t',0,x1);
5 disp(x);
6 disp('valid for t>=0');
```

Scilab code Exa 2.21.B to determine the unit step response of the impulse response

```
1 //EXAMPLE 2.21.B
2 clc;
3 x1=10;
4 x=integrate('exp((-5)*t)','t',0,x1);
5 disp(x)
6 disp('valid for t>=0')
```

Scilab code Exa 2.21.C to determine the unit step response of the impulse response

```
1 //EXAMPLE 2.21.C
```

```
clc;
x1=10;
x=integrate('1','t',-2,x1);
disp(x)
disp('valid for t<=2')
```

Scilab code Exa 2.21.D to determine the unit step response of the impulse response

```
//EXAMPLE 2.21.D
clc;
x1=10;
x=integrate('1','t',2,x1);
disp(x)
disp('valid for t>=2')
```

Scilab code Exa 2.21.E to determine the unit step response of the impulse response

```
//EXAMPLE 2.21.E
clc;
x1=100;
x=integrate('1','t',-2,x1);
y=integrate('1','t',2,x1);
disp(x);
disp('valid for t>=-2 to t<=2');
disp(x+y);
disp('valid for t>=2');
```

Scilab code Exa 2.21.a to determine the unit step response of the impulse response

```
//EXAMPLE 2.21.A
```

```
clc;
x1=100;
x=integrate('3*t','t',0,x1);
disp(x);
disp('valid for t>=0');
```

Scilab code Exa 2.22.A TO PERFORM CONVOLUTION

```
//EXAMPLE 2.22.A
clc;
t=0:1:15;
t2=0:0.1:15
x1=exp((-3)*t2).*(t2>=0);
x2=t.*(t>=0);
subplot(3,1,1);plot(t2,x1);
xlabel('t');ylabel('x1(t)');
title('signal x1(t)');
subplot(3,1,2);plot(t,x2);
xlabel('t');ylabel('x2(t)');
title('signal x2(t)');
T1=length(x1);
T2=length(x2);
x3=convol(x1,x2);
t1=0:1:T1+T2-2;
subplot(3,1,3);plot(t1,x3);
xlabel('t');ylabel('x3(t)');
title('signal x3(t) =x1(t)*x2(t)');
```

Scilab code Exa 2.22.b TO PERFORM CONVOLUTION

```
//EXAMPLE 2.22.B
clc;
t=0:1:15;
```

```
t2=0:0.1:15
a=2;
x1=exp(-a*t2).*(t2>=0);
x2=t.*(t>=0);
subplot(3,1,1);plot(t2,x1);
xlabel('t');ylabel('x1(t)');
title('signal x1(t)');
subplot(3,1,2);plot(t,x2);
xlabel('t');ylabel('x2(t)');
title('signal x2(t)');
T1=length(x1);
T2=length(x2);
x3=convol(x1,x2);
t1=0:1:T1+T2-2;
subplot(3,1,3);plot(t1,x3);
xlabel('t');ylabel('x3(t)');
title('signal x3(t) =x1(t)*x2(t)');
```

Chapter 3

LAPLACE TRANSFORM

Scilab code Exa 3.1.A TO PERFORM LAPLACE TRANSFORM

```
1 //Example 3.1.A
2 clc;
3 Syms s t;
4 A=3
5 B=laplace(A,t,s)
6 disp(B)
```

Scilab code Exa 3.1.B TO PERFORM LAPLACE TRANSFORM

```
1 //EXAMPLE 3.1.B
2 clc;
3 Syms s t
4 B=laplace(t,t,s)
5 disp(B)
```

Scilab code Exa 3.1.C TO PERFORM LAPLACE TRANSFORM

```
1 //EXAMPLE 3.1.C
2 clc;
3 Syms s t
4 B=laplace(exp(-3*t),t,s)
5 disp(B)
```

Scilab code Exa 3.1.D TO PERFORM LAPLACE TRANSFORM

```
1 //EXAMPLE 3.1.D
2 clc;
3 Syms s t
4 B=laplace(-exp(-3*t),t,s)
5 disp(B)
```

Scilab code Exa 3.1.E TO PERFORM LAPLACE TRANSFORM

```
1 //EXAMPLE 3.1.E
2 clc;
3 Syms s t;
4 e=laplace(exp(-4*t),t,s)-laplace(exp(4*t),t,s)
5 disp(e)
```

Scilab code Exa 3.2.A TO PERFORM LAPLACE TRANSFORM

```
1 //EXAMPLE 3.2.A
2 clc;
3 Syms s t
4 w=2;
5 B=laplace(sin(w*t),t,s)
6 disp(B)
```

Scilab code Exa 3.2.b TO PERFORM LAPLACE TRANSFORM

```
1 //EXAMPLE 3.2.B
2 clc;
3 Syms s t
4 w=2;
5 B=laplace(cos(w*t),t,s)
6 disp(B)
```

Scilab code Exa 3.2.c TO PERFORM LAPLACE TRANSFORM

```
1 //EXAMPLE 3.2.C
2 clc;
3 Syms s t
4 w=2;
5 B=laplace(cosh(w*t),t,s)
6 disp(B)
```

Scilab code Exa 3.2.d TO PERFORM LAPLACE TRANSFORM

```
1 //EXAMPLE 3.2.D
2 clc;
3 Syms s t
4 w=2;
5 a=5;
6 F=exp(-a*t)*sin(w*t)
7 B=laplace(F,t,s)
8 disp(B)
```

Scilab code Exa 3.2.e TO PERFORM LAPLACE TRANSFORM

```
1 //EXAMPLE 3.2.E
2 clc;
3 Syms s t
4 w=2;
5 a=5;
6 F=exp(-a*t)*cos(w*t)
7 B=laplace(F,t,s)
8 disp(B)
```

Scilab code Exa 3.4 Laplacian Transform of the Given Equation

```
1 //Example 3.4
2 clc;
3 Syms s,x;
4 A=2;
5 B=A*(sin(x)*exp(-s*x))
6 C=integrate(B,x,0,%pi);
7 disp(C)
```

Scilab code Exa 3.7.a TO PERFORM LAPLACE TRANSFORM

```
1 //example 3.7.a
2 clc;
3 Syms s t
4 F=(t^(2)-2*t)*unit_step(t-1)
5 b=laplace(F,t,s)
6 disp(b)
```

Scilab code Exa 3.7.b TO PERFORM LAPLACE TRANSFORM

```
1 //example 3.7.b
2 clc;
3 Syms s t
4 a=5;
5 F=(t-a)*unit_step(t-a)
6 T=laplace(F,t,s)
7 disp(T)
```

Scilab code Exa 3.12 INVERSE LAPLACE TRANSFORM

```
1 //Example 3.12
2 clc;
3 Syms s,t;
4 Z=(2/(s*(s+1)*(s+2)))
5 i=ilt(Z,s,t);
6 disp(i);
```

Scilab code Exa 3.13 INVERSE LAPLACE TRANSFORM

```
1 //Example 3.13
2 clc;
3 Syms s,t;
4 I=2/((s)*(s+1)*(s+2)^(2));
5 i=ilt(I,s,t);
6 disp(i);
```

Scilab code Exa 3.14 INVERSE LAPLACE TRANSFORM

```
1 //Example 3.14
2 clc;
3 Syms s,t;
4 I=1/((s+2)*(((s)^(2))+s+1));
5 i=ilt(I,s,t);
6 disp(i);
```

Scilab code Exa 3.15 INVERSE LAPLACE TRANSFORM

```
1 //Example 3.15
2 clc;
3 Syms s,t;
4 I=4/((s^(2))*(s^(2)+16));
5 i=ilt(I,s,t);
6 disp(i);
```

Scilab code Exa 3.16.a INVERSE LAPLACE TRANSFORM

```
1 //Example 3.16.A
2 clc;
3 Syms s,t;
4 I=(3*s^(2)+8*s+23)/((s+3)*(s^(2)+2*s+10));
5 i=ilt(I,s,t);
6 disp(i);
```

Scilab code Exa 3.16.b INVERSE LAPLACE TRANSFORM

```
1 //Example 3.16.B
2 clc;
3 Syms s,t;
4 I=(8*(s^(2)))/((s+2)*(s+1)^(3));
5 i=ilt(I,s,t);
6 disp(i);
```

Scilab code Exa 3.22 convolution using laplace transform

```
1 //Example 3.22
2 clc;
3 Syms s t;
4 x=laplace(exp(-2*t)*cos(3*t),t,s);
5 y=laplace(4*sin(3*t),t,s);
6 z=x*y
7 i=ilt(z,s,t);
8 disp(i);
```

Scilab code Exa 3.24.a to determine the transfer function of the system

```
1 //Example 3.24.a
2 clc;
3 Syms s t;
4 x=laplace((2+t)*(exp(-3*t)),t,s);
5 disp(x);
```

Scilab code Exa 3.24.b to determine the transfer function of the system

```
1 //Example 3.24.B
2 clc;
3 Syms s t;
4 x=laplace((t^(2)-exp(-4*t)+exp(-7*t)),t,s);
5 disp(x);
```

Scilab code Exa 3.24.c to determine the transfer function of the system

```
1 //Example 3.24.C
2 clc;
3 Syms s t;
4 x=laplace((1+0.5*exp(-6*t)+0.2*exp(-3*t)),t,s);
5 disp(x);
```

Scilab code Exa 3.25.a to determine the transfer function of the system

```
1 //EXAMPLE 3.25.A
2 clc;
3 Syms s,t;
4 u=laplace(1,t,s)+laplace(exp(-2*t),t,s);
5 F=u*laplace(1,t,s)
6 disp(F);
```

Scilab code Exa 3.25.b to determine the transfer function of the system

```
1 //EXAMPLE 3.25.b
2 clc;
3 Syms s,t;
4 u=laplace((t)^(2),t,s)+laplace(t*exp(-4*t),t,s);
5 F=u*laplace(1,t,s)
6 disp(F);
```

Scilab code Exa 3.25.c to determine the transfer function of the system

```
1 //EXAMPLE 3.25.c
2 clc;
3 Syms s,t;
4 u=laplace(t,t,s)+laplace(sin(t),t,s);
5 F=u*laplace(1,t,s)
6 disp(F);
```

Scilab code Exa 3.26.a to find the impulse response from the transfer function

```
1 //Example 3.26.A
2 clc;
3 Syms s t;
4 F=1/(s^(2)*(s-2));
5 f=ilt(F,s,t);
6 disp(f);
```

Scilab code Exa 3.26.b to find the impulse response from the transfer function

```
1 //Example 3.26.B
2 clc;
3 Syms s t;
4 F=1/(s*(s+1)*(s-2));
5 f=ilt(F,s,t);
6 disp(f);
```

Scilab code Exa 3.26.c to find the impulse response from the transfer function

```
1 //Example 3.26.c
2 clc;
3 Syms s,t;
4 F=1/(s^(2)+s+1);
5 f=ilt(F,s,t);
6 disp(f);
```

Scilab code Exa 3.27.a output response of input and impulse response to the system

```
1 //Example 3.27.A
2 clc;
3 a=2;
4 Syms s t;
5 y=laplace(exp(-a*t),t,s);
6 z=1*y;
7 f=ilt(z,s,t);
8 disp(f);
```

Scilab code Exa 3.27.b output response of input and impulse response to the system

```
1 //Example 3.27.B
2 clc;
3 Syms s t;
4 x=laplace(exp(-2*t),t,s);
5 y=laplace(1,t,s);
6 z=x*y;
7 f=ilt(z,s,t);
8 disp(f);
```

Scilab code Exa 3.28.B to perform convolution of the causal signal

```
1 //Example 3.28.B
2 clc;
3 Syms s t;
4 x=laplace(exp(-2*t),t,s);
5 y=laplace(exp(-5*t),t,s);
6 z=x*y;
7 f=ilt(z,s,t);
8 disp(f);
```

Scilab code Exa 3.28.C to perform convolution of the causal signal

```
1 //Example 3.28.c
2 clc;
3 Syms s t;
4 x=laplace(t,t,s);
5 y=laplace(exp(-5*t),t,s);
6 z=x*y;
7 f=ilt(z,s,t);
8 disp(f);
```

Scilab code Exa 3.28.D to perform convolution of the causal signal

```
1 //Example 3.28.D
2 clc;
3 Syms s t;
4 x=laplace(cos(t),t,s);
5 y=laplace(t,t,s);
6 z=x*y;
7 f=ilt(z,s,t);
8 disp(f);
```

Scilab code Exa 3.28.a to perform convolution of the causal signal

```
1 //Example 3.28.A
2 clc;
3 Syms s t;
4 x=laplace(2,t,s);
5 y=laplace(1,t,s);
6 z=x*y;
7 f=ilt(z,s,t);
8 disp(f);
```

Scilab code Exa 3.29.A TO PERFORM DECONVOLUION OPERATION

```
1 //Example 3.29.A
2 clc;
3 Syms s t;
4 x=laplace(2*t,t,s);
5 y=laplace(1,t,s);
6 z=x/y;
7 f=ilt(z,s,t);
8 disp(f);
```

Scilab code Exa 3.29.B TO PERFORM DECONVOLUION OPERATION

```
1 //Example 3.29.b
2 clc;
3 Syms s t;
4 x=laplace(exp(-2*t),t,s);
5 y=laplace(exp(-5*t),t,s);
```

```
6  z=(1/3)*(x-y);
7  A=laplace(exp(-5*t),t,s);
8  B=z/A
9  f=ilt(B,s,t);
10 disp(f);
```

Scilab code Exa 3.29.C TO PERFORM DECONVOLUTION OPERATION

```
1  //Example 3.29.C
2  clc;
3  Syms s t;
4  x=laplace((1/25)*(exp(-5*t)+5*t-1),t,s);
5  y=laplace(exp(-5*t),t,s);
6  z=x/y;
7  f=ilt(z,s,t);
8  disp(f);
```

Scilab code Exa 3.29.D TO PERFORM DECONVOLUTION OPERATION

```
1  //Example 3.29.d
2  clc;
3  Syms s t;
4  x=laplace(1-cos(t),t,s);
5  y=laplace(t,t,s);
6  z=x/y;
7  f=ilt(z,s,t);
8  disp(f);
```

Scilab code Exa 3.30.A to determine the unit step response of the impulse response

```
1 //Example 3.30.A
2 clc;
3 Syms s t;
4 x=laplace(3*t,t,s);
5 y=laplace(1,t,s);
6 z=x*y;
7 f=ilt(z,s,t);
8 disp(f);
```

Scilab code Exa 3.30.B to determine the unit step response of the impulse response

```
1 //Example 3.30.B
2 clc;
3 Syms s t;
4 x=laplace(exp(-5*t),t,s);
5 y=laplace(1,t,s);
6 z=x*y;
7 f=ilt(z,s,t);
8 disp(f);
```

Chapter 4

FOURIER TRANSFORM OF CONTINUOS TIME SIGNAL

Scilab code Exa 4.13.A Determine the Fourier Transform of Continuos Time SIgnal

```
1 //Example 4.13.A
2 clc;
3 Syms o,x;
4 A=(1-x^2)/(exp(%i*o*x))
5 Y=integrate(A,x,-1,1)
6 disp(Y)
```

Scilab code Exa 4.13.B Determine the Fourier Transform of Continuos Time SIgnal

```
1 //Example 4.13.B
2 clc;
3 Syms o,x,a,o1;
4 A=(exp(-a*x)*cos(o1*x))/(exp(%i*o*x))
5 Y=integrate(A,x,-1,1)
6 disp(Y)
```

Scilab code Exa 4.14 Fourier Transform of Rectangular Pulse

```
1 //Example 4.14
2 clc;
3 Syms o,x,T;
4 A=1/(exp(%i*o*x));
5 Y=integrate(A,x,-1,1)
6 disp(Y)
```

Scilab code Exa 4.15 Fourier Transform of Given Equation

```
1 //Example 4.15
2 clc;
3 Syms o,x,T;
4 A=(1+x/2)/(exp(%i*o*x));
5 B=(1-x/2)/(exp(%i*o*x));
6 Y=integrate(A,x,-2,2)
7 L=integrate(B,x,-2,2)
8 Z=Y+L;
9 disp(Z)
```

Chapter 6

DISCRETE TIME SIGNALS AND SYSTEMS

Scilab code Exa 6.4.A To check the signals are periodic or not

```
1 //example 6.4.A
2 //check the signal is periodic or not
3 clc ;
4 n=-15:0.01:15;
5 y =sin((6*(%pi)*n/7)+1);
6 xlabel('n')
7 ylabel('x(n)')
8 plot2d(n,y);
9 disp ( 'Plot  shows that given signal is periodic of
    fundamental period=7 samples' ) ;
```

Scilab code Exa 6.4.b To check the signals are periodic or not

```
1 //example 6_4_B
2 //check the signal is periodic or not
3 clc ;
```

```
4 n=-15:0.01:15;
5 y =cos((n/8)-(%pi));
6 xlabel('n')
7 ylabel('x(n)')
8 plot(n,y);
9 disp ( 'Plot  shows that given signal is NOT
    periodic ' ) ;
```

Scilab code Exa 6.4.c To check the signals are periodic or not

```
1 //example 6.4.C
2 //check the signal is periodic or not
3 clc ;
4 n=-15:0.01:15;
5 y =(1+cos(2*(%pi)*n/8)/2);
6 xlabel('n')
7 ylabel('x(n)')
8 plot(n,y);
9 disp ( 'Plot  shows that given signal is periodic of
    fundamental period=4 samples' ) ;
```

Scilab code Exa 6.4.d To check the signals are periodic or not

```
1 //example 6.4.d
2 //check the signal is periodic or not
3 clc ;
4 n=-15:0.01:15;
5 y =(cos(7*%pi*n)+%i*sin(7*%pi*n));
6 xlabel('n')
7 ylabel('x(n)')
8 plot(n,y);
9 disp ( 'Plot  shows that given signal is periodic of
    fundamental period=2 samples' ) ;
```

Scilab code Exa 6.5.a To determine the even and odd part of the signal

```
1  //ex_6.5.a even and odd signals of x(n)
2  clear;
3  clc;
4  close;
5  a=2;
6  n= 0:0.01:5;
7  x=a^(n);
8  figure
9  a=gca();
10 xtitle('x(n)')
11 plot2d(n,x)
12 figure
13 a=gca();
14 xtitle('even signal')
15 plot2d(n,x/2)
16 t1=-5:1/100:0;
17 plot2d(t1,x($:-1:1)/2)
18 a.y_location='origin'
19 figure
20 a=gca();
21 xtitle('odd signal')
22 plot2d(n,x/2)
23 t1=-5:1/100:0;
24 plot2d(t1,-x($:-1:1)/2)
25 a.y_location='origin'
26 a.x_location='origin'
```

Scilab code Exa 6.5.b To determine the even and odd part of the signal

```
1  //ex_6.5.b even and odd signals of x(n)
2  clear;
3  clc;
4  close;
5  n= 0:0.01:5;
6  x=2*exp(%i*((%pi)/3)*n);
7  figure
8  a=gca();
9  xtitle('x(n)')
10 plot2d(n,x)
11 figure
12 a=gca();
13 xtitle('even signal')
14 plot2d(n,x/2)
15 t1=-5:1/100:0;
16 plot2d(t1,x($:-1:1)/2)
17 a.y_location='origin'
18 figure
19 a=gca();
20 xtitle('odd signal')
21 plot2d(n,x/2)
22 t1=-5:1/100:0;
23 plot2d(t1,-x($:-1:1)/2)
24 a.y_location='origin'
25 a.x_location='origin'
```

Scilab code Exa 6.6.a To determine the power and energy of the given signal

```
1  //ex 6.6.a
2  clc;
3  E=(1/(1-(0.25)^(2)))
4  disp(E);
5  disp('AS THE ENERGY OF THE SIGNAL IS FINITE HENCE
     THE FOLLOWING SIGNALIS ENERGY SIGNAL');
```

Scilab code Exa 6.6.c To determine the power and energy of the given signal

```
1 //ex 6.6.c
2 clc;
3 N=100//ASSUMING THE N=100
4 p=(N)/(2*N)//AS LIMIT N TENDS TO INFINITY HENCE THE
    EQUATION
5 disp(p);
6 disp('AS THE POWER OF THE SIGNAL IS FINITE HENCE THE
    FOLLOWING SIGNALIS POWER SIGNAL');
```

Scilab code Exa 6.10.b to determine the system is whether time invariant or not

```
1 //Example 6 . 6 10 b
2 clc ;
3 n0 =1;
4 N =10;
5 for n =1: N
6 x ( n ) =n;
7 y ( n ) =n*x(n);
8 end
9 inputshift = (n)*x(N-n0) ;
10 outputshift = y (N - n0 ) ;
11 if( inputshift == outputshift )
12 disp ( 'THE GIVEN SYSTEM I S TIME INVARIANT ' )
13 else
14 disp ( 'THE GIVEN SYSTEM I S TIME VARIANT ' ) ;
15 end
```

Scilab code Exa 6.12.a to check whether the system is linear or non linear

```
1  //example 6.12.a
2  clear;
3  clc;
4  x1 = [1,1,1,1];
5  x2 = [2,2,2,2];
6  a = 1;
7  b = 1;
8  for n = 1:length(x1)
9    x3(n) = a*x1(n)+b*x2(n);
10 end
11 for n = 1:length(x1)
12   y1(n) = n*x1(n);
13   y2(n) = n*x2(n);
14   y3(n) = n*x3(n);
15 end
16 for n = 1:length(y1)
17   z(n) = a*y1(n)+b*y2(n);
18 end
19 count = 0;
20 for n =1:length(y1)
21   if(y3(n)== z(n))
22     count = count+1;
23   end
24 end
25 if(count == length(y3))
26    disp('Since It satisifies the superposition
        principle')
27    disp('The given system is a Linear system')
28    y3
29    z
30   else
31     disp('Since It does not satisify the
          superposition principle')
32     disp('The given system is a Non-Linear system')
33 end
```

Scilab code Exa 6.12.c to check whether the system is linear or non linear

```
1  //EXAMPLE 6.12.C
2  clear;
3  clc;
4  x1 = [1,1,1,1];
5  x2 = [2,2,2,2];
6  a = 1;
7  b = 1;
8  for n = 1:length(x1)
9    x3(n) = a*x1(n)+b*x2(n);
10 end
11 for n = 1:length(x1)
12   y1(n) = (x1(n)^2);
13   y2(n) = (x2(n)^2);
14   y3(n) = (x3(n)^2);
15 end
16 for n = 1:length(y1)
17   z(n) = a*y1(n)+b*y2(n);
18 end
19 count = 0;
20 for n =1:length(y1)
21   if(y3(n)== z(n))
22     count = count+1;
23   end
24 end
25 if(count == length(y3))
26    disp('Since It satisifies the superposition
        principle')
27    disp('The given system is a Linear system')
28    y3
29    z
30   else
31     disp('Since It does not satisify the
```

```
        superposition principle')
32      disp('The given system is a Non-Linear system')
33  end
```

Scilab code Exa 6.12.d to check whether the system is linear or non linear

```
1  //EXAMPLE 6.12.D
2  clear;
3  clc;
4  x1 = [1,1,1,1];
5  x2 = [2,2,2,2];
6  a = 1;
7  b = 1;
8  A=2
9  B=3;
10 for n = 1:length(x1)
11   x3(n) = a*x1(n)+b*x2(n);
12 end
13 for n = 1:length(x1)
14   y1(n) = A*x1(n)+B;
15   y2(n) = A*x2(n)+B;
16   y3(n) = A*x3(n)+B;
17 end
18 for n = 1:length(y1)
19   z(n) = a*y1(n)+b*y2(n);
20 end
21 count = 0;
22 for n =1:length(y1)
23   if(y3(n)== z(n))
24     count = count+1;
25   end
26 end
27 if(count == length(y3))
28    disp('Since It satisifies the superposition
        principle')
```

```
29     disp('The given system is a Linear system')
30     y3
31     z
32    else
33      disp('Since It does not satisify the
           superposition principle')
34      disp('The given system is a Non-Linear system')
35  end
```

Scilab code Exa 6.12.e to check whether the system is linear or non linear

```
1  //EXAMPLE 6.12.e
2  clear;
3  clc;
4  x1 = [1,1,1,1];
5  x2 = [2,2,2,2];
6  a = 1;
7  b = 1;
8  for n = 1:length(x1)
9    x3(n) = a*x1(n)+b*x2(n);
10 end
11 for n = 1:length(x1)
12   y1(n) = exp(x1(n));
13   y2(n) = exp(x2(n));
14   y3(n) = exp(x3(n));
15 end
16 for n = 1:length(y1)
17   z(n) = a*y1(n)+b*y2(n);
18 end
19 count = 0;
20 for n =1:length(y1)
21   if(y3(n)== z(n))
22     count = count+1;
23   end
24 end
```

```
25 if(count == length(y3))
26    disp('Since It satisifies the superposition
         principle')
27    disp('The given system is a Linear system')
28    y3
29    z
30   else
31     disp('Since It does not satisify the
          superposition principle')
32     disp('The given system is a Non-Linear system')
33 end
```

Scilab code Exa 6.13.a to check whether the system is linear or non linear

```
1  //EXAMPLE 6.13.a
2  clear;
3  clc;
4  x1 = [1,1,1,1];
5  x2 = [2,2,2,2];
6  a = 1;
7  b = 1;
8  C=3;
9  for n = 1:length(x1)
10   x3(n) = a*x1(n)+b*x2(n);
11 end
12 for n = 1:length(x1)
13   y1(n) = x1(n)+C;
14   y2(n) = x2(n)+C;
15   y3(n) = x3(n)+C;
16 end
17 for n = 1:length(y1)
18   z(n) = a*y1(n)+b*y2(n);
19 end
20 count = 0;
21 for n =1:length(y1)
```

```
22    if(y3(n)== z(n))
23      count = count+1;
24    end
25  end
26  if(count == length(y3))
27     disp('Since It satisifies the superposition
          principle')
28     disp('The given system is a Linear system')
29     y3
30     z
31    else
32      disp('Since It does not satisify the
           superposition principle')
33      disp('The given system is a Non-Linear system')
34  end
```

Scilab code Exa 6.13.b to check whether the system is linear or non linear

```
1  //EXAMPLE 6.13.b
2  clear;
3  clc;
4  x1 = [1,1,1,1];
5  x2 = [2,2,2,2];
6  a = 1;
7  b = 1;
8  for n = 1:length(x1)
9    x3(n) = a*x1(n)+b*x2(n);
10 end
11 for n = 1:length(x1)
12   y1(n) = a^(x1(n));
13   y2(n) = a^(x2(n));
14   y3(n) = a^(x3(n));
15 end
16 for n = 1:length(y1)
17   z(n) = a*y1(n)+b*y2(n);
```

```
18 end
19 count = 0;
20 for n =1:length(y1)
21   if(y3(n)== z(n))
22     count = count+1;
23   end
24 end
25 if(count == length(y3))
26    disp('Since It satisifies the superposition
        principle')
27    disp('The given system is a Linear system')
28    y3
29    z
30   else
31     disp('Since It does not satisify the
         superposition principle')
32     disp('The given system is a Non-Linear system')
33 end
```

Scilab code Exa 6.13.c to check whether the system is linear or non linear

```
1 //EXAMPLE 6.13.C
2 clear;
3 clc;
4 x1 = [1,1,1,1];
5 x2 = [2,2,2,2];
6 a = 1;
7 b = 1;
8 for n = 1:length(x1)
9   x3(n) = a*x1(n)+b*x2(n);
10 end
11 for n = 1:length(x1)
12   y1(n) = n*(x1(n))^(2);
13   y2(n) = n*(x2(n))^(2);
14   y3(n) = n*(x3(n))^(2);
```

```
15 end
16 for n = 1:length(y1)
17   z(n) = a*y1(n)+b*y2(n);
18 end
19 count = 0;
20 for n =1:length(y1)
21   if(y3(n)== z(n))
22     count = count+1;
23   end
24 end
25 if(count == length(y3))
26    disp('Since It satisifies the superposition
        principle')
27    disp('The given system is a Linear system')
28    y3
29    z
30   else
31     disp('Since It does not satisify the
         superposition principle')
32     disp('The given system is a Non-Linear system')
33 end
```

Chapter 7

Z TRANSFORM

Scilab code Exa 7.1.A Apply The Z transform

```
1 //Example 7.1.A
2 clc;
3 Y=[3,2,5,7];
4 X=0;
5 Syms z;
6 for i=1:4
7     X=X+Y(i)/z^(i-1)
8 end
9 disp(X,"X(z)=")
```

Scilab code Exa 7.1.B Apply The Z transform

```
1 //Example 7.1.B
2 clc;
3 Y=[6,4,5,3];
4 X=0;
5 Syms z;
6 for i=1:4
```

```
7       X=X+Y(i)/z^(i-4)
8   end
9   disp(X,"X(z)=")
```

Scilab code Exa 7.1.C Apply The Z transform

```
1   //Example 7.1.c
2   clc;
3   Y=[2,4,5,7,3];
4   X=0;
5   Syms z;
6   for i=1:5
7       X=X+Y(i)/z^(i-3)
8   end
9   disp(X,"X(z)=")
```

Scilab code Exa 7.2.A TO PERFORM Z TRANSFORM OPERATION

```
1   //Example 7.2.A
2   clc;
3   Syms z;
4   x=1;
5   X=nusum(x/(z^i),i,0,%inf);
6   disp(X,'X(z)=');
```

Scilab code Exa 7.2.B TO PERFORM Z TRANSFORM OPERATION

```
1   //Example 7.2.B
2   clc;
3   Syms z;
```

```
4 X=nusum((0.5/z)^(i),i,0,%inf);
5 disp(X,'X(z)=');
```

Scilab code Exa 7.2.c TO PERFORM Z TRANSFORM OPERATION

```
1 //Example 7.2.C
2 clc;
3 Syms z;
4 X=nusum((0.8/z)^(i),i,-(%inf),-1);
5 disp(X,'X(z)=');
```

Scilab code Exa 7.2.d TO PERFORM Z TRANSFORM OPERATION

```
1 //Example 7.2.D
2 clc;
3 Syms z;
4 X=nusum((0.5/z)^(i),i,0,%inf);
5 D=nusum((0.8/z)^(i),i,-(%inf),-1);
6 disp(D+X,'X(z)=');
```

Chapter 9

DIscrete Fourier Transform and Fast Fourier Transform

Scilab code Exa 9.1 Compute DFT

```
1  //Example 9.1
2  clc;
3  i=0;
4  k0=0;
5  k1=1;
6  k2=2;
7  k3=3;
8  k4=4;
9  k5=5;
10 k6=6;
11 k7=7;
12 X0=0;
13 X1=0;
14 X2=0;
15 X3=0;
16 for i=0:2
17 X0=X0+((1*(exp(-%i*2*%pi*k0*i/4))/3))
18 X1=X1+((1*(exp(-%i*2*%pi*k1*i/4))/3))
19 X2=X2+((1*(exp(-%i*2*%pi*k2*i/4))/3))
```

```
X3=X3+((1*(exp(-%i*2*%pi*k3*i/4))/3))
end
disp('For N=4 points:')
disp('X(K)=')
disp(X3,X2,X1,X0)
x1=[X0,X1,X2,X3]
x2=(real(x1).^2+imag(x1).^2).^(1/2)
disp('Magnitude:')
disp(x2)
x3=atan(imag(x1),real(x1))
disp('Phase:')
disp(x3)
X0=0;
X1=0;
X2=0;
X3=0;
X4=0;
X5=0;
X6=0;
X7=0;
disp('For N=8 points:')
disp('X(K)=')
for i=0:2
X0=X0+((1*(exp(-%i*2*%pi*k0*i/8))/3))
X1=X1+((1*(exp(-%i*2*%pi*k1*i/8))/3))
X2=X2+((1*(exp(-%i*2*%pi*k2*i/8))/3))
X3=X3+((1*(exp(-%i*2*%pi*k3*i/8))/3))
X4=X4+((1*(exp(-%i*2*%pi*k4*i/8))/3))
X5=X5+((1*(exp(-%i*2*%pi*k5*i/8))/3))
X6=X6+((1*(exp(-%i*2*%pi*k6*i/8))/3))
X7=X7+((1*(exp(-%i*2*%pi*k7*i/8))/3))
end
disp(X7,X6,X5,X4,X3,X2,X1,X0)
x1=[X0,X1,X2,X3,X4,X5,X6,X7]
x2=(real(x1).^2+imag(x1).^2).^(1/2)
disp('Magnitude:')
disp(x2)
x3=atan(imag(x1),real(x1))
```

```
58 disp('Phase:')
59 disp(x3)
```

Scilab code Exa 9.2 Compute DFT

```
1  //Example 9.2
2  clc;
3  x0=[0,1,2,3]
4  x1=fft(x0);
5  disp('X(K)=')
6  disp(x1);
7  x2=(real(x1).^2+imag(x1).^2).^(1/2)
8  disp('Magnitude:')
9  disp(x2)
10 x3=atan(imag(x1),real(x1))
11 disp('Phase:')
12 disp(x3)
```

Scilab code Exa 9.3 Compute Circular Convolution using DFT

```
1  //Example 9.3
2  clc;
3  x0=[2,1,2,1]
4  x1=fft(x0);
5  disp('X1(K)=')
6  disp(x1);
7  x2=[1,2,3,4]
8  x3=fft(x2)
9  disp('X2(K)=')
10 disp(x3);
11 x4=x1.*x3
12 disp('X1(K)*X2(K)=')
13 disp(x4)
```

```
14 x5=ifft(x4)
15 disp('x1(n)*x2(n)')
16 disp(x5)
```

Scilab code Exa 9.4 Compute Linear Convolution using DFT

```
1 //Example 9.4
2 clc;
3 x0=[1,0.5,0]
4 x1=fft(x0);
5 disp('X1(K)=')
6 disp(x1);
7 h=[0.5,1,0]
8 x2=fft(h)
9 disp('H(K)=')
10 disp(x2);
11 x3=x1.*x2
12 disp('X1(K)*H(K)=')
13 disp(x3)
14 x4=ifft(x3)
15 disp('x1(n)*x2(n)')
16 disp(x4)
```

Scilab code Exa 9.5 Compute DFT

```
1 //Example 9.5
2 clc;
3 x0=[2,2,2,2,1,1,1,1]
4 x1=fft(x0);
5 disp('X(K)=')
6 disp(x1);
7 x2=(real(x1).^2+imag(x1).^2).^(1/2)
8 disp('Magnitude:')
```

```
disp(x2)
x3=atan(imag(x1),real(x1))
disp('Phase:')
disp(x3)
```

Scilab code Exa 9.6 Compute Impulse Response using DFT

```
//Example 9.6
clc;
x0=[1,1,1,0]
x1=fft(x0);
disp('X1(K)=')
disp(x1);
h=[-1,-1,0,0]
x2=fft(h)
disp('H(K)=')
disp(x2);
x3=x1.*x2
disp('X1(K)*H(K)=')
disp(x3)
x4=ifft(x3)
disp('x1(n)*x2(n)')
disp(x4)
```

Scilab code Exa 9.7 Compute Impulse Response using DFT

```
//Example 9.7
clc;
x0=[-1,1,2,1,-1,0,0,0]
x1=fft(x0);
disp('X1(K)=')
disp(x1);
h=[-1,1,-1,1,0,0,0,0]
```

```
x2=fft(h)
disp('H(K)=')
disp(x2);
x3=x1.*x2
disp('X1(K)*H(K)=')
disp(x3)
x4=ifft(x3)
disp('x1(n)*x2(n)')
disp(x4)
```