Scilab Textbook Companion for
Principles Of Linear Systems And Signals
by B. P. Lathi[1]

Created by
A. Lasya Priya
B.Tech (pursuing)
Electrical Engineering
NIT, Surathkal
College Teacher
H. Girisha Navada, NIT Surathkal
Cross-Checked by
S.M. Giridharan, IIT Bombay

August 11, 2013

---

[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: Principles Of Linear Systems And Signals

Author: B. P. Lathi

Publisher: Oxford University Press

Edition: 2

Year: 2009

ISBN: 0-19-806227-3

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

List of Figures

Chapter 1

signals and systems

Scilab code Exa 1.2 power and rms value

```
//signals and systems
//power and rms value of a signal
clear all
close
clc
//part a is a periodic function with period 2*pi/w0

disp("consider the power for almost infinite range")
   ;
disp('part (a)')
disp("integrating ((c*cos(w0*t +theta))^2) for this
   big range gives c^2/2 as the power which is
   irrespective of w0");
disp("rms value is the square root of power and
   therefpre equal to sqrt(c^2/2)\n\n");
//part b is the sum of 2 sinusoids
disp('part (b)')
disp("again integrating in the same way and ignoring
    the zero terms we get (c1^2+c2^2)/2");
//part c deals with a complex signal
disp('part (c)')
```

```
disp(”integrating the expression we get |D|^2 as the
    power and |D| as the rms value”);
```

Scilab code Exa 1.3 time shifting

```
//signals and systems
//time shifting
clear all
close
clc
t=[-4:0.001:4];
a=gca();
plot(t,(exp(-2*t)).*(t>0))
a.thickness=2;
a.y_location=”middle”;
xtitle=('the signal x(t)')
//delaying the function by 1 second we obtain
figure
a=gca();
plot(t,(exp(-2*(t-1))).*((t>1)))
a.thickness=2;
a.y_location=”middle”;
title=('the signal x(t−1)')
//advancing the function by 1 second we obtain
figure
a=gca();
plot(t,(exp(-2*(t+1))).*(t>-1))
a.thickness=2;
a.y_location=”middle”;
xtitle=('the signal x(t+1)')
```

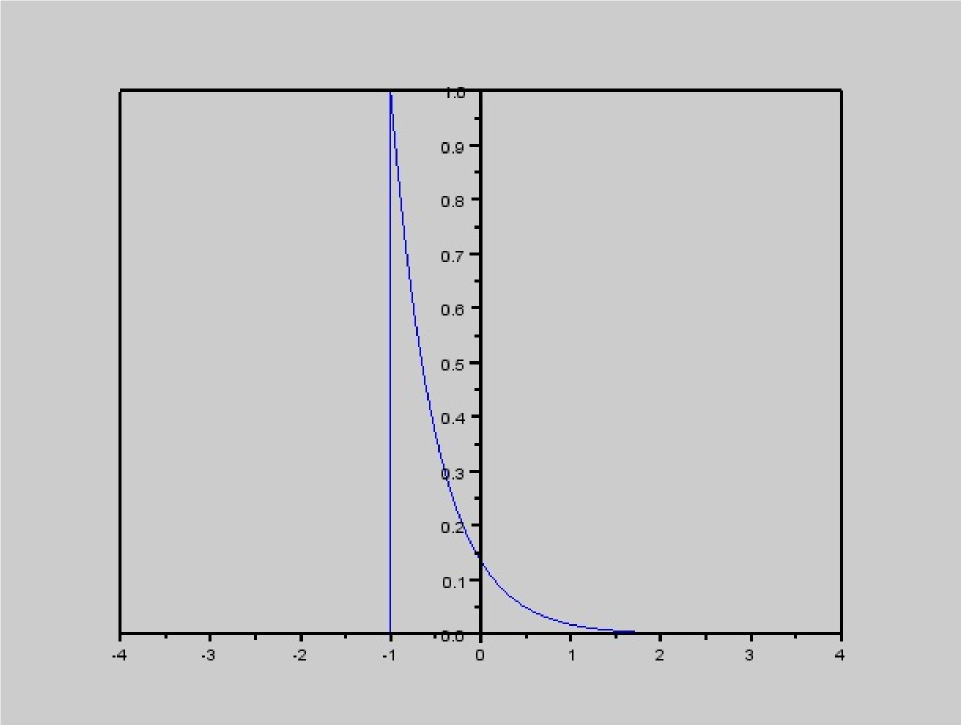

Figure 1.1: time shifting

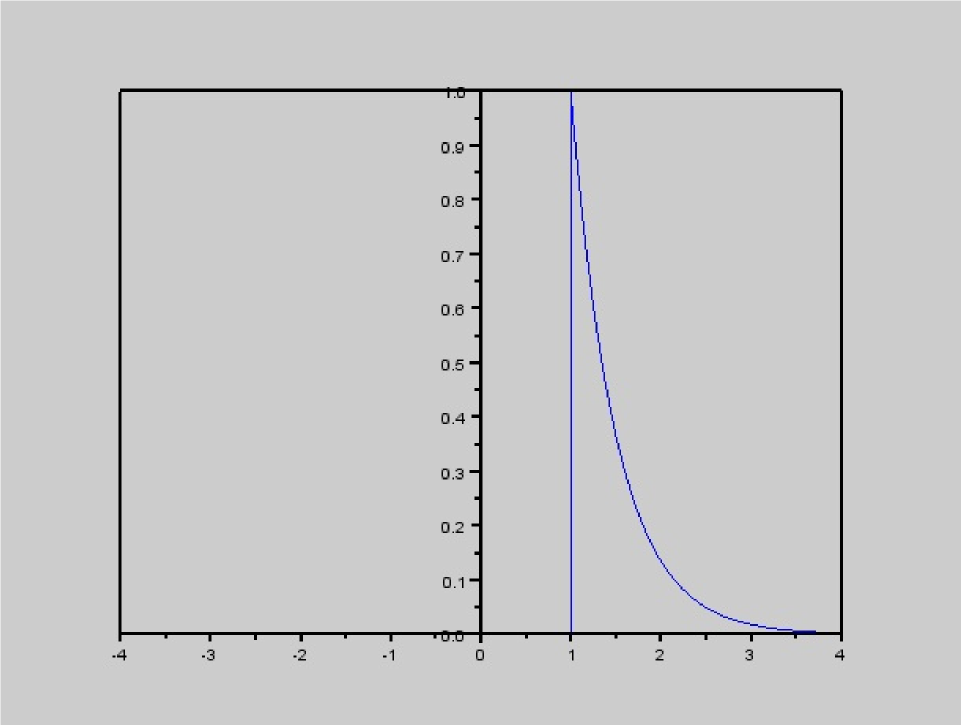

Figure 1.2: time shifting

Scilab code Exa 1.4 time scaling

```
//signals and systems
//time scaling
clear all
close
clc
t=[-4:0.1:6];
a=gca();
plot(t,2.*((t>-1.5)&(t<=0))+2*exp(-t/2).*((t>0)&(t
   <=3)));
figure
a.thickness=2;
a.y_location="middle";
xtitle=('the signal x(t)');
//compressing this graph by a factor 3
a=gca();
plot(t,2.*((t>-0.5)&(t<=0))+2*exp(-3*t/2).*((t>0)&(t
   <=1)));
figure
a.thickness=2;
a.y_location="middle";
xtitle=('the signal x(3t)');
//expanding this signal by a factor 2
a=gca();
plot(t,2.*((t>-3)&(t<=0))+2*exp(-t/4).*((t>0)&(t<=6)
   ));
a.thickness=2;
a.y_location="middle";
xtitle=('the signal x(t/2)');
//the coordinates can b easily obtained from the
   graphs
```

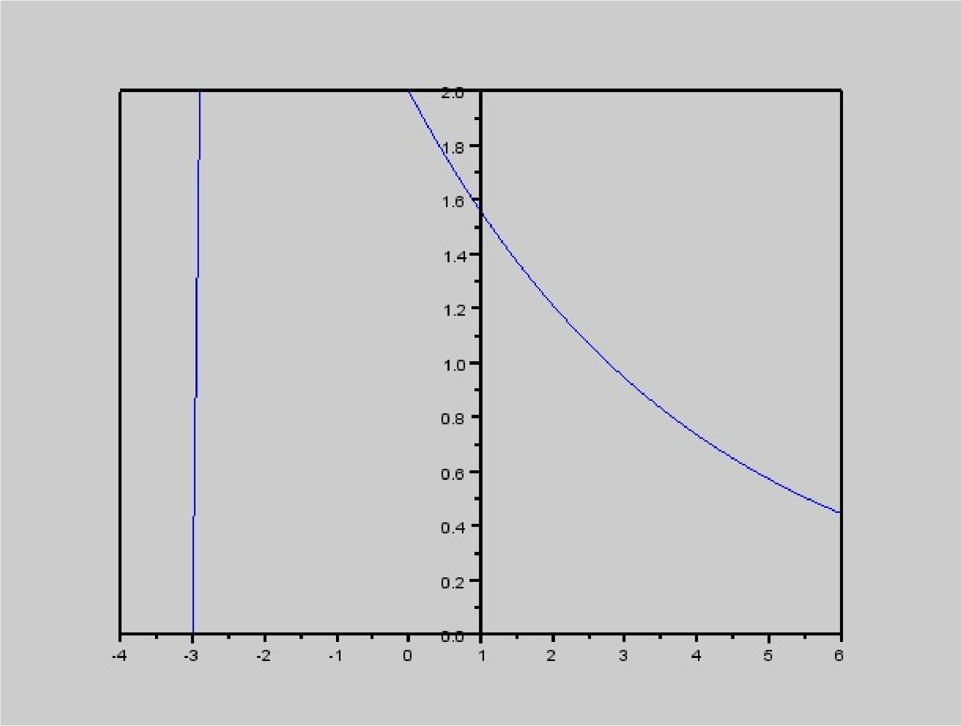

Figure 1.3: time scaling

Scilab code Exa 1.5 time reversal

```
//signals and systems
//time reversal
clear all
close
clc
t=[-6:0.1:6];
```

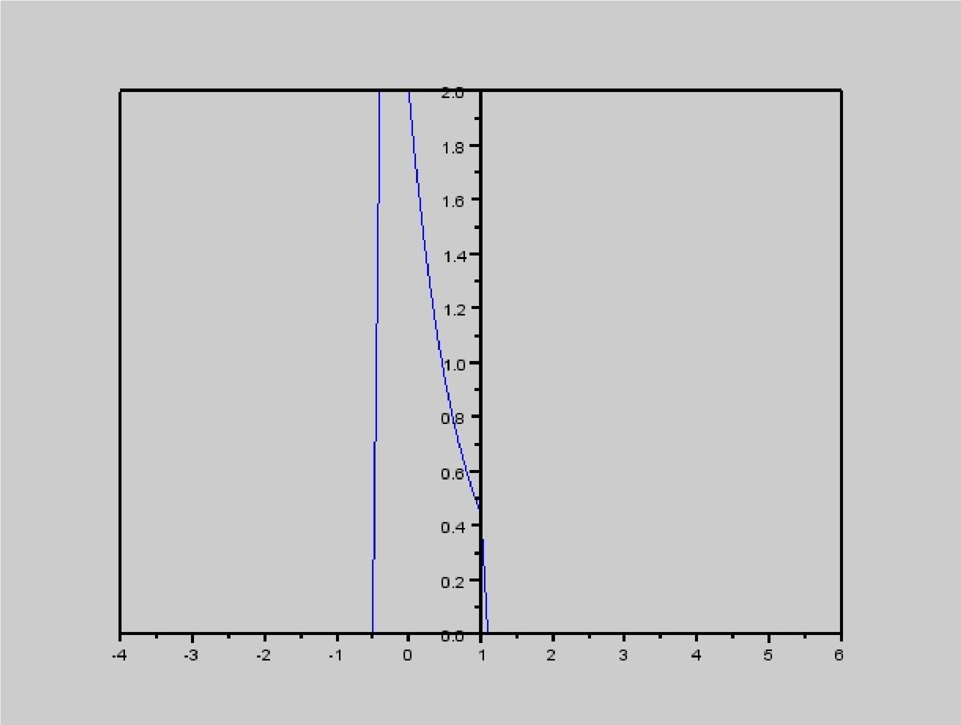

Figure 1.4: time scaling

```
a=gca();
plot(t,exp(t/2).*((t>=-5)&(t<=-1)));
figure
a.thickness=2;
a.y_location="middle";
xtitle=('the signal x(t)')
//by replacing t by -t we get
a=gca();
plot(t,exp(-t/2).*((t>=1)&(t<5)));
a.thickness=2;
a.y_location="middle";
xtitle=('the signal x(-t)')
//the coordinates can be easily observed from the graphs
```

Scilab code Exa 1.6 basic signal models

```
//signals and systems
//representation of a signal
clear all
close
clc
t=[0:0.1:5];
a=gca();
plot(t,t.*((t>=0)&(t<=2)) - 2*(t-3).*((t>2)&(t<=3)));
a.thickness=2;
a.y_location="middle";
xtitle=('the signal x(t)')
```

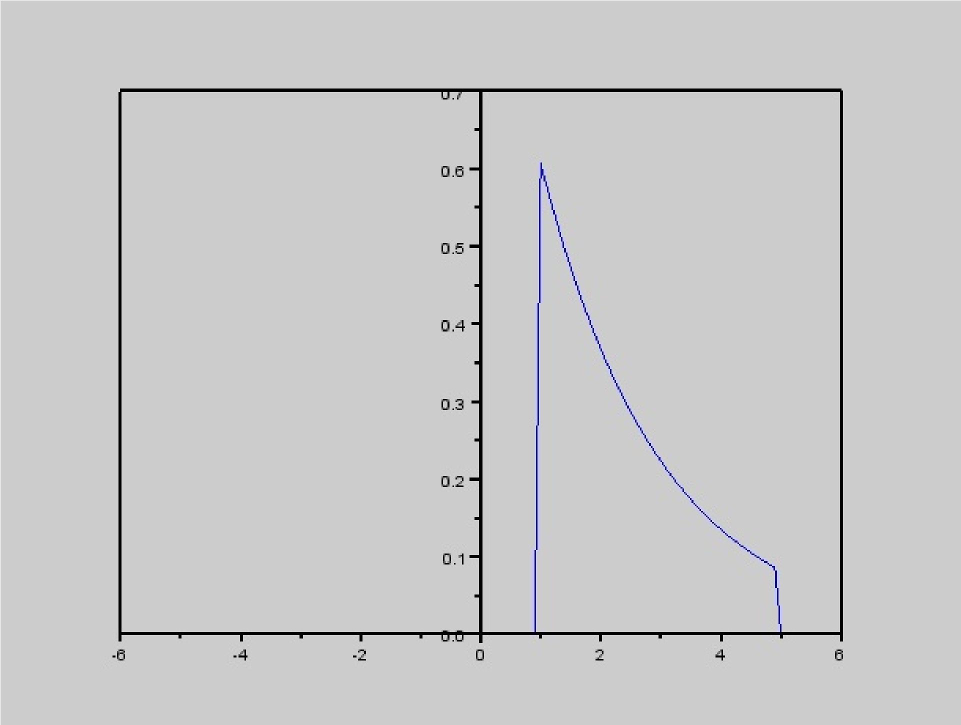

Figure 1.5: time reversal

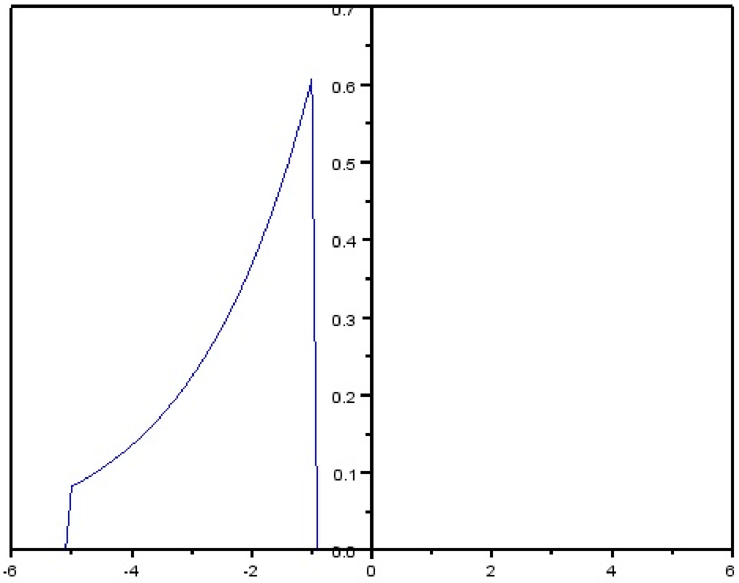

Figure 1.6: time reversal

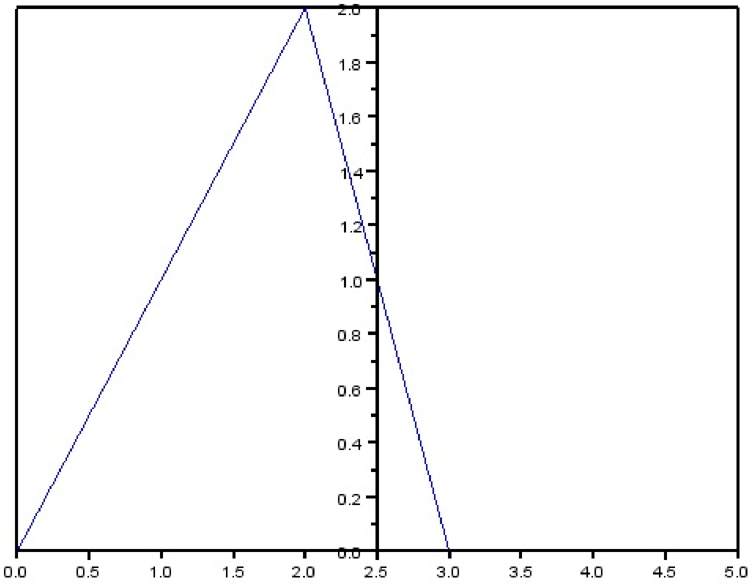

Figure 1.7: basic signal models

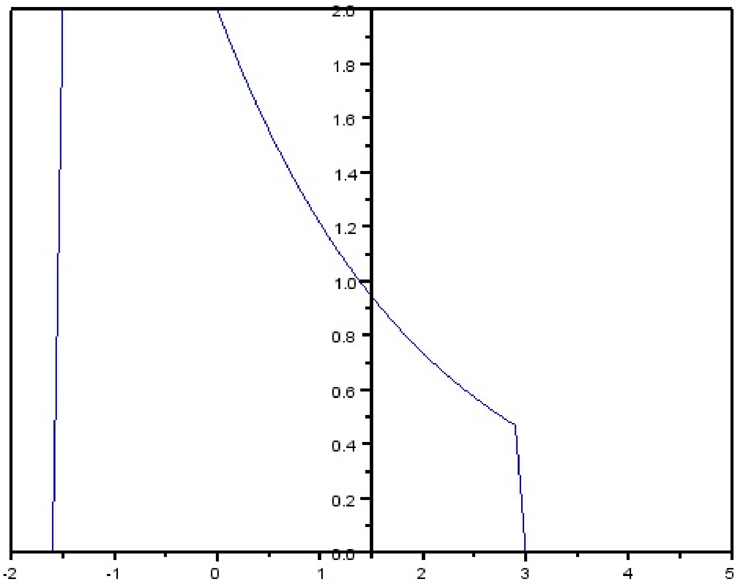

Figure 1.8: describing a signal in a single expression

```
//this can be written as a combination of 2 lines
disp("x(t)=x1(t)+x2(t)= tu(t)-3(t-2)u(t-2)+2(t-3)u(t-3)");
```

Scilab code Exa 1.7 describing a signal in a single expression

```
//signals and systems
//representation of a signal
clear all
close
clc
```

```
t=[-2:0.1:5];
a=gca();
plot(t,2.*((t>=-1.5)&(t<0))+2*exp(-t/2).*((t>=0)&(t<3)));
a.thickness=2;
a.y_location="middle";
xtitle=('the signal x(t-1)')
//this is a cobination of a constant function and an exponential function
disp("x(t)=x1(t)+x2(t)= 2u(t+1.5)-2(1-exp(-t/2))u(t)-2exp(-t/2)u(t-3)");
```

Scilab code Exa 1.8 even and odd components of a signal

```
//signals and systems
//odd and even components
clear all
close
clc
t = 0:1/100:5;
x=exp(%i.*t);
y=exp(-%i.*t);
even=x./2+y./2;
odd=x./2-y./2;
figure
a=gca();
plot2d(t,even)
a.x_location='origin'
xtitle=('even')
figure
a=gca();
plot2d(t,odd./%i)
a.x_location='origin'
xtitle=('odd')
```

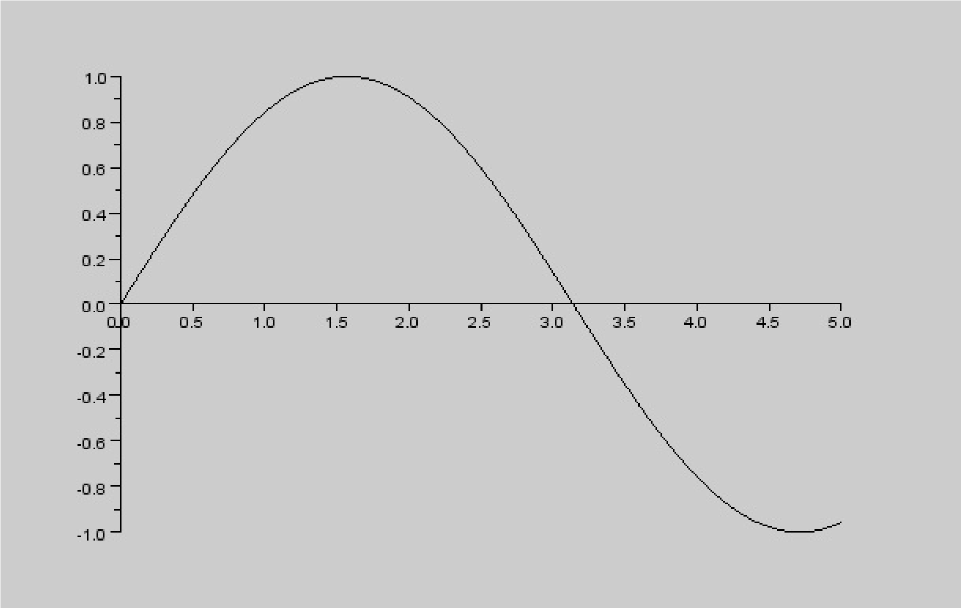

Figure 1.9: even and odd components of a signal

Scilab code Exa 1.10 input output equation

```
//signals and systems
//formation of differential equation for a series RC
    circuit
clear all
close
clc
r=15;
c=0.2;
//let the input voltage be x(t)
```

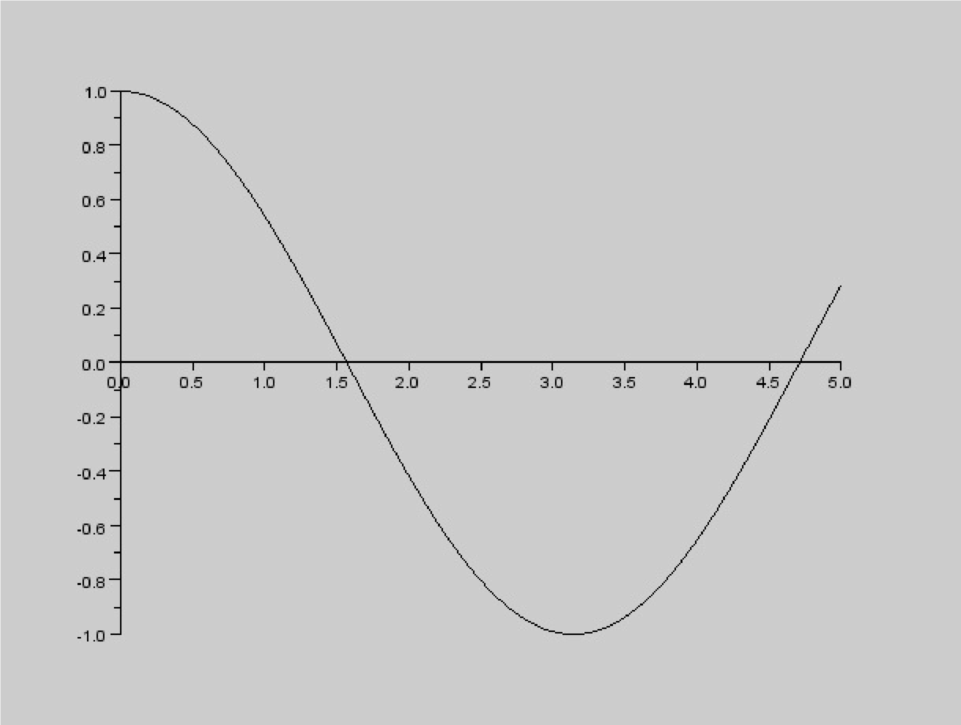

Figure 1.10: even and odd components of a signal

```
//let the loop current be i(t)
//let capacitor voltage be y(t)
disp("the loop equation 4 the circuit is given by r*i(t)+(5/D)*i(t)=x(t)")
disp("final form - (3D+1)y(t)=x(t)")
//the next few problems are of the same type where we have to frame the eqation based on the scenario
```

Chapter 2

time domain analysis of continuous time systems

Scilab code Exa 2.5 unit impulse response for an LTIC system

```
//time domain analysis of continuous time systems
//Convolution Integral of input x(t) = (e^-t).u(t) and g(t) =(e^-2*t)u(t)
clear all;
close;
clc;
Max_Limit = 10;
t = 0:0.001:10;
for i=1:length(t)
     g(i) =(exp(-2*t(i)));
end
  x= exp(-(t));

y = convol(x,g)
figure
```

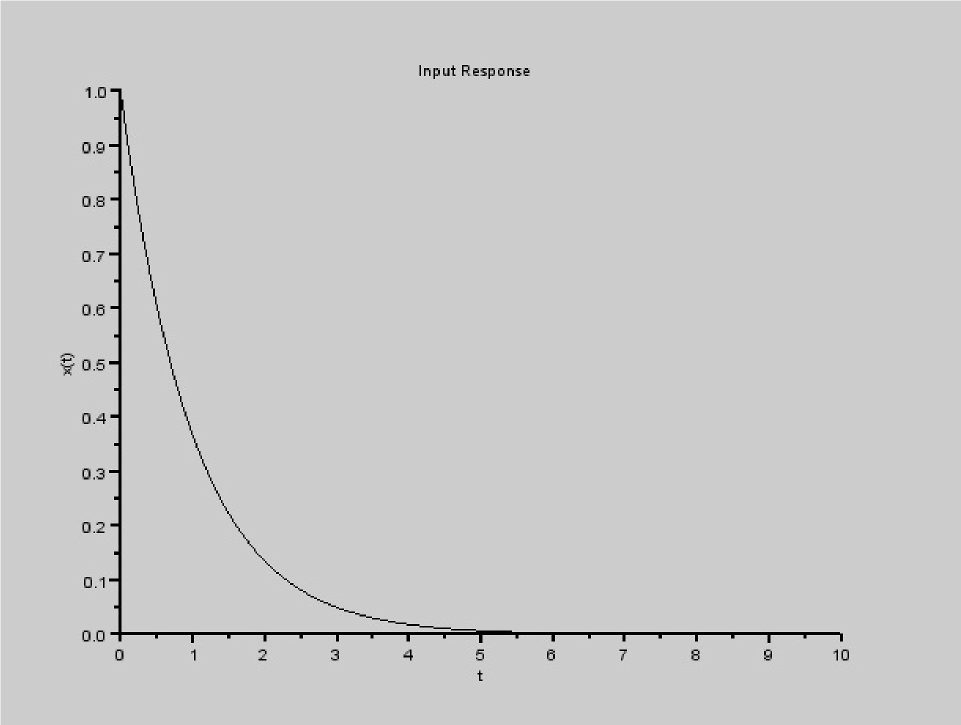

Figure 2.1: unit impulse response for an LTIC system

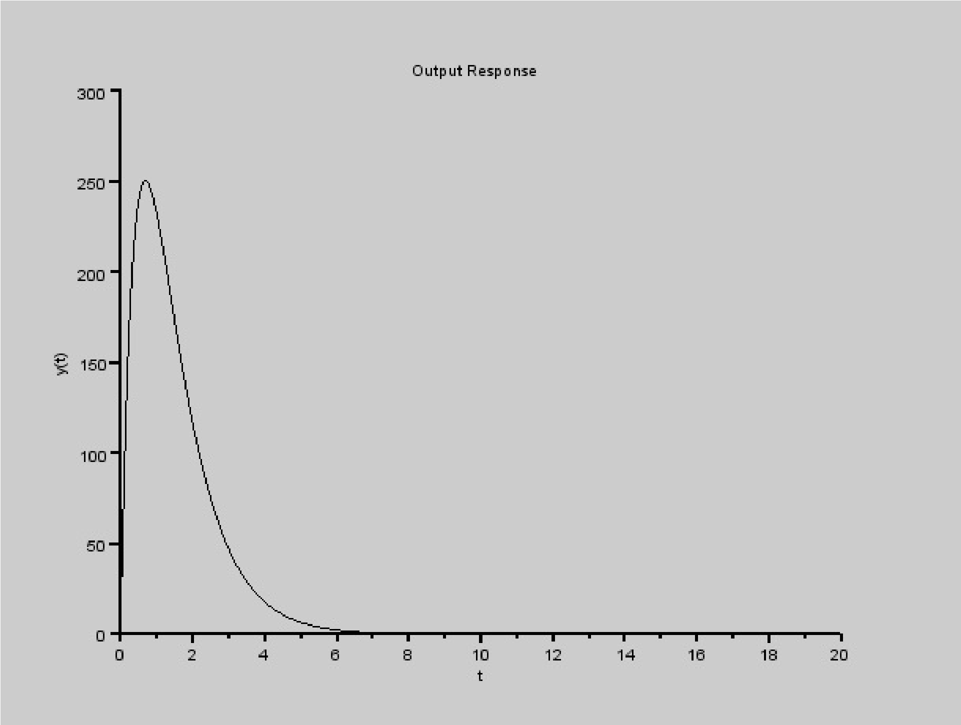

Figure 2.2: unit impulse response for an LTIC system

```
a=gca();
plot2d(t,g)
xtitle('Impulse Response','t','h(t)');
a.thickness = 2;
figure
a=gca();
plot2d(t,x)
xtitle('Input Response','t','x(t)');
a.thickness = 2;
figure
a=gca();
T=0:0.001:20;
plot2d(T,y)
xtitle('Output Response','t','y(t)');
a.thickness = 2;
```

Scilab code Exa 2.6 zero state response

```
//time domain analysis of continuous time systems
//Convolution Integral of input x(t) = (e^-3t).u(t) and h(t) =(2*e^-2*t-e^-t)u(t)
clear;
close;
clc;
Max_Limit = 10;
t = 0:0.001:10;
for i=1:length(t)
     g(i) =(2*exp(-2*t(i))-exp(-t(i)));
end
  x= exp(-3*(t));

```

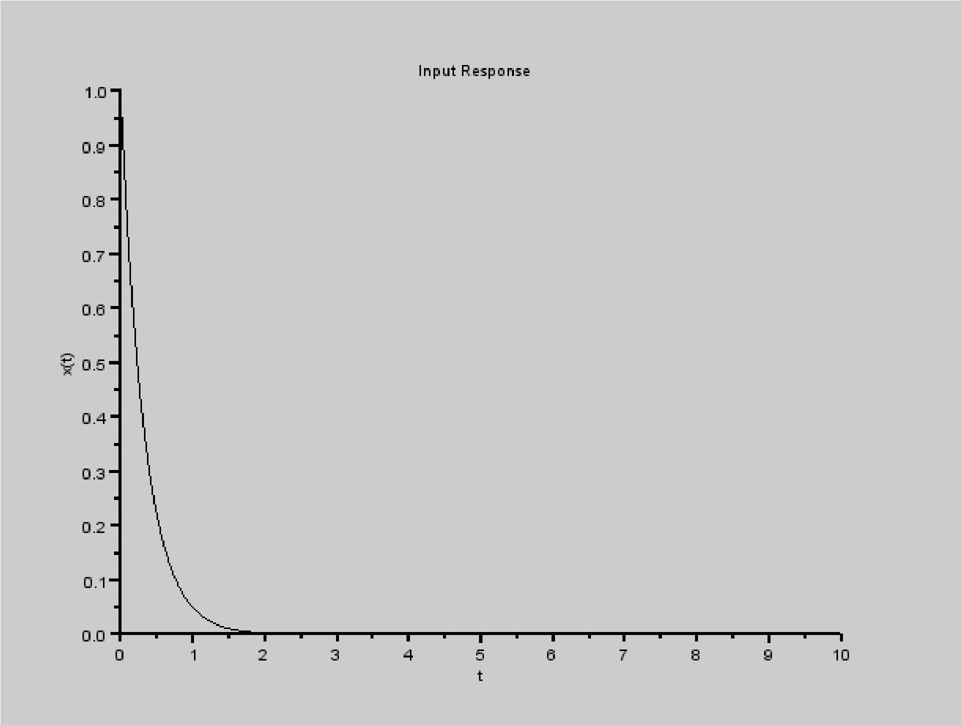

Figure 2.3: zero state response

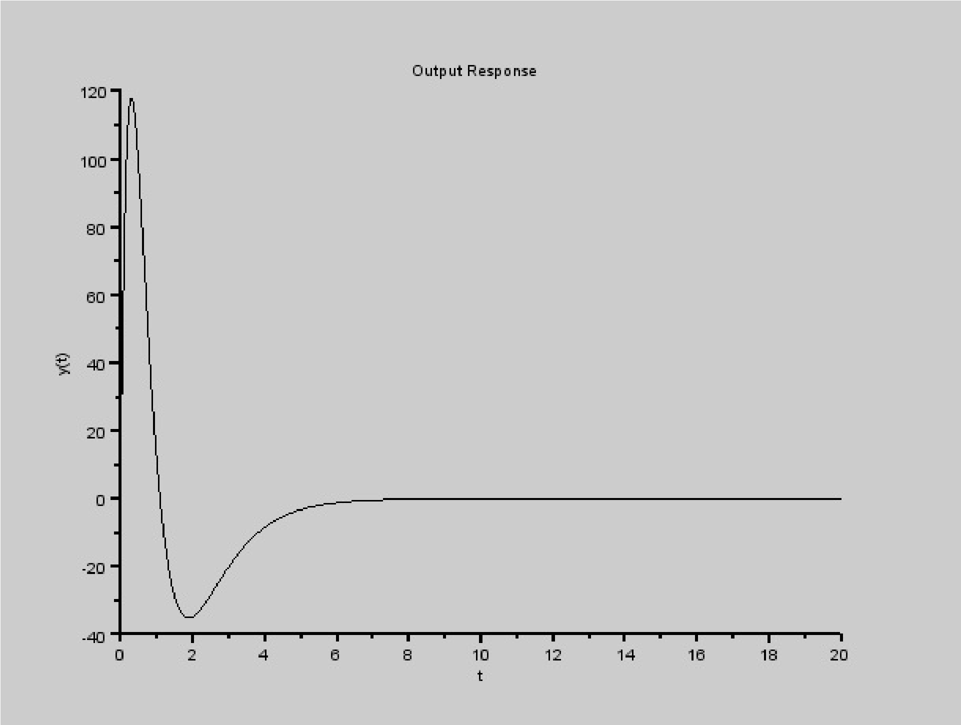

Figure 2.4: zero state response

```
y = convol(x,g)
figure
a=gca();
plot2d(t,g)
xtitle('Impulse Response','t','h(t)');
a.thickness = 2;
figure
a=gca();
plot2d(t,x)
xtitle('Input Response','t','x(t)');
a.thickness = 2;
figure
a=gca();
T=0:0.001:20;
plot2d(T,y)
xtitle('Output Response','t','y(t)');
a.thickness = 2;
```

Scilab code Exa 2.7 graphical convolution

```
//time domain analysis of continuous time systems
//Convolution Integral of input x(t) = (e^-t).u(t) and g(t) =u(t)
clear all;
close;
clc;
Max_Limit = 10;
t = -10:0.001:10;
for i=1:length(t)

       g(i)=exp(-t(i));
       x(i)=exp(-2*t(i));
end

y = convol(x,g)
```

```
figure
a=gca();
plot2d(t,g)
xtitle('Impulse Response','t','h(t)');
a.thickness = 2;
figure
a=gca();
plot2d(t,x)
xtitle('Input Response','t','x(t)');
a.thickness = 2;
figure
a=gca();
T=-20:0.001:20;
plot2d(T,y)
xtitle('Output Response','t','y(t)');
a.thickness = 2;
```

Scilab code Exa 2.8 graphical convolution

```
//time domain analysis of continuous time systems
//Convolution Integral of input x(t) = (e^-t).u(t)
   and g(t) =u(t)
clear all;
close;
clc;
Max_Limit = 10;
t = -10:0.001:10;
```

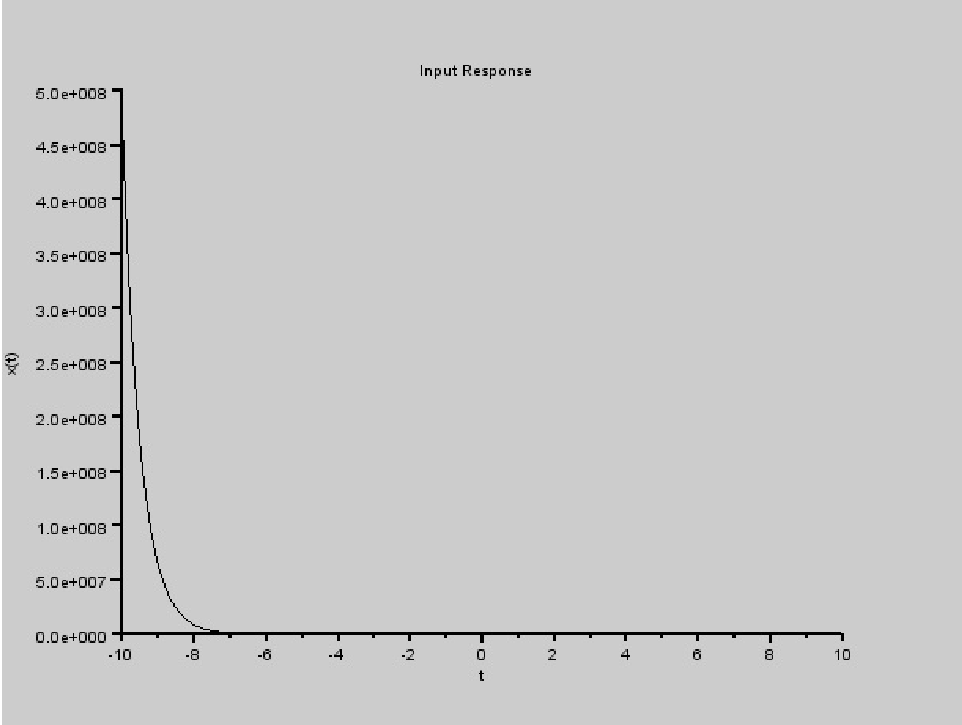

Figure 2.5: graphical convolution

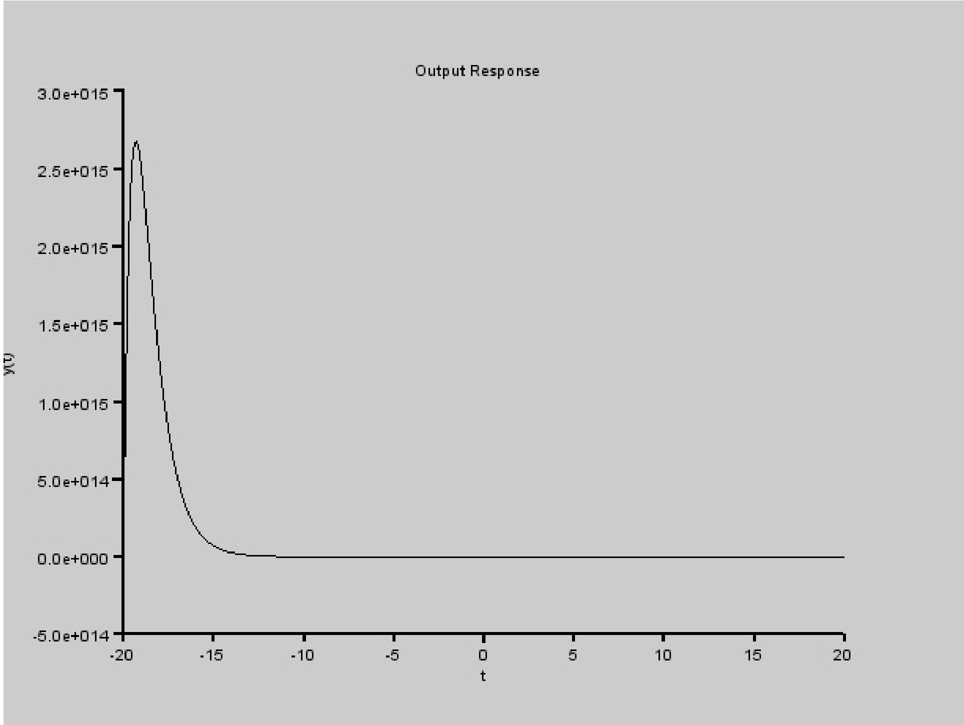

Figure 2.6: graphical convolution

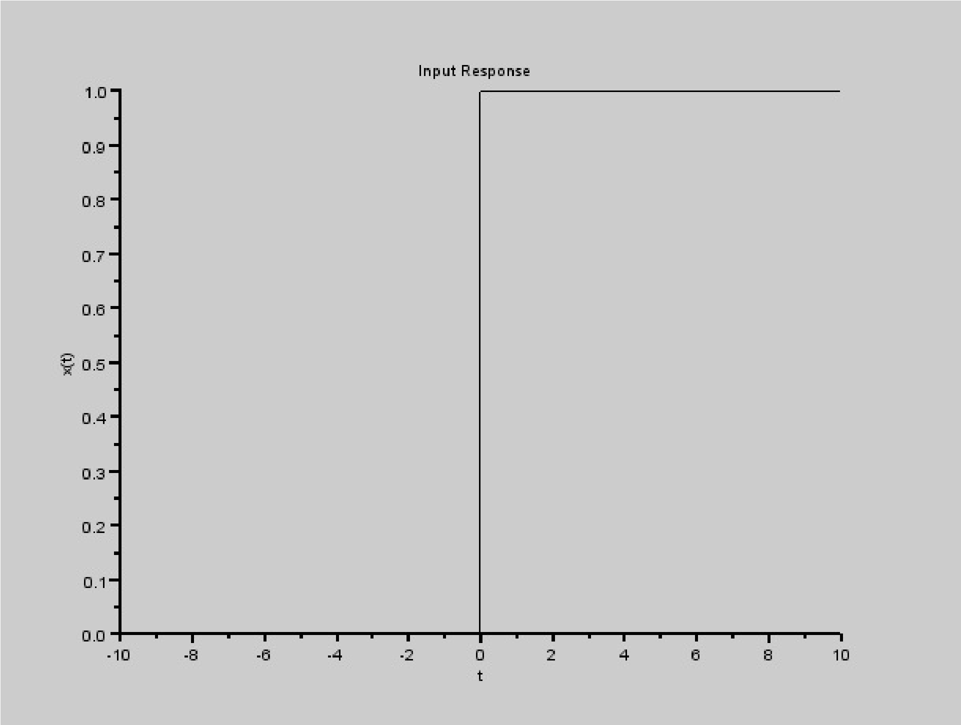

Figure 2.7: graphical convolution

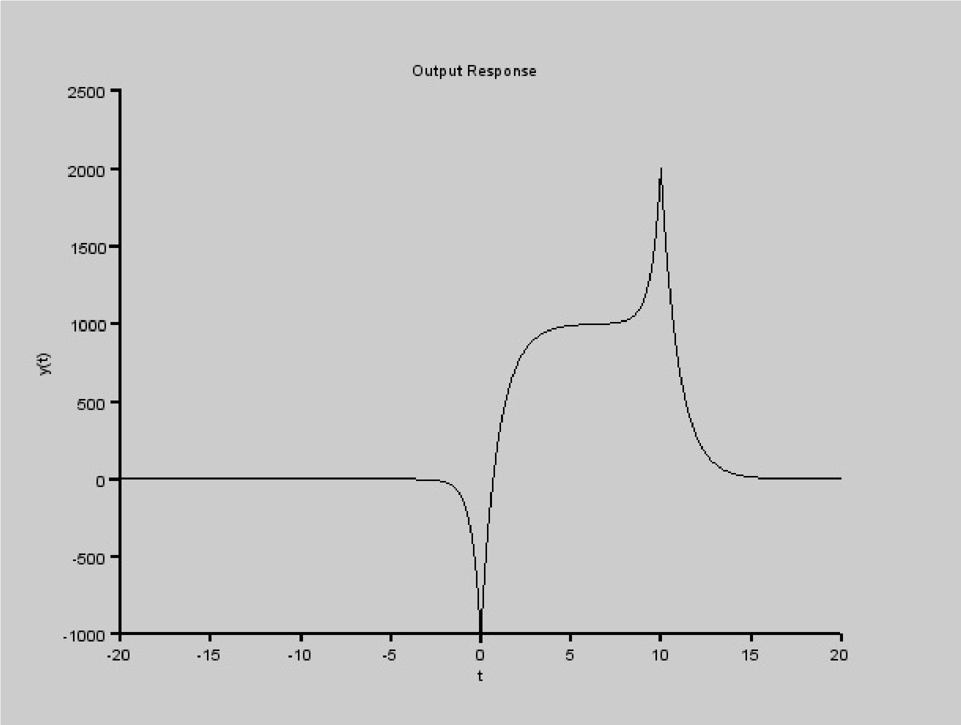

Figure 2.8: graphical convolution

```
for i=1:length(t)
    if t(i)<0 then
       g(i)=-2*exp(2*t(i));
       x(i)=0;
    else
        g(i)=2*exp(-t(i));
        x(i)=1;
    end
end

y = convol(x,g)
figure
a=gca();
plot2d(t,g)
xtitle('Impulse Response','t','h(t)');
a.thickness = 2;
figure
a=gca();
plot2d(t,x)
xtitle('Input Response','t','x(t)');
a.thickness = 2;
figure
a=gca();
T=-20:0.001:20;
plot2d(T,y)
xtitle('Output Response','t','y(t)');
a.thickness = 2;
```

Scilab code Exa 2.9 graphical convolution

```
//time domain analysis of continuous time systems
```

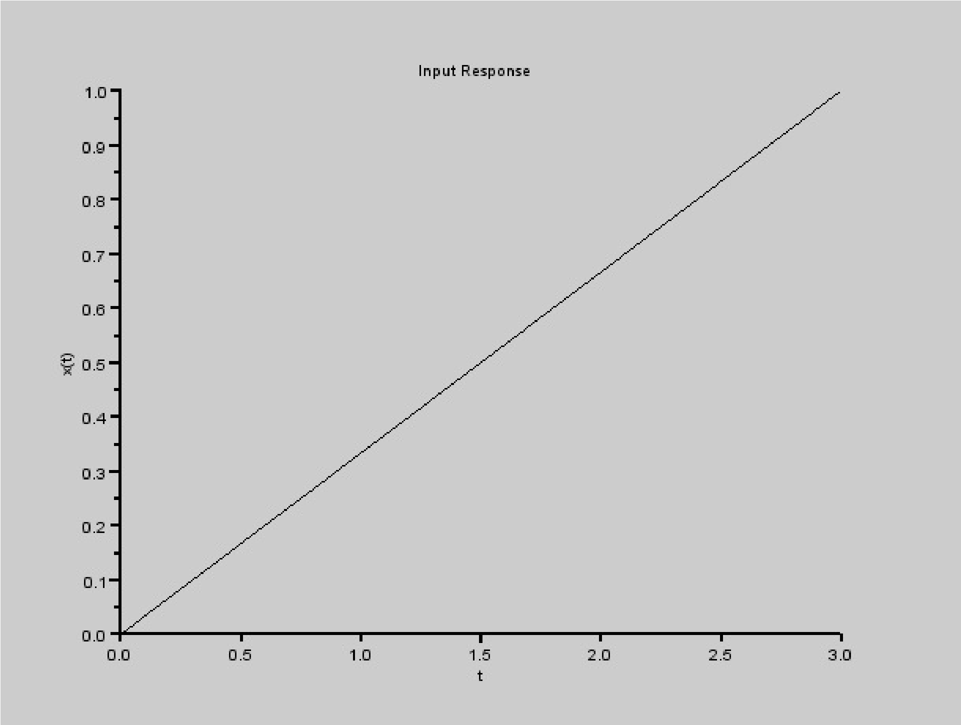

Figure 2.9: graphical convolution

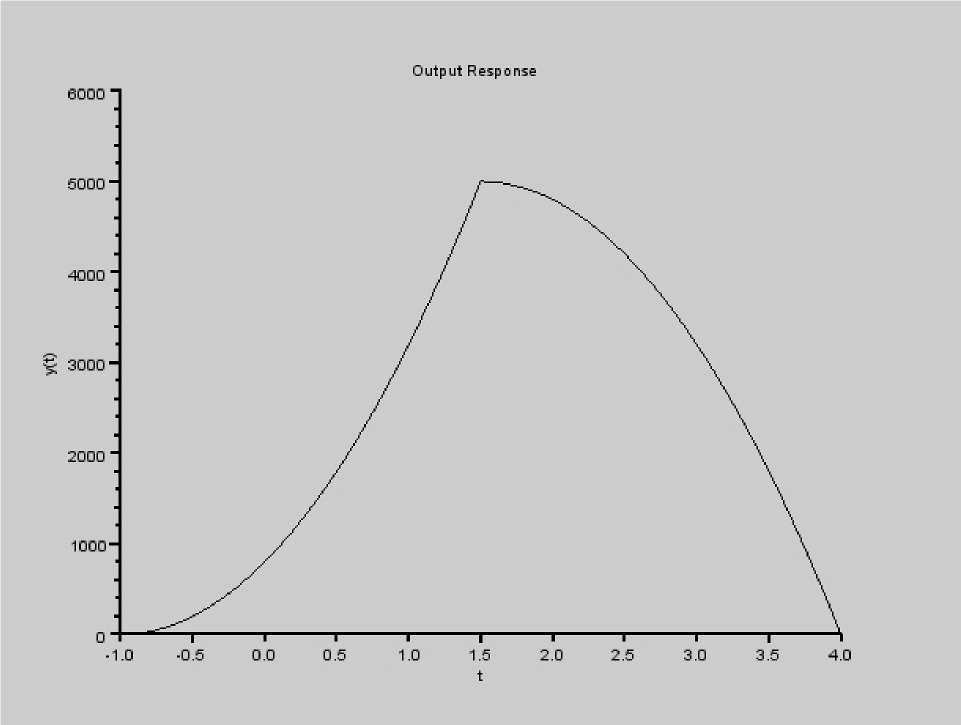

Figure 2.10: graphical convolution

```
//Convolution Integral of input x(t) = (e^-t).u(t) and g(t) =u(t)
clear all;
close;
clc;
Max_Limit = 10;
t =linspace(-1,1,10001);
for i=1:length(t)
    g(i)=1;
end
t1=linspace(0,3,10001);
for i=1:length(t1)
x(i)= t1(i)/3;
end
y = convol(x,g);
figure
a=gca();
size(t)
size(g)
plot2d(t,g)
xtitle('Impulse Response','t','h(t)');
a.thickness = 2;
figure
a=gca();
size(x)
plot2d(t1,x)
xtitle('Input Response','t','x(t)');
a.thickness = 2;
figure
a=gca();
T=linspace(-1,4,20001);
size(y)
plot2d(T,y)
xtitle('Output Response','t','y(t)');
a.thickness = 2;
```

Chapter 3

time domain analysis of discrete time systems

Scilab code Exa 3.1 energy and power of a signal

```
//signals and systems
//time domain analysis of discreet time systems
//energy of a signal
clear all;
close;
clc;
n=0:1:5
figure
a=gca();
plot2d(n,n);
energy=sum(n^2)
power=(1/6)*sum(n^2)
disp(energy)
disp(power)
```

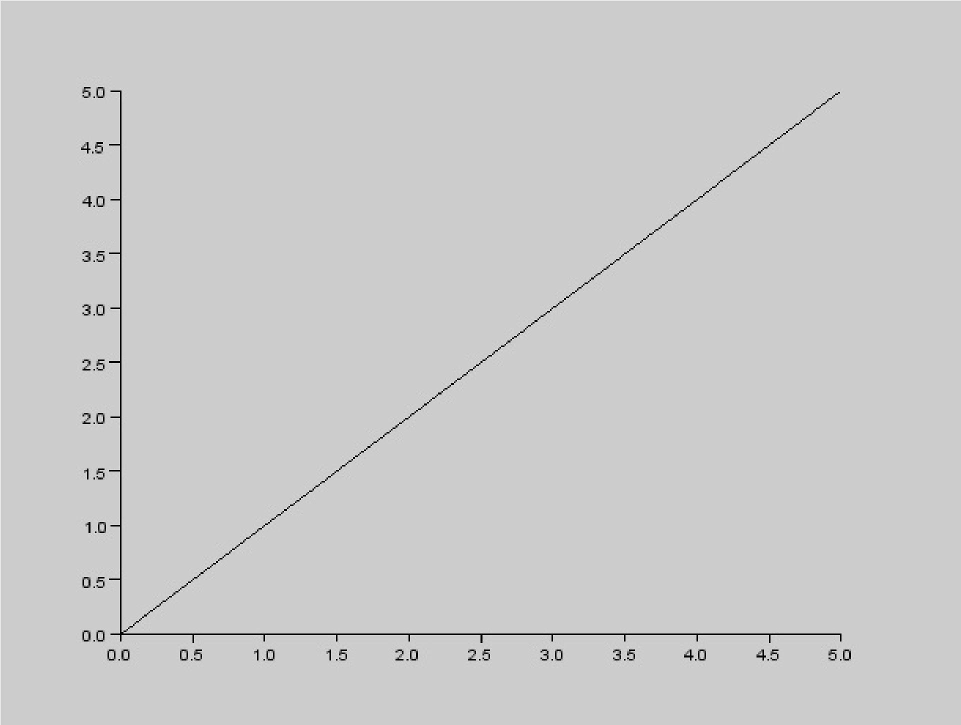

Figure 3.1: energy and power of a signal

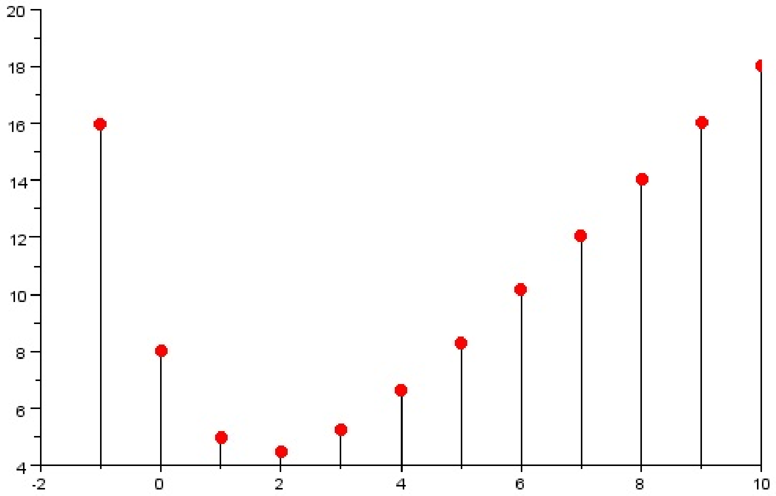

Figure 3.2: iterative solution

Scilab code Exa 3.8 iterative solution

```
//signals and systems
//time domain analysis of discreet time systems
//iterative solution
clear all;
close;
clc;
n=(-1:10)';
y=[16;0;zeros(length(n)-2,1)];
x=[0;0;n(3:length(n))];
for k=1:length(n)-1
    y(k+1)=0.5*y(k)+x(k+1);
end;
clf;
```

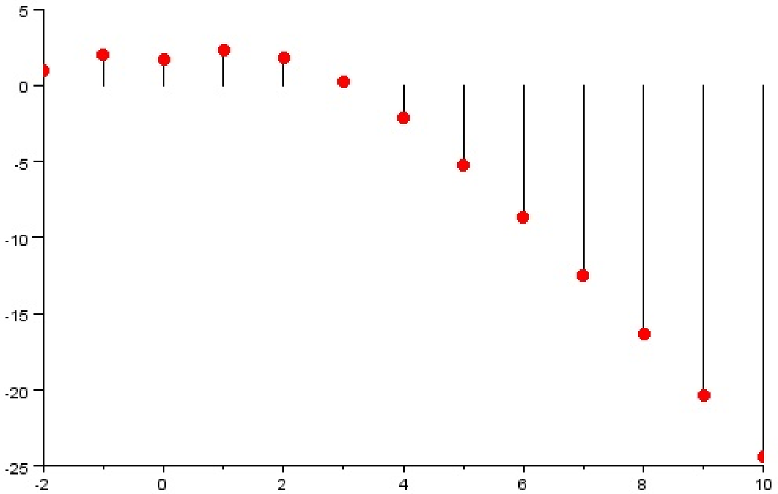

Figure 3.3: iterative solution

```
size(y)
size(n)
plot2d3(n,y);
plot(n,y,'r.')
disp([msprintf([n,y])]);
```

Scilab code Exa 3.9 iterative solution

```
//signals and systems
//time domain analysis of discreet time systems
//iterative solution
clear all;
close;
clc;
```

```
n=(-2:10)';
y=[1;2;zeros(length(n)-2,1)];
x=[0;0;n(3:length(n))];
for k=1:length(n)-2
    y(k+2)=y(k+1)-0.24*y(k)+x(k+2)-2*x(k+1);
end;
clf;
plot2d3(n,y);
disp([msprintf([n,y])]);
```

Scilab code Exa 3.10 total response with given initial conditions

```
//signals and systems
//time domain analysis of discreet time systems
//total response with initial conditions
clear all;
close;
clc;
n=(-2:10)';
y=[25/4;0;zeros(length(n)-2,1)];
x=[0;0;4^-n(3:length(n))];
for k=1:length(n)-2
    y(k+2)=0.6*y(k+1)+0.16*y(k)+5*x(k+2);
end;
clf;
a=gca();
plot2d3(n,y);

y1=[25/4;0;zeros(length(n)-2,1)];
x=[0;0;4^-n(3:length(n))];
for k=1:length(n)-2
    y1(k+2)=-6*y1(k+1)-9*y1(k)+2*x(k+2)+6*x(k+1);
end
figure
a=gca();
```

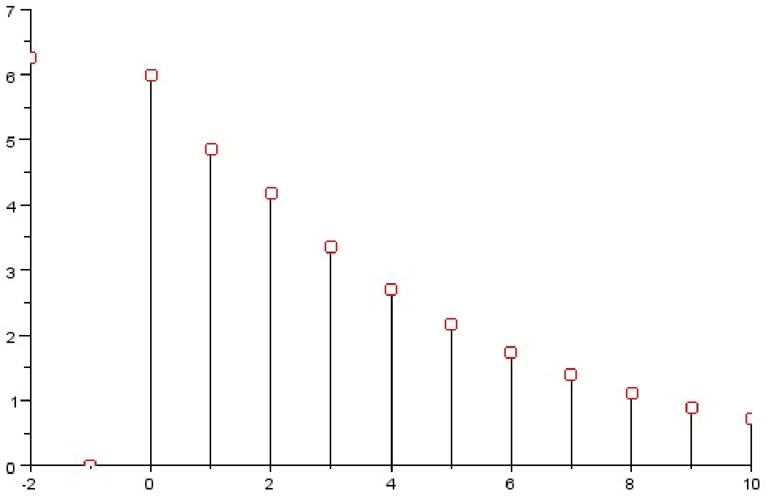

Figure 3.4: total response with given initial conditions

```
plot2d3(n,y1);


y2=[25/4;0;zeros(length(n)-2,1)];
x=[0;0;4^-n(3:length(n))];
for k=1:length(n)-2
    y2(k+2)=1.56*y2(k+1)-0.81*y2(k)+ x(k+1)+3*x(k);
end
figure
a=gca();
plot2d3(n,y2);
```

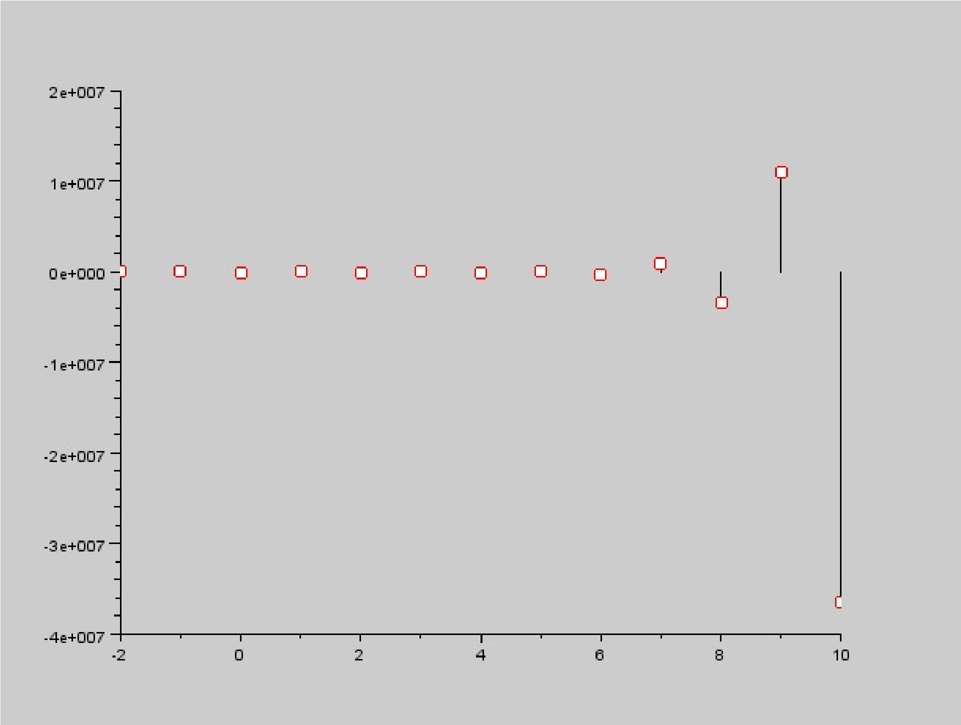

Figure 3.5: total response with given initial conditions

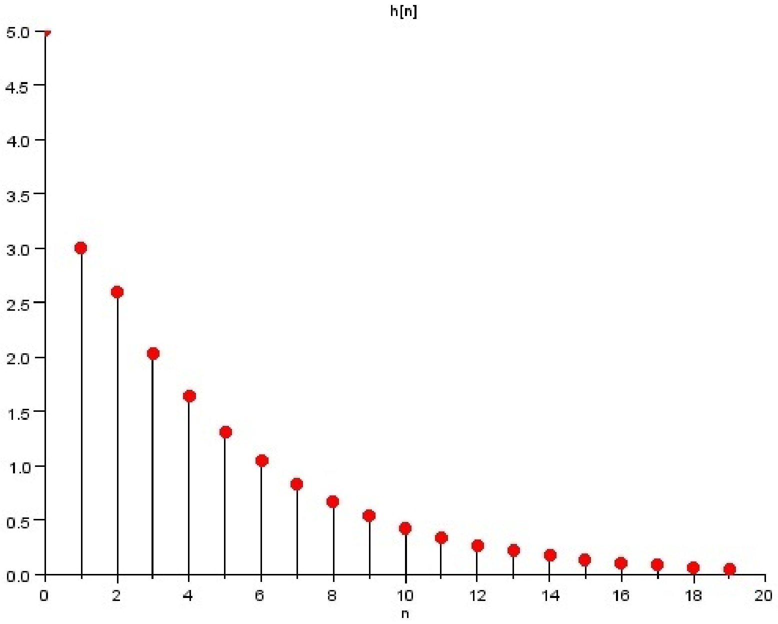

Figure 3.6: iterative determination of unit impulse response

Scilab code Exa 3.11 iterative determination of unit impulse response

```
//signals and systems
//time domain analysis of discreet time systems
//impulse response with initial conditions
clear all;
close;
clc;
n=(0:19);
x=[1 zeros(1,length(n)-1)];
a=[1 -0.6 -0.16];
b=[5 0 0];
h=filter(b,a,x);
clf;
plot2d3(n,h); xlabel('n'); ylabel('h[n]');
```

Scilab code Exa 3.13 convolution of discrete signals

```
//signals and systems
//time domain analysis of discreet time systems
//convolution
clear all;
close;
clc;
n=(0:19);
x=0.8^n;
g=0.3^n;
n1=(0:1:length(x)+length(g)-2);
c=convol(x,g);
plot2d3(n1,c);
```

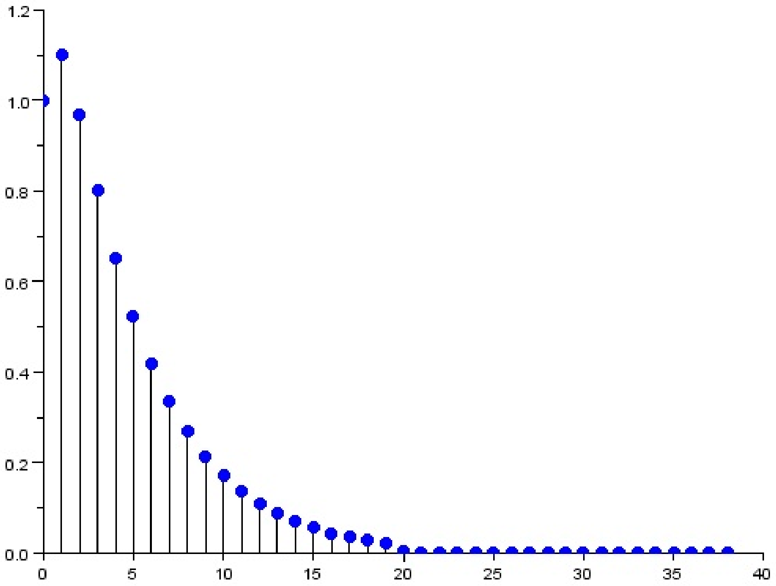

Figure 3.7: convolution of discrete signals

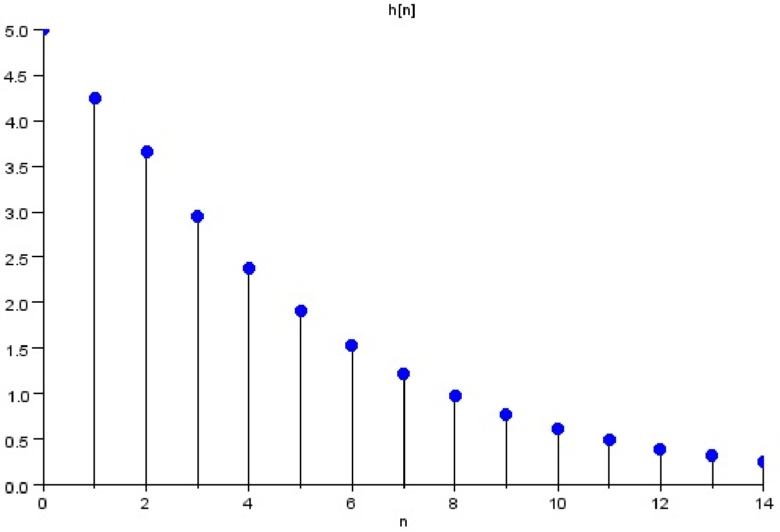

Figure 3.8: convolution of discrete signals

Scilab code Exa 3.14 convolution of discrete signals

```
//signals and systems
//time domain analysis of discreet time systems
//convolution
clear all;
close;
clc;
n=(0:14);
x=4^-n;
a=[1 -0.6 -0.16];
b=[5 0 0];
y=filter(b,a,x);
```

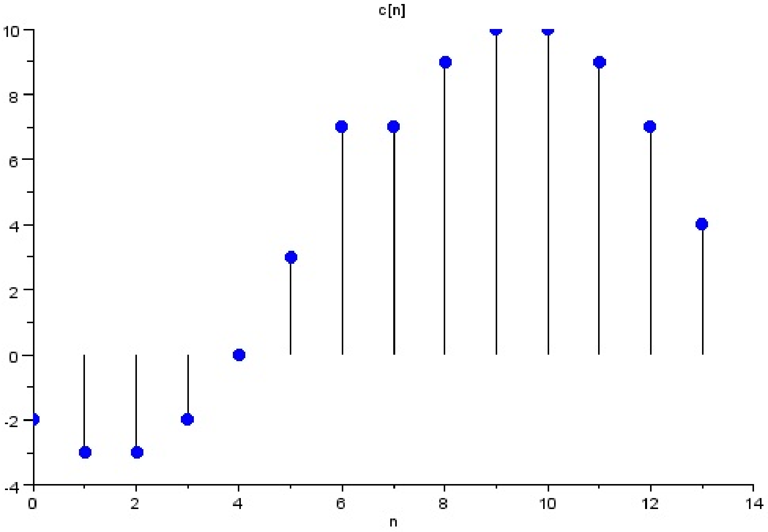

Figure 3.9: sliding tape method of convolution

```
clf;
plot2d3(n,y); xlabel('n'); ylabel('y[n]');
```

Scilab code Exa 3.16 sliding tape method of convolution

```
//signals and systems
//time domain analysis of discreet time systems
//convolution by sliding tape method
clear all;
close;
clc;
x=[-2 -1 0 1 2 3 4];
g=[1 1 1 1 1 1 1 1];
n=(0:1:length(x)+length(g)-2);
```

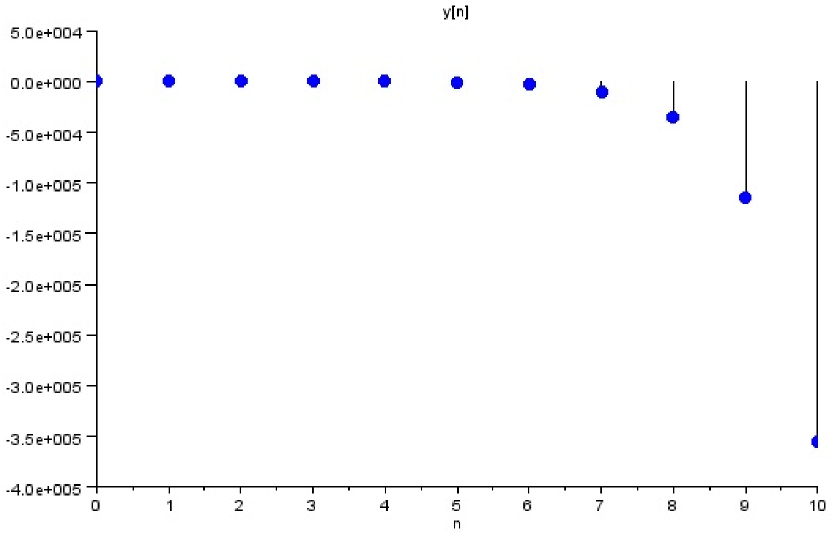

Figure 3.10: total response with given initial conditions

```
c=convol(x,g);
clf;
plot2d3(n,c); xlabel('n'); ylabel('c[n]');
```

Scilab code Exa 3.17 total response with given initial conditions

```
//signals and systems
//time domain analysis of discreet time systems
//convolution by sliding tape method
clear all;
close;
clc;
n=(0:10)';
y=[4;13;zeros(length(n)-2,1)];
```

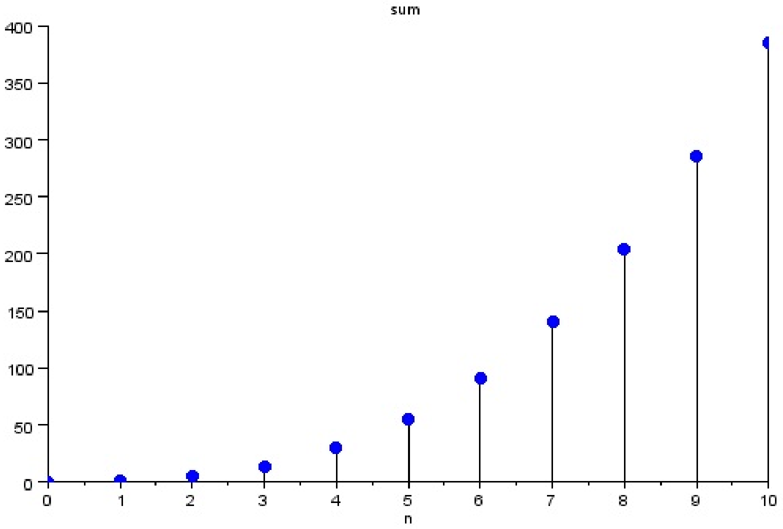

Figure 3.11: total response with given initial conditions

```
x=(3*n+5).*(n>=0);
for k=1:length(n)-2
    y(k+2)=5*y(k+1)-6*y(k)+x(k+1)-5*x(k);
end
clf;
plot2d3(n,y); xlabel('n'); ylabel('y[n]');
disp('n                                  y');
disp(msprintf('%f\t\t%f\n',[n,y]));
```

Scilab code Exa 3.18 total response with given initial conditions

```
//signals and systems
//time domain analysis of discreet time systems
//convolution by sliding tape method
```

```
clear all;
close;
clc;
n=(0:10)';
y=[0;zeros(length(n)-1,1)];
x=(n+1)^2;
for k=1:length(n)-1
    y(k+1)=y(k)+x(k);
end;
clf;
a=gca();
plot2d3(n,y);xtitle('sum','n')
plot(n,y,'b.')
```

Scilab code Exa 3.19 forced response

```
//signals and systems
//time domain analysis of discreet time systems
//convolution by sliding tape method
clear all;
close;
clc;
n=(0:14);
x=3^n;
a=[1 -3 2];
b=[0 1 2];
y=filter(b,a,x);
clf;
plot2d3(n,y); xlabel('n'); ylabel('y[n]');
```

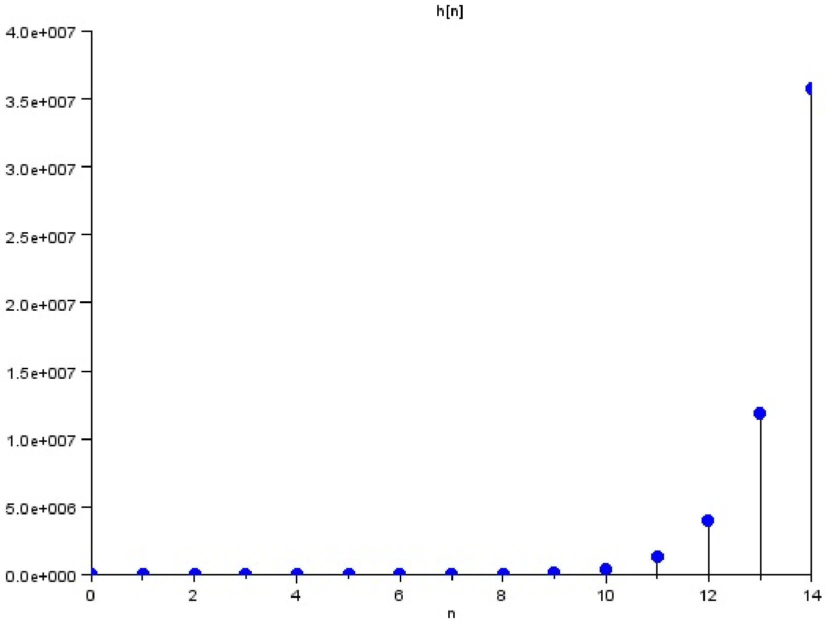

Figure 3.12: forced response

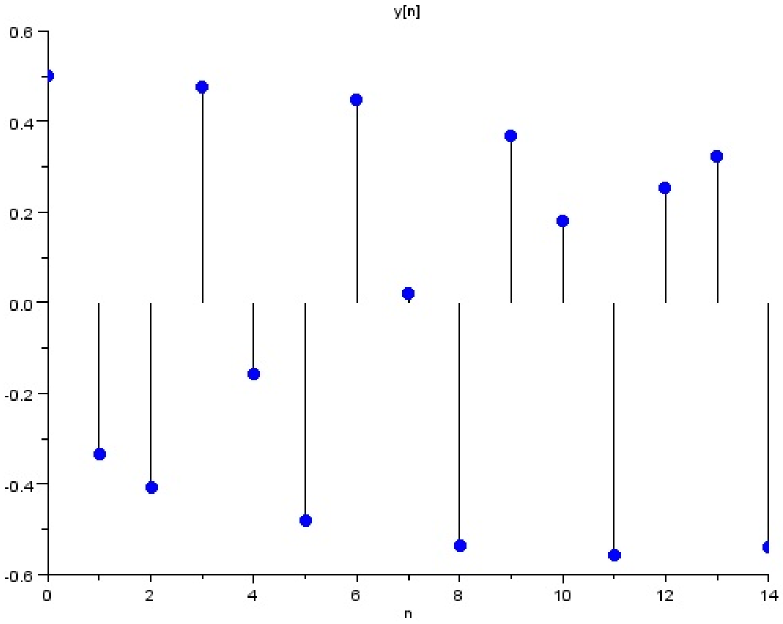

Figure 3.13: forced response

Scilab code Exa 3.20 forced response

```
//signals and systems
//time domain analysis of discreet time systems
//convolution by sliding tape method
clear all;
close;
clc;
pi=3.14;
n=(0:14);
x=cos(2*n+pi/3);
a=[1 -1 0.16];
b=[0 1 0.32];
y=filter(b,a,x);
clf;
plot2d3(n,y); xlabel('n'); ylabel('y[n]');
```

Chapter 4

continuous time system analysis

Scilab code Exa 4.1 laplace transform of exponential signal

```
//signals and systems
//Laplace Transform x(t) = exp(-at).u(t) for t negative and positive
syms t s;
a = 3;
y =laplace('%e^(-a*t)',t,s);
t1=0:0.001:10;
plot2d(t1,exp(-a*t1));
disp(y)
y1 = laplace('%e^(a*-t)',t,s);
disp(y1)
```

Scilab code Exa 4.2 laplace transform of given fsignal

```
//signals and systems
//(a) laplace transform x(t) = del(t)
syms t s;
```

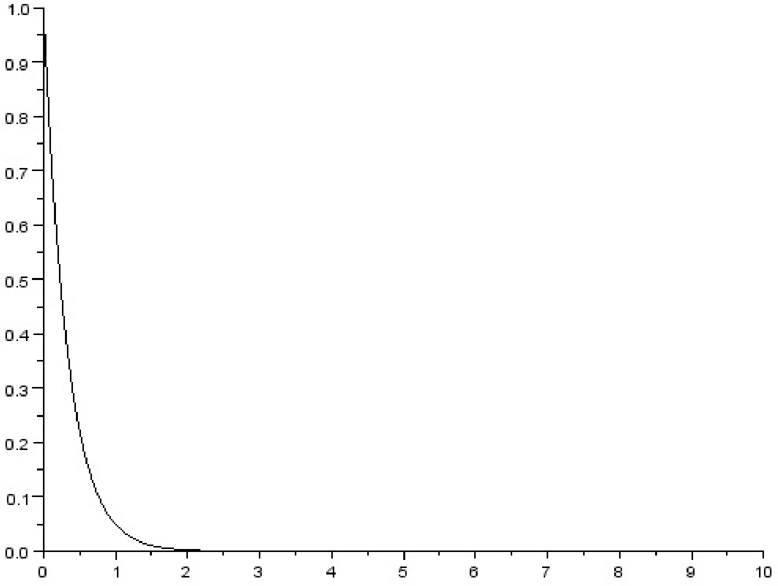

Figure 4.1: laplace transform of exponential signal

```

y =laplace('0',t,s)
disp(y)
//(b) Laplace Transform x(t) = u(t)

y1 =laplace('1',t,s);
disp(y1)
//(c) laplace transform x(t) = cos(w0*t)u(t)

y2 =laplace('cos(w0*t)',t,s);
disp(y2)
```

Scilab code Exa 4.3.a laplace transform in case of different roots

```
//signals and systems
//Inverse Lapalce Transform
//(a) X(S) = (7s-6)/s^2-s-6  Re(s)>-1
s =%s ;
syms t ;
[A]=pfss((7*s-6)/((s^2-s-6))); //partial fraction of
    F(s)
F1 = ilaplace(A(1),s,t)
F2 = ilaplace(A(2),s,t)
//F3 = ilaplace(A(3),s,t)
F = F1+F2;
disp(F,"f(t)=")
```

Scilab code Exa 4.3.b laplace transform in case of similar roots

```
//example 4.3
//(b) X(S) = (2*s^2+5)/s^2-3*s+2  Re(s)>-1
s =%s ;
syms t ;
```

```
[A]=pfss((2*s^2+5)/((s^2-3*s+2))); //partial fraction of F(s)
F1 = ilaplace(A(1),s,t)
F2 = ilaplace(A(2),s,t)
//F3 = ilaplace(A(3),s,t)
F = F1+F2;
disp(F,”f(t)=”)
```

Scilab code Exa 4.3.c laplace transform in case of imaginary roots

```
//example4.3
//(c) X(S) = 6(s+34)/s(s^2+10*s+34)  Re(s)>-1
s =%s ;
syms t ;
[A]=pfss((6*(s+34))/(s*(s^2+10*s+34))); //partial fraction of F(s)
F1 = ilaplace(A(1),s,t)
F2 = ilaplace(A(2),s,t)
//F3 = ilaplace(A(3),s,t)
F = F1+F2;
disp(F,”f(t)=”)
```

Scilab code Exa 4.4 laplace transform of a given signal

```
//signals and systems
//Lapalce Transform x(t) = (t-1)u(t-1)-(t-2)u(t-2)-u(t-4), 0<t<T
syms t s;
a = 3;
T = 1;
//t = T;
y1 = laplace('t',t,s);
y2 = laplace('t',t,s);
```

```
y3 = laplace('1',t,s);
y=y1*(%e^(-s))+y2*(%e^(-2*s))+y3*(%e^(-4*s))
disp(y)
```

Scilab code Exa 4.5 inverse laplace transform

```
//signals and systems
//example4.5
// X(S) = s+3+5*exp(-2*s)/(s+1)*(s+2))  Re(s)>-1
s1 =%s ;
syms t s;
[A]=pfss((s1+3)/((s1+1)*(s1+2))); //partial fraction
    of F(s)
F1 = ilaplace(A(1),s,t)
F2 = ilaplace(A(2),s,t)
//F3 = ilaplace(A(3),s,t)
Fa = F1+F2;
disp(Fa,"f1(t)=")
[B]=pfss((5)/((s1+1)*(s1+2))); //partial fraction of
    F(s)
F1 = ilaplace(B(1),s,t)
F2 = ilaplace(B(2),s,t)
Fb = (F1+F2)*(%e^(-2*s));
disp(Fb,"f2(t)=")
disp(Fa+Fb,"f(t)=")
```

Scilab code Exa 4.8 time convolution property

```
 //signals and systems
//Example 4.8
//Lapalce Transform for convolution
s=%s
syms t ;
```

```
a=3;b=2;
[A]=pfss(1/(s^2-5*s+6)); //partial fraction of F(s)
F1 = ilaplace(A(1),s,t)
F2 = ilaplace(A(2),s,t)
//F3 = ilaplace(A(3),s,t)
F = F1+F2;
disp(F,"f(t)=")
```

Scilab code Exa 4.9 initial and final value

```
//Initial and final Value Theorem of Lapalace Transform
syms s;
num =poly([30 20],'s','coeff')
den =poly([0 5 2 1],'s','coeff')
X = num/den
disp (X,"X(s)=")
SX = s*X;
Initial_Value =limit(SX,s,%inf);
final_value =limit(SX,s,0);
disp(Initial_Value,"x(0)=")
disp(final_value,"x(inf)=")
```

Scilab code Exa 4.10 second order linear differential equation

```
//signals and systems
//Unilateral  Laplace Transform:Solving Differential Equation
//example 4.10
s = %s;
syms t;
[A] = pfss((2*s^2+20*s+45)/((s+2)*(s+3)*(s+4)));
F1 = ilaplace(A(1),s,t)
```

```
F2 = ilaplace(A(2),s,t)
F3 = ilaplace(A(3),s,t)
F = F1+F2+F3
disp(F)
```

Scilab code Exa 4.11 solution to ode using laplace transform

```
//signals and systems
//Unilateral  Laplace Transform:Solving Differential Equation
//example 4.11
s = %s;
syms t;
[A] = pfss((2*s)/(s^2+2*s+5));
F1 = ilaplace(A(1),s,t)
//F2 = ilaplace(A(2),s,t)
//F3 = ilaplace(A(3),s,t)
F = F1+F2+F3
disp(F)
```

Scilab code Exa 4.12 response to LTIC system

```
//signals and systems
//Unilateral  Laplace Transform:Solving Differential Equation
//example 4.12
s = %s;
syms t;
[A] = pfss((3*s+3)/((s+5)*(s^2+5*s+6)));
F1 = ilaplace(A(1),s,t)
F2 = ilaplace(A(2),s,t)
F3 = ilaplace(A(3),s,t)
F = F1+F2+F3
```

```
disp(F)
```

Scilab code Exa 4.15 loop current in a given network

```
//signals and systems
//Unilateral  Laplace Transform:Solving Differential
    Equation
//example 4.15
s = %s;
syms t;
[A] = pfss((10)/(s^2+3*s+2));
F1 = ilaplace(A(1),s,t)
F2 = ilaplace(A(2),s,t)
//F3 = ilaplace(A(3),s,t)
F = F1+F2+F3
disp(F)
```

Scilab code Exa 4.16 loop current in a given network

```
//signals and systems
//Unilateral  Laplace Transform: transfer function
//example 4.16
s = %s;
syms t s;
y1 =laplace('24*%e^(-3*t)+48*%e^(-4*t)',t,s);
disp(y1)
y2 =laplace('16*%e^(-3*t)-12*%e^(-4*t)',t,s);
disp(y2)
```

Scilab code Exa 4.17 voltage and current of a given network

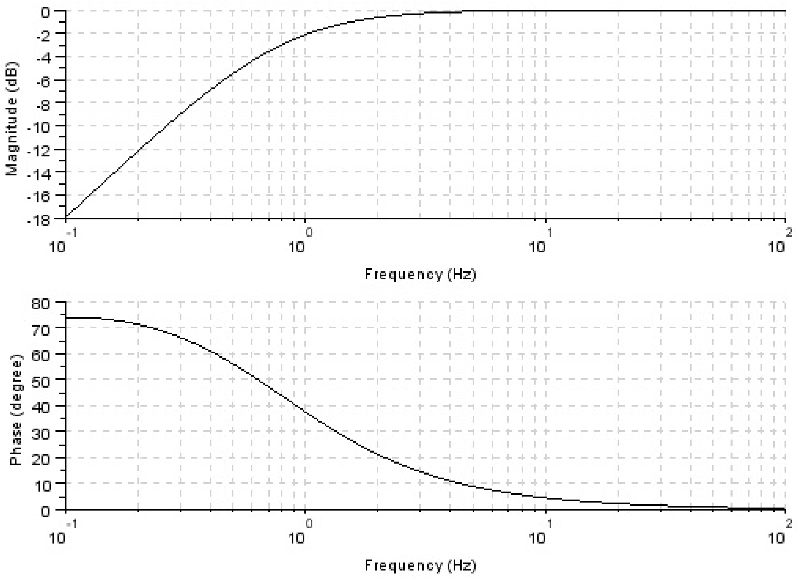

Figure 4.2: frequency response of a given system

```
//signals and systems
//Unilateral Laplace Transform:Solving Differential Equation
//example 4.17
s= %s;
syms t;
[A] = pfss((2*s^2+9*s+4)/((s)*(s^2+3*s+1)));
F1 = ilaplace(A(1),s,t)
F2 = ilaplace(A(2),s,t)
F3 = ilaplace(A(3),s,t)
F = F1+F2+F3
disp(F)
```

Scilab code Exa 4.23 frequency response of a given system

```
s=poly(0,'s')
h=syslin('c',(s+0.1)/(s+5))
clf();bode(h,0.1,100);
```

Scilab code Exa 4.24 frequency response of a given system

```
s=poly(0,'s')
h=syslin('c',(s^2/s))
clf();bode(h,0.1,100);
h1=syslin('c',(1/s))
clf(); bode(h1,0.1,100);
```

Scilab code Exa 4.25 bode plots for given transfer function

```
s=poly(0,'s')
h=syslin('c',((20*s^2+2000*s)/(s^2+12*s+20)))
clf();bode(h,0.1,100);
```

Scilab code Exa 4.26 bode plots for given transfer function

```
s=poly(0,'s')
h=syslin('c',((10*s+1000)/(s^2+2*s+100)))
clf();bode(h,0.1,100);
```

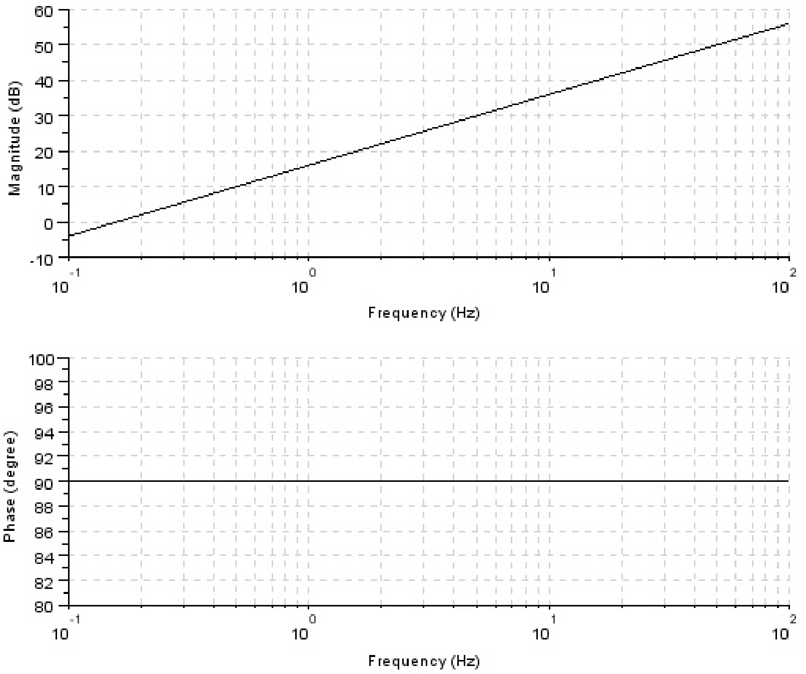

Figure 4.3: frequency response of a given system

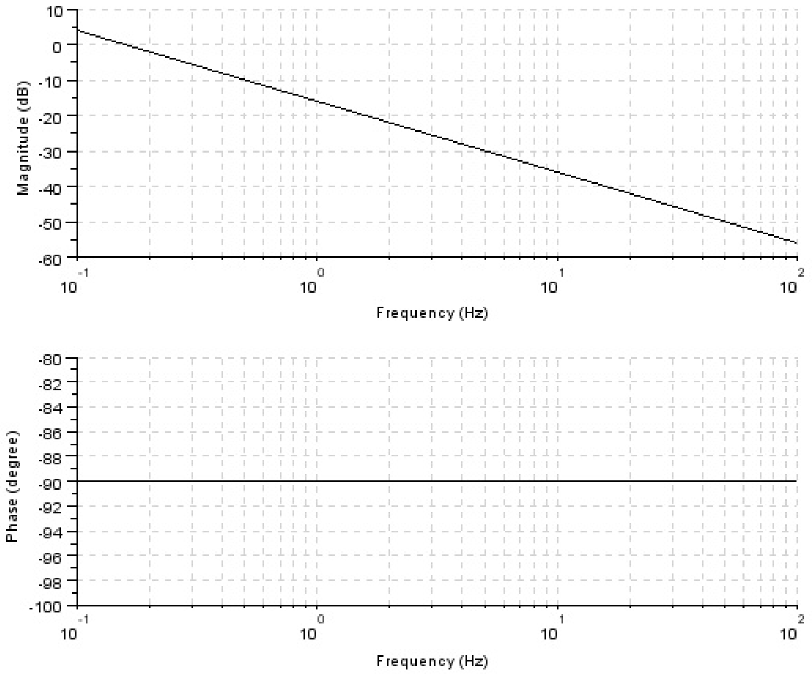

Figure 4.4: frequency response of a given system

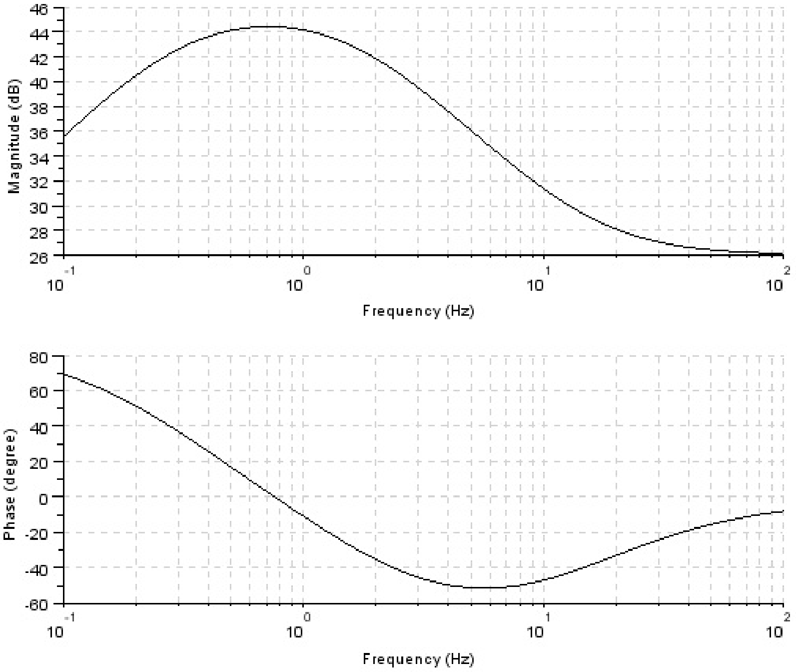

Figure 4.5: bode plots for given transfer function

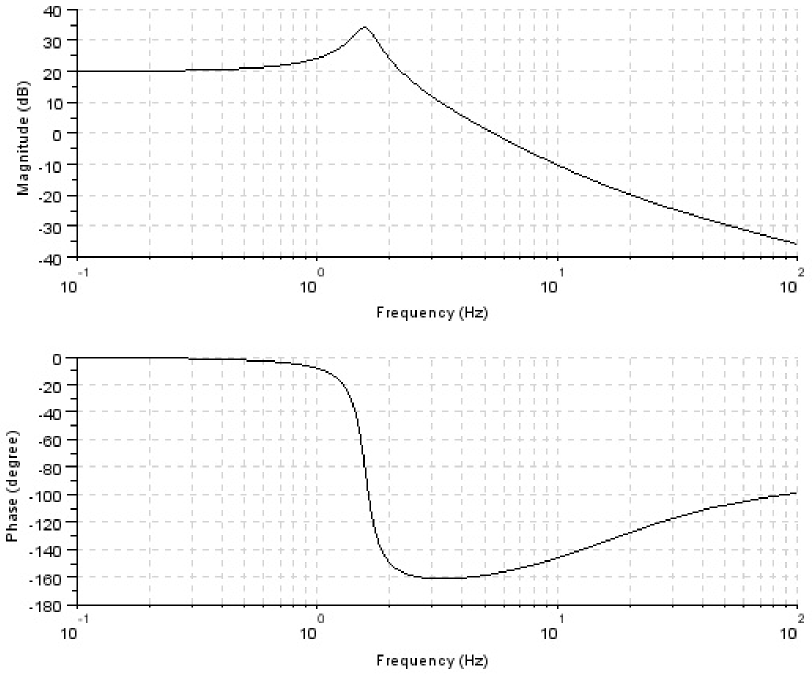

Figure 4.6: bode plots for given transfer function

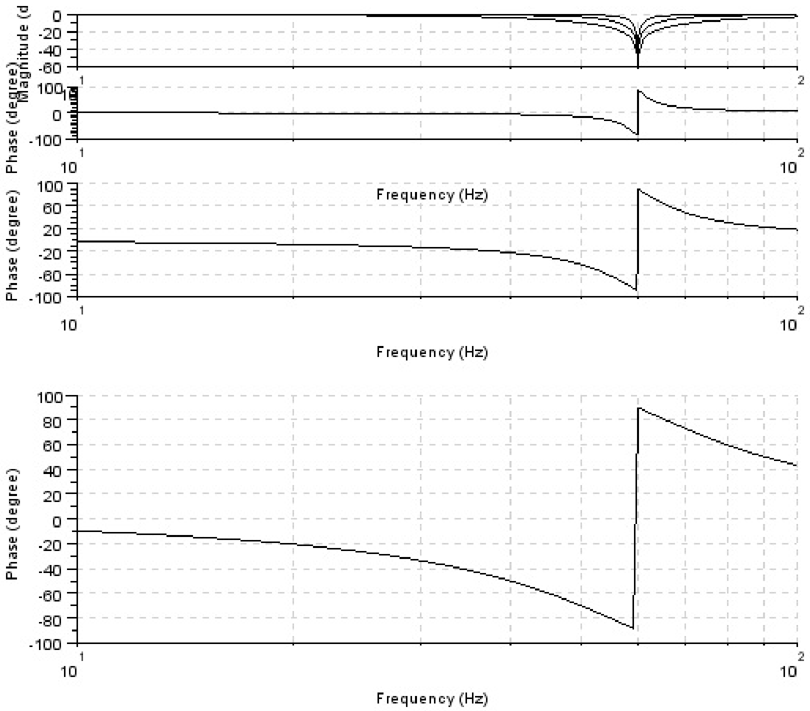

Figure 4.7: second order notch filter to suppress 60Hz hum

Scilab code Exa 4.27 second order notch filter to suppress 60Hz hum

```
omega_0=2*%pi*60; theta = [60 80 87]*(%pi/180);
omega = (0:0.5:1000)'; mag = zeros(3,length(omega));
s=poly(0,'s')
for m =1:length(theta)
    H=syslin('c',((s^2+omega_0^2)/(s^2+2*omega_0*cos
       (theta(m))*s +omega_0^2)));
    bode(H,10,100);
end
```

```
f=omega/((2*%pi))plot(f,mag(1,:),'k-',f mag(2,:),'k--',f,mag(3,:),'k-.');
xlabel('f[hz]'); ylabel('|H(j2/pi f)|');
legend('\theta=60^\circ','\theta = 80^\circ','\theta = 87^\circ',0)
```

Scilab code Exa 4.28 bilateral inverse transform

```
//signals and systems
//bilateral Inverse Lapalce Transform
//X(S) = 1/((s-1)(s+2))
s =%s ;
syms t ;
[A]=pfss(1/((s-1)*(s+2))) //partial fraction of F(s)
F1 = ilaplace(A(1),s,t)
F2 = ilaplace(A(2),s,t)
F=F1+F2;
disp(F,"f(t)=")


//X(S) = 1/((s-1)(s+2))  Re(s)> -1,Re(s)< -2
s =%s ;
syms t ;
[A]=pfss(1/((s-1)*(s+2))) //partial fraction of F(s)
F1 = ilaplace(A(1),s,t)
F2 = ilaplace(A(2),s,t)
F = -F1-F2;
disp(F,"f(t)=")


//X(S) = 1/((s-1)(s+2))  -2< Re(s)< 1
s =%s ;
syms t ;
[A]=pfss(1/((s-1)*(s+2))) //partial fraction of F(s)
F1 = ilaplace(A(1),s,t)
```

```
F2 = ilaplace(A(2),s,t)
F = -F1+F2;
disp(F,"f(t)=")
```

Scilab code Exa 4.29 current for a given RC network

```
//signals and systems
//Unilateral  Laplace Transform:Solving Differential
    Equation
//example 4.30
s= %s;
syms t;
[A] = pfss((-s)/((s-1)*(s-2)*(s+1)));
F1 = ilaplace(A(1),s,t)
F2 = ilaplace(A(2),s,t)
F3 = ilaplace(A(3),s,t)
F = F1+F2+F3
disp(F)
```

Scilab code Exa 4.30 response of a noncausal sytem

```
//signals and systems
//Unilateral  Laplace Transform:Solving Differential
    Equation
//example 4.30
s= %s;
syms t;
[A] = pfss((-1)/((s-1)*(s+2)));
F1 = ilaplace(A(1),s,t)
F2 = ilaplace(A(2),s,t)
//F3 = ilaplace(A(3),s,t)
F = F1+F2
disp(F)
```

Scilab code Exa 4.31 response of a fn with given tf

```
//signals and systems
//Unilateral  Laplace Transform:Solving Differential
   Equation
//example 4.17
s= %s;
syms t;
// Re s>-1
[A] = pfss(1/((s+1)*(s+5)));
F1 = ilaplace(A(1),s,t)
F2 = ilaplace(A(2),s,t)
//F3 = ilaplace(A(3),s,t)
F = F1+F2
disp(F)
//-5< Re s <-2
[B] = pfss(-1/((s+2)*(s+5)));
G1 = ilaplace(B(1),s,t)
G2 = ilaplace(B(2),s,t)
//F3 = ilaplace(A(3),s,t)
G = G1+G2
disp(G)
```

Chapter 5

discrete time system analysis using the z transform

Scilab code Exa 5.1 z transform of a given signal

```
//signals and systems
// Ztransform of x[n]  = (a)^n.u[n]
syms n z;
a = 0.5;
x =(a)^n;
n1=0:10;
plot2d3(n1,a^n1); xtitle('a^n','n');
plot(n1,a^n1,'r.')
X = symsum(x*(z^(-n)),n,0,%inf)
disp(X,"ans=")
```

Scilab code Exa 5.2 z transform of a given signal

```
//example 5.2 (c)
```

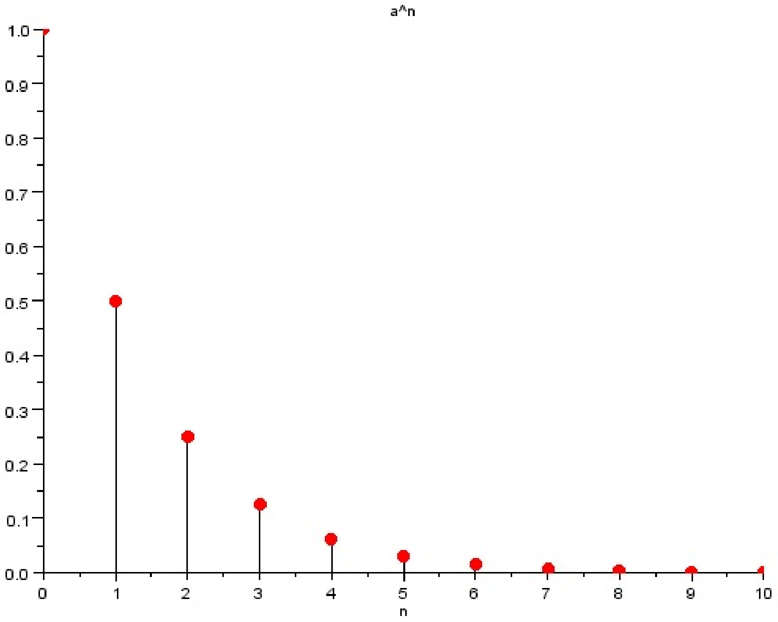

Figure 5.1: z transform of a given signal

```
//Z−transform of sine signal
syms n z;
Wo =%pi/4;
a = (0.33)^n;
x1=%e^(sqrt(-1)*Wo*n);
X1=symsum(a*x1*(z^(-n)),n,0,%inf)
x2=%e^(-sqrt(-1)*Wo*n)
X2=symsum(a*x2*(z^(-n)),n,0,%inf)
X =(1/(2*sqrt(-1)))*(X1+X2)
disp(X,"ans=")

//example 5.2 (a)
//Z−transform of Impulse Sequence
syms n z;
X=symsum(1*(z^(-n)),n,0,0);
disp(X,"ans=")

//example 5.2 (d)
//Z−transform of given Sequence
syms n z;
X=symsum(1*(z^(-n)),n,0,4);
disp(X,"ans=")

//example 5.2 (b)
//Z−transform of unit function Sequence
syms n z;
X=symsum(1*(z^(-n)),n,0,%inf);
disp(X,"ans=")
```

Scilab code Exa 5.3.a z transform of a given signal with different roots

```
//signals and systems
//Inverse Z Transform:ROC |z|>1/3
z = %z;
syms n z1;//To find out Inverse z transform z must
```

```
   be linear z = z1
X  =(8*z-19)/((z-2)*(z-3))
X1 = denom(X);
zp = roots(X1);
X1 = (8*z1-19)/((z1-2)*(z1-3))
F1 = X1*(z1^(n-1))*(z1-zp(1));
F2 = X1*(z1^(n-1))*(z1-zp(2));
h1 = limit(F1,z1,zp(1));
disp(h1,'h1[n]=')
h2 = limit(F2,z1,zp(2));
disp(h2,'h2[n]=')
h = h1+h2;
disp(h,'h[n]=')
```

Scilab code Exa 5.3.c z transform of a given signal with imaginary roots

```
//signals and systems
//Inverse Z Transform:ROC |z|>1/3
z = %z;
syms n z1;//To find out Inverse z transform z must
   be linear z = z1
X  =(2*z*(3*z+17))/((z-1)*(z^2-6*z+25))
X1 = denom(X);
zp = roots(X1);
X1 = 2*z1*(3*z1+17)/((z1-1)*(z1^2-6*z1+25))
F1 = X1*(z1^(n-1))*(z1-zp(1));
F2 = X1*(z1^(n-1))*(z1-zp(2));
h1 = limit(F1,z1,zp(1));
disp(h1,'h1[n]=')
h2 = limit(F2,z1,zp(2));
disp(h2,'h2[n]=')
h = h1+h2;
disp(h,'h[n]=')
```

Scilab code Exa 5.5 solution to differential equation

```
//LTi Systems characterized by Linear Constant
//Coefficient Difference equations
//Inverse Z Transform
//z = %z;
syms n z;
H1 = (26/15)/(z-(1/2));
H2 = (7/3)/(z-2);
H3 = (18/5)/(z-3);
F1 = H1*z^(n)*(z-(1/2));
F2 = H2*z^(n)*(z-2);
F3 = H3*z^(n)*(z-3);
h1 = limit(F1,z,1/2);
disp(h1,'h1[n]=')
h2 = limit(F2,z,2);
disp(h2,'h2[n]=')
h3 = limit(F3,z,3);
disp(h3,'h3[n]=')
h = h1-h2+h3;
disp(h,'h[n]=')
```

Scilab code Exa 5.6 response of an LTID system using difference eq

```
//LTi Systems characterized by Linear Constant
//Coefficient Difference equations
//Inverse Z Transform
//z = %z;
syms n z;
H1 = (2/3)/(z+0.2);
H2 = (8/3)/(z+0.8);
H3 = (2)/(z+0.5);
```

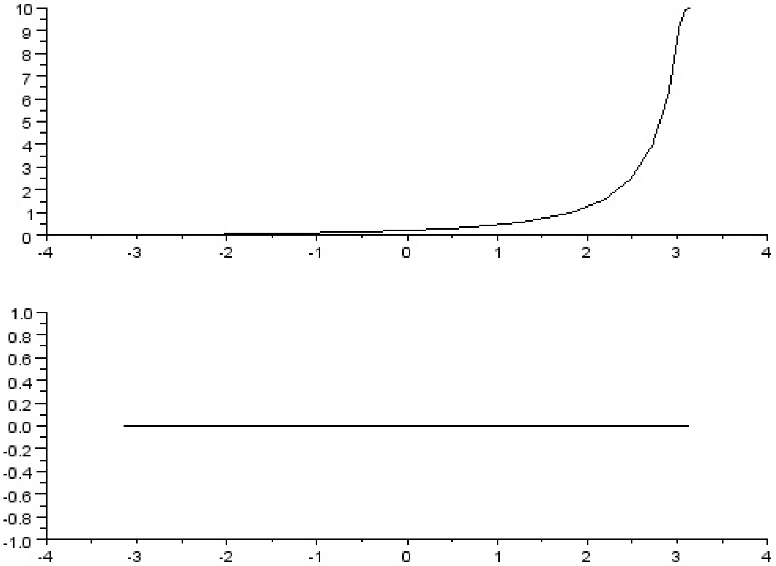

Figure 5.2: response of an LTID system using difference eq

```
F1 = H1*z^(n)*(z+0.2);
F2 = H2*z^(n)*(z+0.8);
F3 = H3*z^(n)*(z+0.5);
h1 = limit(F1,z,-0.2);
disp(h1,'h1[n]=')
h2 = limit(F2,z,-0.8);
disp(h2,'h2[n]=')
h3 = limit(F3,z,-0.5);
disp(h3,'h3[n]=')
h = h1-h2+h3;
disp(h,'h[n]=')
```

Scilab code Exa 5.10 response of an LTID system using difference eq

```
omega= linspace(-%pi,%pi,106);
H= syslin('c',(s/(s-0.8)));
H_omega= squeeze(calfrq(H,0.01,10));
size(H_omega)
subplot(2,1,1); plot2d(omega, abs(H_omega));
//xlabel('\omega');
//ylabel('|H[e^{j\omega}]|');
subplot(2,1,2); plot2d(omega,atan(imag(H_omega),real
    (H_omega))*180/%pi);
//xlabel('\omega');
//ylabel('\angle H[e^{j \omega}] [deg]');
```

Scilab code Exa 5.12 maximum sampling timeinterval

```
//signals and systems
//maximum sampling interval
f=50*10^3;
T=0.5/f;
disp(T)//in seconds
```

Scilab code Exa 5.13 discrete time amplifier highest frequency

```
//signals and systems
//highest frequency of a signal
T=25*10^-6
f=0.5/T
disp(f)//in hertz
```

Scilab code Exa 5.17 bilateral z transfrom

```
//Z transform of x[n] = a^n.u[n]+b^-n.u[-n-1]
syms n z;
a=0.9
b = 1.2;

x1=(a)^(n)
x2=(b)^(-n)
//plot2d3(n1,x1+x2)
X1=symsum(x1*(z^(-n)),n,0,%inf)
X2=symsum(x2*(z^(n)),n,1,%inf)
X = X1+X2;
disp(X,"ans=")
```

Scilab code Exa 5.18 bilateral inverse z transform

```
//signals and systems
//Inverse Z Transform:ROC |z|>2
z = %z;
syms n z1;//To find out Inverse z transform z must
   be linear z = z1
X  =-z*(z+0.4)/((z-0.8)*(z-2))
X1 = denom(X);
zp = roots(X1);
X1 = -z1*(z1+0.4)/((z1-0.8)*(z1-2))
F1 = X1*(z1^(n-1))*(z1-zp(1));
F2 = X1*(z1^(n-1))*(z1-zp(2));
h1 = limit(F1,z1,zp(1));
disp(h1,'h1[n]=')
h2 = limit(F2,z1,zp(2));
disp(h2,'h2[n]=')
h = h1+h2;
disp(h,'h[n]=')

//Inverse Z Transform:ROC 0.8<|z|<2
z = %z;
```

```
syms n z1;
X  =-z*(z+0.4)/((z-0.8)*(z-2))
X1 = denom(X);
zp = roots(X1);
X1 = -z1*(z1+0.4)/((z1-0.8)*(z1-2))
F1 = X1*(z1^(n-1))*(z1-zp(1));
F2 = X1*(z1^(n-1))*(z1-zp(2));
h1 = limit(F1,z1,zp(1));
disp(h1*'u(n)','h1[n]=')
h2 = limit(F2,z1,zp(2));
disp((h2)*'u(-n-1)','h2[n]=')
disp((h1)*'u(n)'-(h2)*'u(n-1)','h[n]=')

//Inverse Z Transform:ROC |z|<0.8
z = %z;
syms n z1;
X  =-z*(z+0.4)/((z-0.8)*(z-2))
X1 = denom(X);
zp = roots(X1);
X1 = -z1*(z1+0.4)/((z1-0.8)*(z1-2))
F1 = X1*(z1^(n-1))*(z1-zp(1));
F2 = X1*(z1^(n-1))*(z1-zp(2));
h1 = limit(F1,z1,zp(1));
disp(h1*'u(-n-1)','h1[n]=')
h2 = limit(F2,z1,zp(2));
disp((h2)*'u(-n-1)','h2[n]=')
disp(-(h1)*'u(-n-1)'-(h2)*'u(-n-1)','h[n]=')
```

Scilab code Exa 5.19 transfer function for a causal system

```
//LTi Systems characterized by Linear Constant
//Coefficient Difference equations
//Inverse Z Transform
//z = %z;
syms n z;
```

```
H1 = -z/(z-0.5);
H2 = (8/3)*z/(z-0.8);
H3=(-8/3)*z/(z-2);
F1 = H1*z^(n-1)*(z-0.5);
F2 = H2*z^(n-1)*(z-0.8);
F3 = H3*z^(n-1)*(z-2);
h1 = limit(F1,z,0.5);
disp(h1,'h1[n]=')
h2 = limit(F2,z,0.8);
disp(h2,'h2[n]=')
h3 = limit(F3,z,2);
disp(h3,'h3[n]=')
h = h1+h2+h3;
disp(h,'h[n]=')
```

Scilab code Exa 5.20 zero state response for a given input

```
//LTi Systems characterized by Linear Constant
//Coefficient Difference equations
//Inverse Z Transform
//z = %z;
syms n z;
H1 = (-5/3)*z/(z-0.5);
H2 = (8/3)*z/(z-0.8);
H3=5*z/(z-0.5);
H4=-6*z/(z-0.6);
F1 = H1*z^(n-1)*(z-0.5);
F2 = H2*z^(n-1)*(z-0.8);
F3 = H3*z^(n-1)*(z-0.5);
F4 = H4*z^(n-1)*(z-0.6);
h1 = limit(F1,z,0.5);
disp(h1,'h1[n]=')
h2 = limit(F2,z,0.8);
disp(h2,'h2[n]=')
h3 = limit(F3,z,0.5);
```

```
disp(h3,'h3[n]=')
h4 = limit(F4,z,0.6);
disp(h4,'h4[n]=')
h = h1+h2+h3+h4;
disp(h,'h[n]=')
```

Chapter 6

continuous time signal analysis the fourier series

Scilab code Exa 6.1 fourier coefficients of a periodic sequence

```
n=0:10;
a_n=0.504*2*ones(1,length(n))./(1+16*n.^2);
a_n(1)=0.504
b_n=0.504*8*n./(1+16*n.*n);
size(n)
size(a_n)
size(b_n)
disp(b_n(1))
C_n=sqrt(a_n.^2+(b_n).^2);
theta_n(1)=0; theta_n=atan(-b_n,a_n);
//n=[0,n];
clf;
size(n)
subplot(2,2,1); plot2d3(n,a_n);xtitle('a_n','n'); plot(n,a_n,'ro');
subplot(2,2,2); plot2d3(n,b_n);xtitle('b_n','n'); plot(n,b_n,'r.');
```

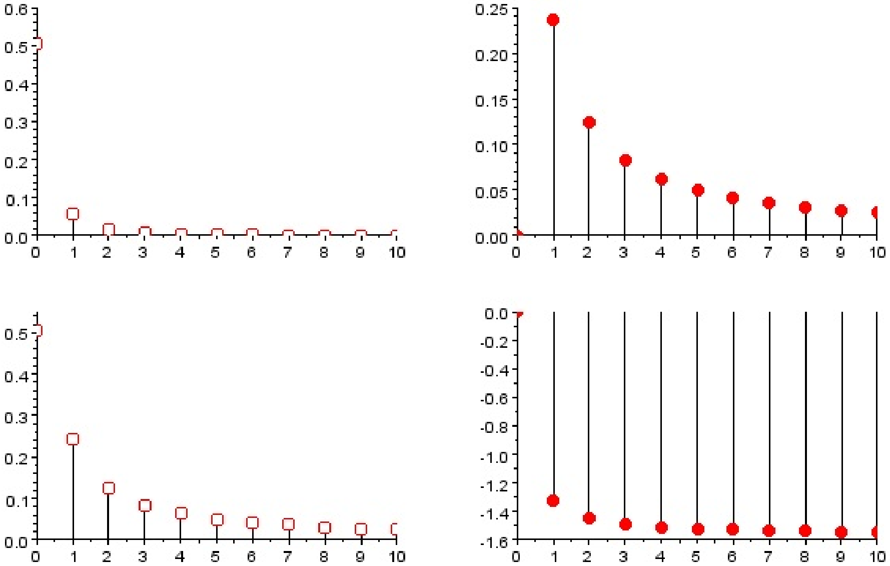

Figure 6.1: fourier coefficients of a periodic sequence

```
subplot(2,2,3); plot2d3(n,C_n);xtitle('C_n','n');plot(n,C_n,'ro');
subplot(2,2,4); plot2d3(n,theta_n,);xtitle('theta_n','n');plot(n,theta_n,'r.')
```

Scilab code Exa 6.2 fourier coefficients of a periodic sequence

```
 n=0:10;
a_n=zeros(1,length(n));
size(a_n)
b_n=(8/%pi^2*n.^2).*sin(n.*%pi/2);
size(n)
size(a_n)
size(b_n)
```

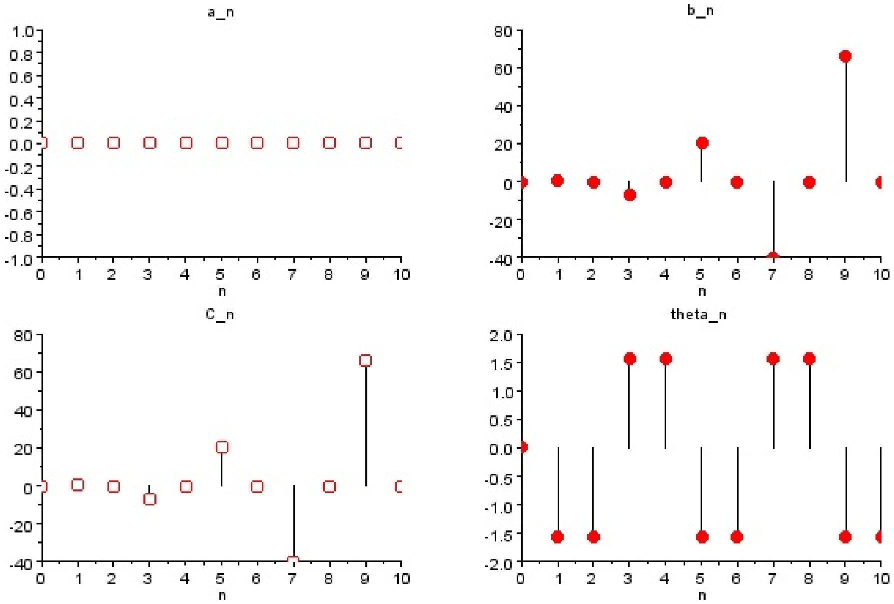

Figure 6.2: fourier coefficients of a periodic sequence

```
disp(b_n(1))
C_n=b_n
//theta_n(1)=0;
 theta_n=atan(-b_n,a_n);
//n=[0,n];
clf;
size(n)
subplot(2,2,1); plot2d3(n,a_n); xtitle('a_n','n');
   plot(n,a_n,'ro')
subplot(2,2,2); plot2d3(n,b_n);xtitle('b_n','n');
   plot(n,b_n,'r.')
subplot(2,2,3); plot2d3(n,C_n);xtitle('C_n','n');
   plot(n,C_n,'ro')
subplot(2,2,4); plot2d3(n,theta_n,);xtitle('theta_n'
   ,'n');plot(n,theta_n,'r.')
```

Scilab code Exa 6.3 fourier spectra of a signal

```
n=0:10;

for n=0:10
     //if(n%2==0)
       //  a_n=0;
     //else
         if (n==4*n-3)
             a_n=2/(%pi.*n);
         else if (n==4*n-1)
                 a_n=-2/(%pi.*n);
             end end end

b_n=zeros(1,length(n));
size(n)
size(a_n)
size(b_n)
disp(b_n(1))
C_n=sqrt(a_n.^2+(b_n).^2);
theta_n(1)=0; theta_n=atan(-b_n,a_n);
//n=[0,n];
clf;
size(n)
subplot(2,2,1); plot2d3(n,a_n);xtitle('a_n','n');plot(n,a_n,'ro');
subplot(2,2,2); plot2d3(n,b_n);xtitle('b_n','n');plot(n,b_n,'r.');
subplot(2,2,3); plot2d3(n,C_n);xtitle('C_n','n');plot(n,C_n,'ro');
subplot(2,2,4); plot2d3(n,theta_n,);xtitle('theta_n','n');plot(n,theta_n,'r.');
```

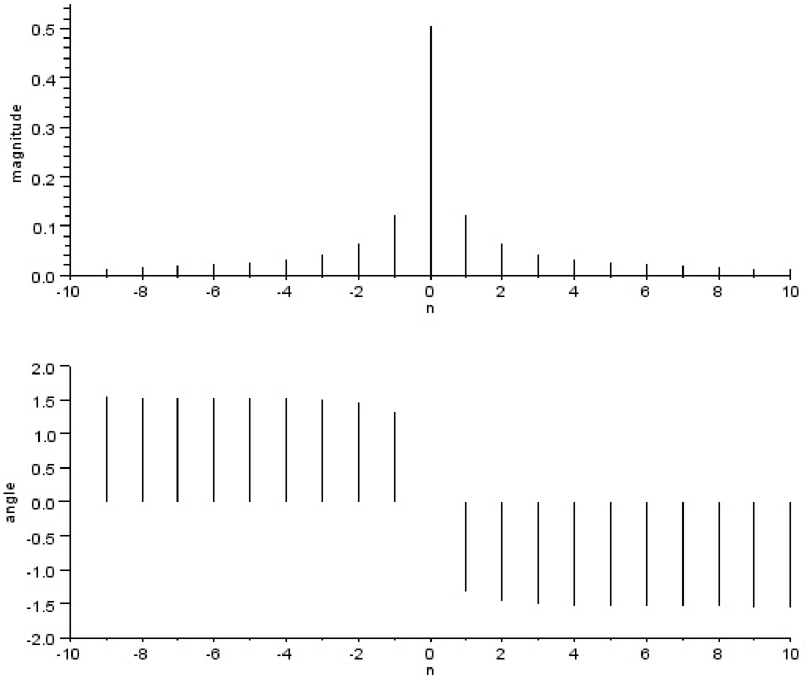

Figure 6.3: exponential fourier series

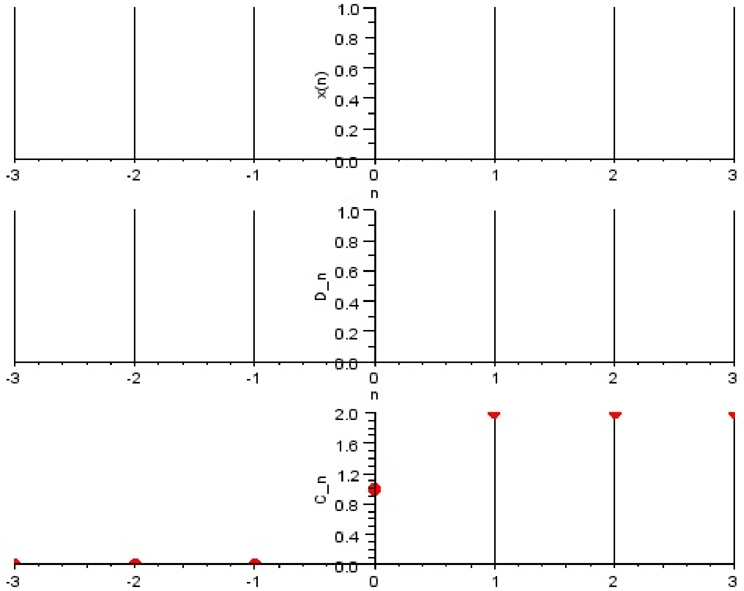

Figure 6.4: exponential fourier series for the impulse train

Scilab code Exa 6.5 exponential fourier series

```
n=(-10:10); D_n=0.504./(1+ %i*4*n);
clf;
subplot(2,1,1); plot2d3(n,abs(D_n));
subplot(2,1,2); plot2d3(n,atan(imag(D_n),real(D_n)))
    ;
```

Scilab code Exa 6.7 exponential fourier series for the impulse train

```
//signals and systems
//fourier series for train of impulses
clear;
close;
```

```
clc;
n=-3:1:3
x = ones(1,length(n))
D_n=ones(1,length(n));
C_n=[0 0 0 1 2 2 2]
subplot(3,1,1)
a = gca();
a.y_location = ”origin”;
a.x_location = ”origin”;
plot2d3(n,x)
subplot(3,1,2)
a = gca();
a.y_location = ”origin”;
a.x_location = ”origin”;
plot2d3(n,D_n)
subplot(3,1,3)
a = gca();
a.y_location = ”origin”;
a.x_location = ”origin”;
plot2d3(n,C_n); plot(n,C_n,’r.’)
```

Scilab code Exa 6.9 exponential fourier series to find the output

```
n=(-10:10); D_n=2/(3.14*(1-4.*n.^2).*(%i*6.*n+1));
clf;
subplot(2,1,1); plot2d3(n,abs(D_n));
subplot(2,1,2); plot2d3(n,atan(imag(D_n),real(D_n)))
    ;
```

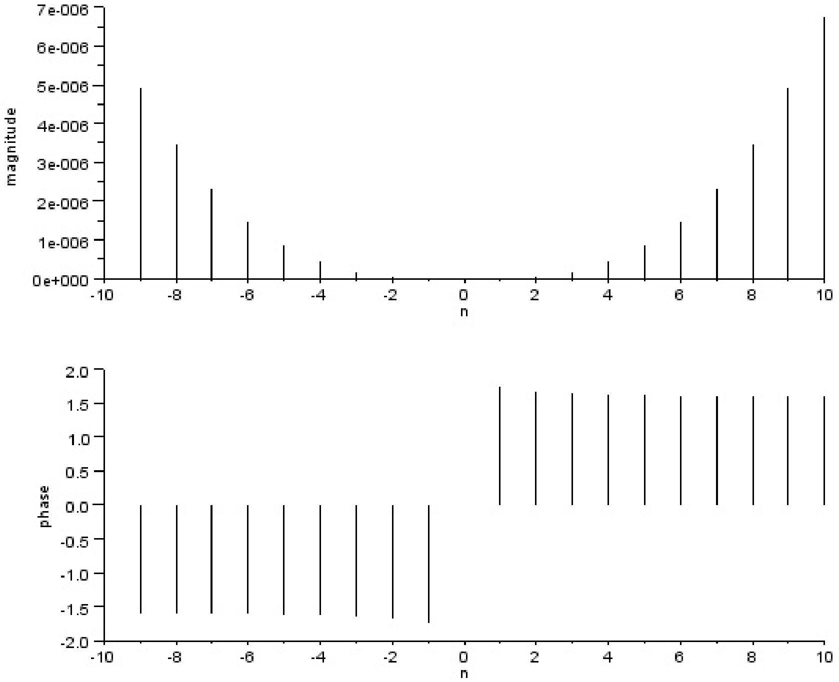

Figure 6.5: exponential fourier series to find the output

Chapter 7

continuous time signal analysis the fourier transform

Scilab code Exa 7.1 fourier transform of exponential function

```
//signals and systems
//continuous time signal analysis the fourier transform
//fourier transform of exp(-A*t)
clear;
clc;
A =1;     //Amplitude
Dt = 0.005;
t = -4.5:Dt:4.5;
xt = exp(-A*abs(t));
Wmax = 2*%pi*1;          //Analog Frequency = 1Hz
K = 4;
k = 0:(K/1000):K;
W = k*Wmax/K;
XW = xt* exp(-sqrt(-1)*t'*W) * Dt;
XW = real(XW);
W = [-mtlb_fliplr(W), W(2:1001)]; // Omega from -
```

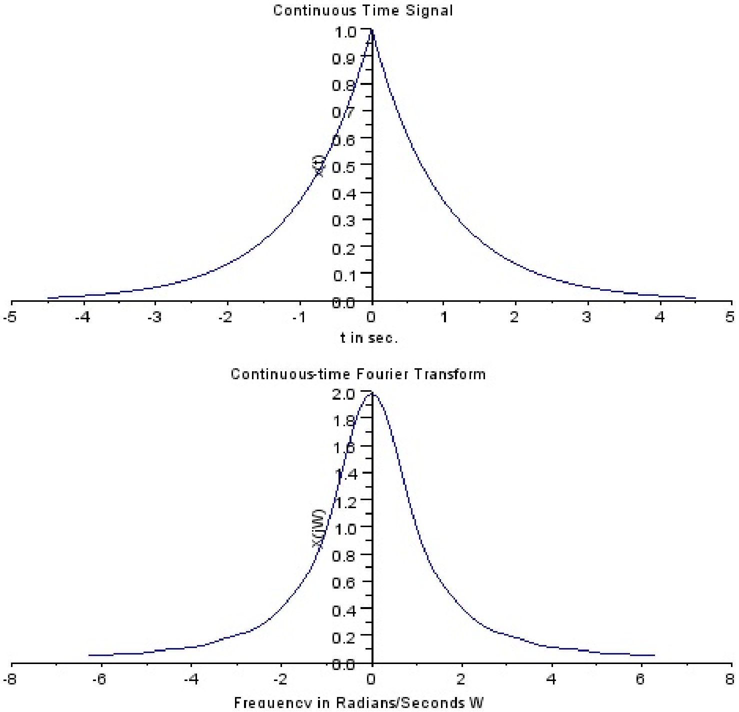

Figure 7.1: fourier transform of exponential function

```
    Wmax to Wmax
XW = [mtlb_fliplr(XW), XW(2:1001)];
subplot(2,1,1);
a = gca();
a.y_location = ”origin”;
plot(t,xt);
xlabel(’t in sec.’);
ylabel(’x(t)’)
title(’Continuous Time Signal’)
subplot(2,1,2);
a = gca();
a.y_location = ”origin”;
plot(W,XW);
xlabel(’Frequency in Radians/Seconds W’);
ylabel(’X(jW)’)
title(’Continuous-time Fourier Transform’)
```

Scilab code Exa 7.4 inverse fourier transform

```
//Example 4.5
// Inverse Continuous Time Fourier Transform
// impulse funtion
clear;
clc;
close;
// CTFT
A =1;     //Amplitude
Dw = 0.005;
W1 = 4;  //Time in seconds
w = -W1/2:Dw:W1/2;
  for i=1:length(w)
      XW(1)=1;
      end
```

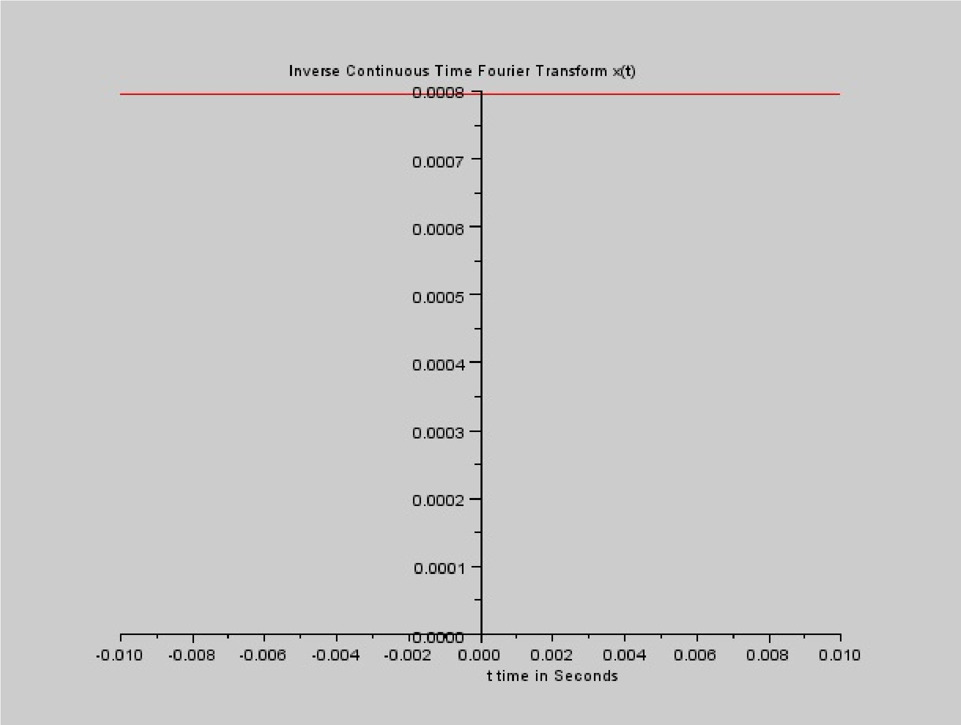

Figure 7.2: inverse fourier transform

```
XW = XW';

//Inverse Continuous-time Fourier Transform
t = -0.01:1/length(w):0.01;
xt =(1/(2*%pi))*XW *exp(sqrt(-1)*w'*t)*Dw;
xt = real(xt);
figure
a = gca();
a.y_location ="origin";
a.x_location ="origin";
plot(t,xt);
xlabel('                                        t time in Seconds');
title('Inverse Continuous Time Fourier Transform x(t)')
```

Scilab code Exa 7.5 inverse fourier transform

```
//signals and systems
// Inverse Continuous Time Fourier Transform
// shifted impulse function
clear;
clc;
close;
w0=1
A =1;     //Amplitude
Dw = 0.005;
W1 = 4;  //Time in seconds
w = -W1/2:Dw:W1/2;
XW=[zeros(1,length(w)/2) 1 zeros(1,length(w/2))];
XW = XW';

//Inverse Continuous-time Fourier Transform
t = -0.01:1/length(w):0.01;
size(XW)
```

```
size(t)
xt =(1/(2*%pi))*XW *exp(sqrt(-1)*w'.*t).*exp(sqrt(-1).*t)*Dw;
xt = real(xt);
figure
a = gca();
a.y_location ="origin";
a.x_location ="origin";
plot(t,xt);
xlabel('                                                  t time in Seconds');
title('Inverse Continuous Time Fourier Transform x(t)')
```

Scilab code Exa 7.6 fourier transform for everlasting sinusoid

```
//signals and systems
// Continuous Time Fourier Transforms
// Sinusoidal waveforms cos(Wot)
clear;
clc;
close;

T1 = 2;
T = 4*T1;
Wo = 2*%pi/T;
W = [-Wo,0,Wo];
ak = (2*%pi*Wo*T1/%pi)/sqrt(-1);
XW = [-ak,0,ak];
ak1 = (2*%pi*Wo*T1/%pi);
XW1 =[ak1,0,ak1];

figure
```

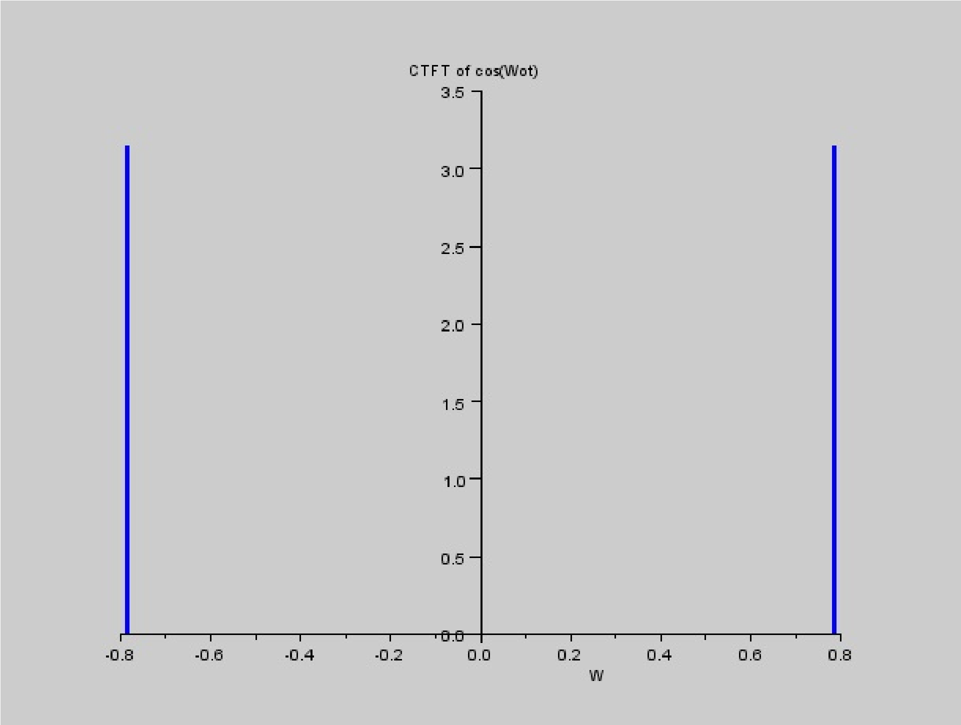

Figure 7.3: fourier transform for everlasting sinusoid

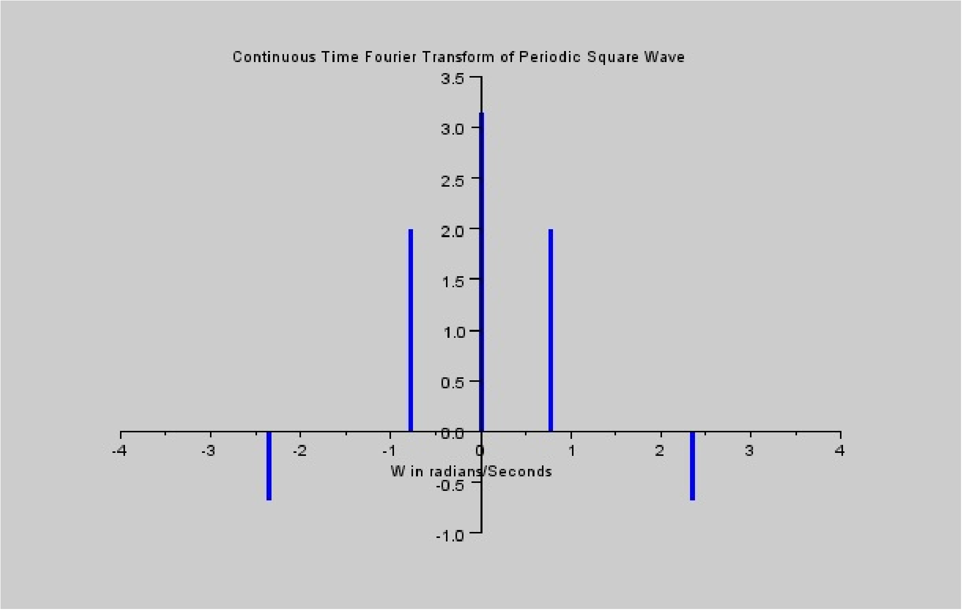

Figure 7.4: fourier transform of a periodic signal

```
a = gca();
a.y_location ="origin";
a.x_location ="origin";
plot2d3('gnn',W,XW1,2);
poly1 = a.children(1).children(1);
poly1.thickness = 3;
xlabel('

   W');
title('CTFT of cos(Wot)')
```

Scilab code Exa 7.7 fourier transform of a periodic signal

```
//signals and systems
```

```
// Continuous Time Fourier Transform of Symmetric
// periodic Square waveform
clear;
clc;
close;

T1 = 2;
T = 4*T1;
Wo = 2*%pi/T;
W = -%pi:Wo:%pi;
delta = ones(1,length(W));
XW(1) = (2*%pi*Wo*T1/%pi);
mid_value = ceil(length(W)/2);
for k = 2:mid_value
  XW(k) = (2*%pi*sin((k-1)*Wo*T1)/(%pi*(k-1)));
end
figure
a = gca();
a.y_location ="origin";
a.x_location ="origin";
plot2d3('gnn',W(mid_value:$),XW,2);
poly1 = a.children(1).children(1);
poly1.thickness = 3;
plot2d3('gnn',W(1:mid_value-1),XW($:-1:2),2);
poly1 = a.children(1).children(1);
poly1.thickness = 3;
xlabel('W in radians/Seconds');
title('Continuous Time Fourier Transform of Periodic Square Wave')
```

Scilab code Exa 7.8 fourier transform of a unit impulse train

```
//signals and systems
```

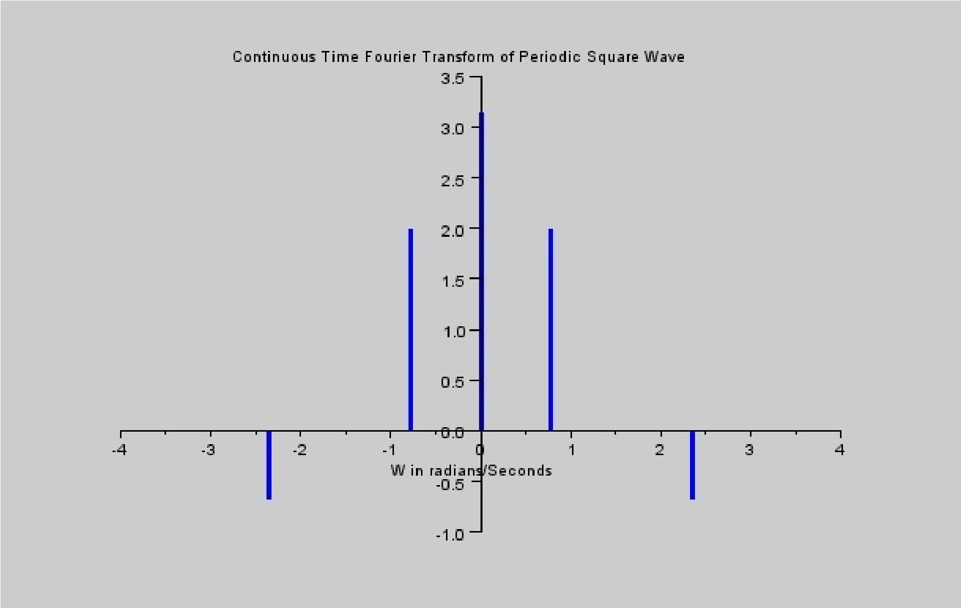

Figure 7.5: fourier transform of a unit impulse train

```
//continuous time signal analysis the fourier transform
// Periodic Impulse Train
clear;
clc;
close;
T = -4:4;;
T1 = 1; //Sampling Interval
xt = ones(1,length(T));
ak = 1/T1;
XW = 2*%pi*ak*ones(1,length(T));
Wo = 2*%pi/T1;
W = Wo*T;
figure
subplot(2,1,1)
a = gca();
a.y_location =”origin”;
a.x_location =”origin”;
```

```
plot2d3('gnn',T,xt,2);
poly1 = a.children(1).children(1);
poly1.thickness = 3;
xlabel('

   t');
title('Periodic Impulse Train')
subplot(2,1,2)
a = gca();
a.y_location ="origin";
a.x_location ="origin";
plot2d3('gnn',W,XW,2);
poly1 = a.children(1).children(1);
poly1.thickness = 3;
xlabel('

   t');
title('CTFT of Periodic Impulse Train')
```

Scilab code Exa 7.9 fourier transform of unit step function

```
//signals and systems
//continuous time signal analysis the fourier transform
//fourier transform of unit step function u(t)
clear;
clc;
A =0.000000001;     //Amplitude
Dt = 0.005;
t = 0:Dt:4.5;
xt = exp(-A*abs(t));
Wmax = 2*%pi*1;         //Analog Frequency = 1Hz
K = 4;
```

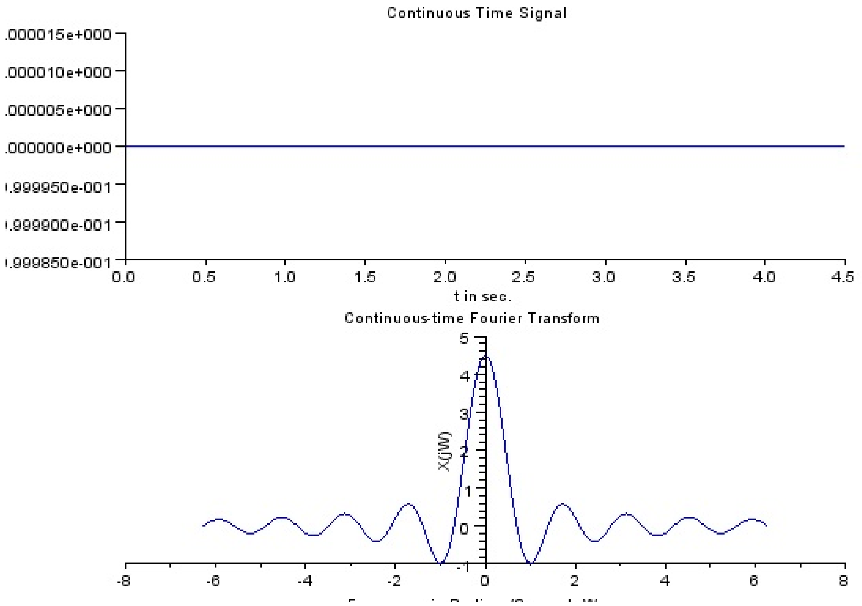

Figure 7.6: fourier transform of unit step function

```
k = 0:(K/500):K;
W = k*Wmax/K;
XW = xt* exp(-sqrt(-1)*t'*W) * Dt;
XW = real(XW);
W = [-mtlb_fliplr(W), W(2:501)]; // Omega from -Wmax to Wmax
XW = [mtlb_fliplr(XW), XW(2:501)];
subplot(2,1,1);
a = gca();
a.y_location = "origin";
plot(t,xt);
xlabel('t in sec.');
ylabel('x(t)')
title('Continuous Time Signal')
subplot(2,1,2);
a = gca();
a.y_location = "origin";
plot(W,XW);
```

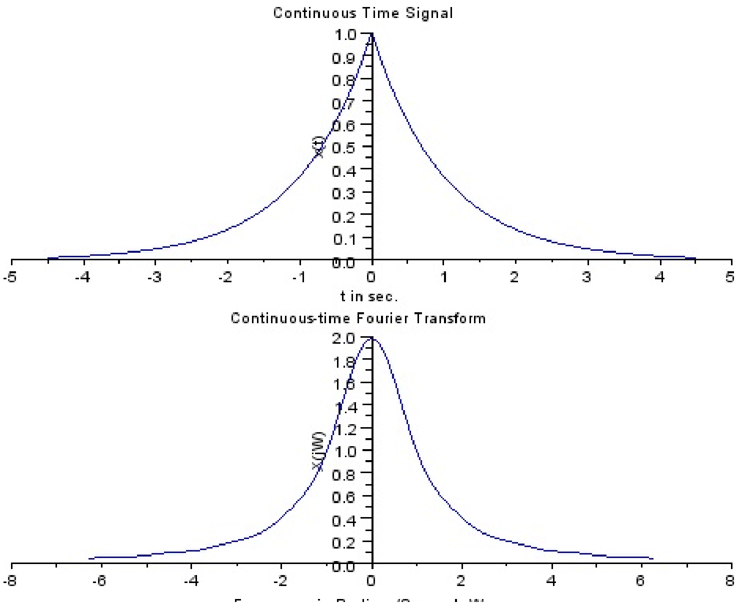

Figure 7.7: fourier transform of exponential function

```
xlabel('Frequency in Radians/Seconds W');
ylabel('X(jW)')
title('Continuous-time Fourier Transform')
```

Scilab code Exa 7.12 fourier transform of exponential function

```
//signals and systems
//Continuous Time Fourier Transform
//Continuous Time Signal x(t)= exp(-A*abs(t))
clear;
clc;
close;

A =1;      //Amplitude
```

```
Dt = 0.005;
t = -4.5:Dt:4.5;
xt = exp(-A*abs(t));

Wmax = 2*%pi*1;         //Analog Frequency = 1Hz
K = 4;
k = 0:(K/1000):K;
W = k*Wmax/K;
XW = xt* exp(-sqrt(-1)*t'*W) * Dt;
XW = real(XW);
W = [-mtlb_fliplr(W), W(2:1001)]; // Omega from -Wmax to Wmax
XW = [mtlb_fliplr(XW), XW(2:1001)];
subplot(1,1,1)
subplot(2,1,1);
a = gca();
a.y_location = "origin";
plot(t,xt);
xlabel('t in sec.');
ylabel('x(t)')
title('Continuous Time Signal')
subplot(2,1,2);
a = gca();
a.y_location = "origin";
plot(W,XW);
xlabel('Frequency in Radians/Seconds W');
ylabel('X(jW)')
title('Continuous-time Fourier Transform')
```

Chapter 8

Sampling The bridge from continuous to discrete

Scilab code Exa 8.8 discrete fourier transform

```
//signals and systems
//sampling:the bridge from continuous to discrete
//DFT to compute the fourier transform of e^-2t.u(t)
T_0 = 4;
N_0 = 256;
T = T_0/N_0;
t = (0:T:T*(N_0-1))';
x = T*exp(-2*t);
x = mtlb_i(x,1,(T*(exp(-2*T_0)+1))/2);
X_r = fft(x);
r = (-N_0/2:N_0/2-1)';
omega_r = ((r*2)*%pi)/T_0;
omega = linspace(-%pi/T,%pi/T,4097);
X = 1 ./(%i*omega+2);
subplot(2,1,1);
a = gca();
a.y_location ="origin";
```

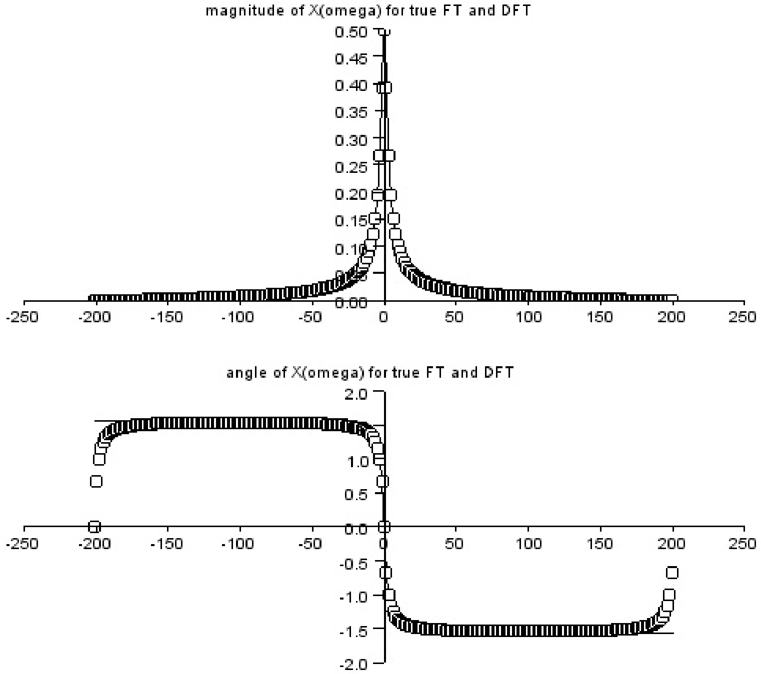

Figure 8.1: discrete fourier transform

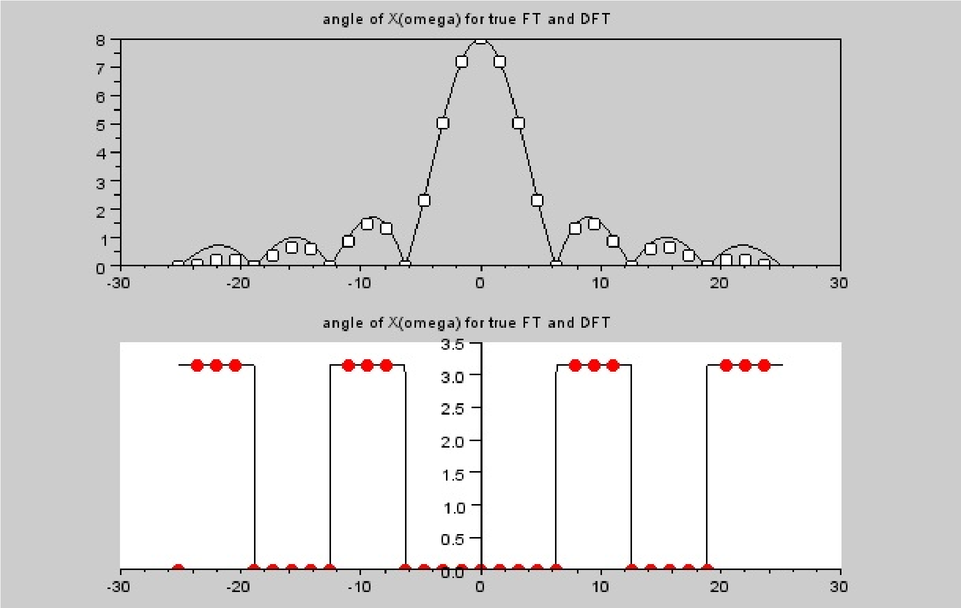

Figure 8.2: discrete fourier transform

```
a.x_location =”origin”;
plot(omega,abs(X),”k”,omega_r,fftshift(abs(X_r)),”ko
    ”);
xtitle(”magnitude of X(omega) for true FT and DFT”);
subplot(2,1,2);
a = gca();
a.y_location =”origin”;
a.x_location =”origin”;
plot(omega,atan(imag(X),real(X)),”k”,omega_r,
    fftshift(atan(imag(X_r),real(X_r))),”ko”);
xtitle(”angle of X(omega) for true FT and DFT”);
```

Scilab code Exa 8.9 discrete fourier transform

```
//signals and systems
//sampling:the bridge from continuous to discrete
//DFT to compute the fourier transform of 8rect(t)
T_0 = 4;
N_0 = 32;
T = T_0/N_0;
x_n = [ones(1,4) 0.5 zeros(1,23) 0.5 ones(1,3)]';
size(x_n)
x_r = fft(x_n);r = (-N_0/2:(N_0/2)-1)';
omega_r = ((r*2)*%pi)/T_0;
size(omega_r)
size(omega)
omega = linspace(-%pi/T,%pi/T,4097);
X = 8*(sinc(omega/2));
size(X)
figure(1);
subplot(2,1,1);
plot(omega,abs(X),"k");
plot(omega_r,fftshift(abs(x_r)),"ko")
xtitle("angle of X(omega) for true FT and DFT");
a=gca();
subplot(2,1,2);
a = gca();
a.y_location ="origin";
a.x_location ="origin";
plot(omega,atan(imag(X),real(X)),"k",omega_r,
    fftshift(atan(imag(x_r),real(x_r))),'r.');
xtitle("angle of X(omega) for true FT and DFT");
```

Scilab code Exa 8.10 frequency response of a low pass filter

```
//signals and systems
// sampling: the bridge between continuous to
```

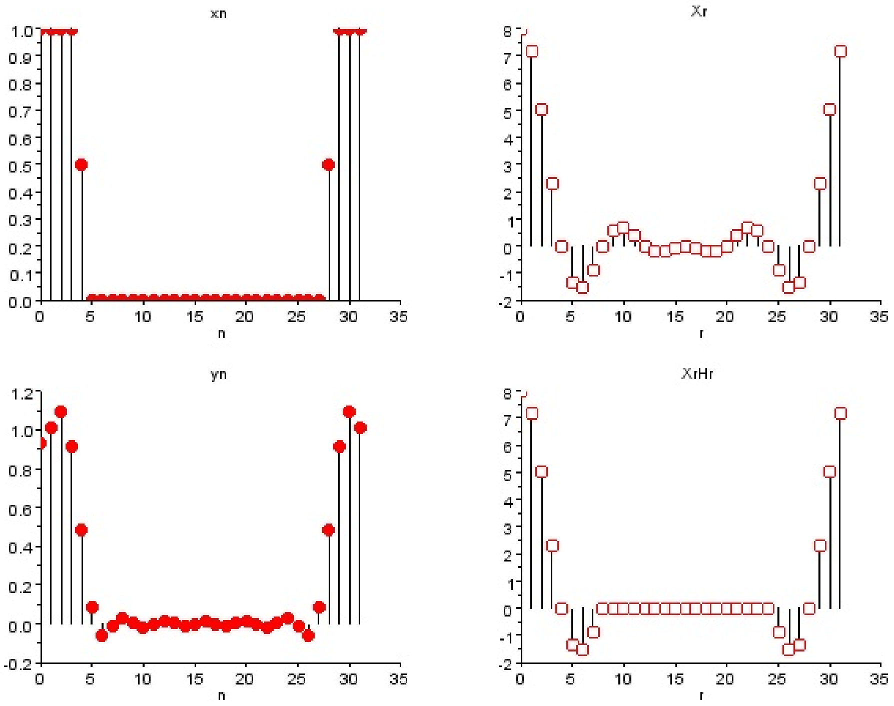

Figure 8.3: frequency response of a low pass filter

```
    discrete
T_0 = 4;
N_0 = 32;
T = T_0/N_0;n = 0:N_0-1;r = n;
x_n = [ones(1,4),0.5,zeros(1,23),0.5,ones(1,3)]';
H_r = [ones(1,8),0.5,zeros(1,15),0.5,ones(1,7)]';
X_r = fft(x_n,-1);
Y_r = H_r .*(X_r);y_n = mtlb_ifft(Y_r);
subplot(2,2,1);
plot2d3(n,x_n);
plot(n,x_n,'r.')
xtitle('xn','n')
subplot(2,2,2);
plot2d3(r,real(X_r));
plot(r,real(X_r),'ro')
xtitle('Xr','r')
subplot(2,2,3);
plot2d3(n,real(y_n));
plot(n,real(y_n),'r.')
xtitle('yn','n')
subplot(2,2,4);
plot2d3(r,(X_r).*H_r);
plot(r,(X_r).*H_r,'ro')
xtitle('XrHr','r')
```

Chapter 9

fourier analysis of discrete time signals

Scilab code Exa 9.1 discrete time fourier series

```
//signals and systems
//fourier analysis of discrete time signals
//Example5.5:Discrete Time Fourier Transform:x[n]=
   sin(nWo)
clear;
clc;
close;
N = 0.1;
Wo = %pi;
W = [-Wo/10,0,Wo/10];
XW =[0.5,0,0.5];
//
figure
a = gca();
a.y_location =”origin”;
a.x_location =”origin”;
plot2d3('gnn',W,XW,2);
```

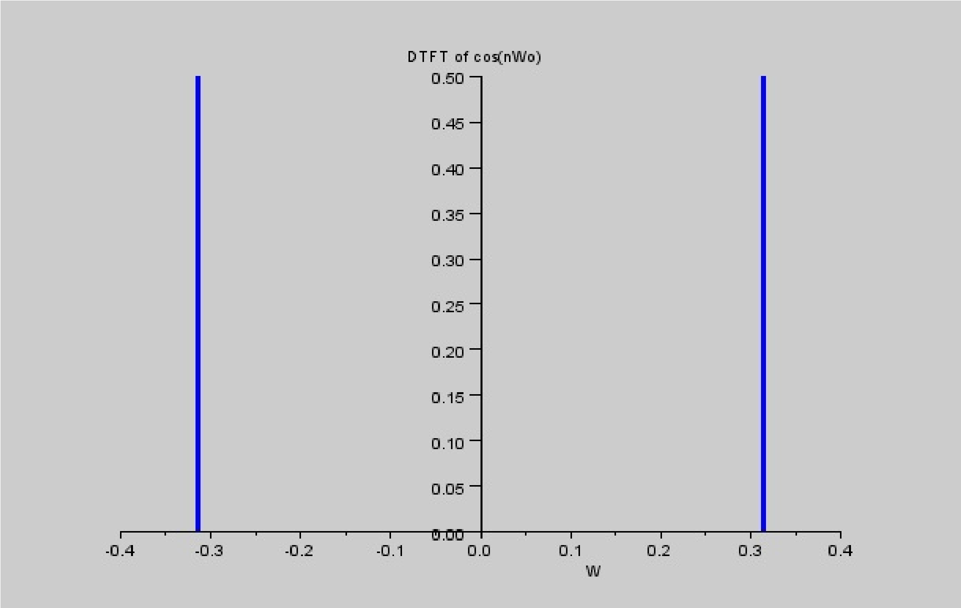

Figure 9.1: discrete time fourier series

```
poly1 = a.children(1).children(1);
poly1.thickness = 3;
xlabel('

   W');
title('DTFT of cos(nWo)')
disp(Wo/10)
```

Scilab code Exa 9.2 DTFT for periodic sampled gate function

```
N_0=32; n=(0:N_0-1);
x_n= [ones(1,5) zeros(1,23) ones(1,4)];
for r=0:31
    X_r(r+1)=sum(x_n.*exp(-sqrt(-1)*r*2*3.14/N_0*n))
```

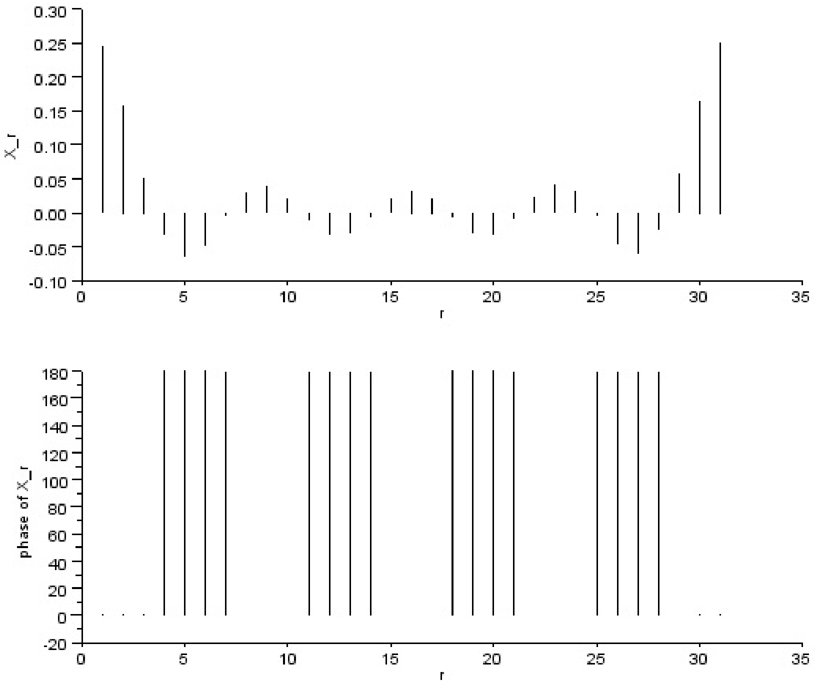

Figure 9.2: DTFT for periodic sampled gate function

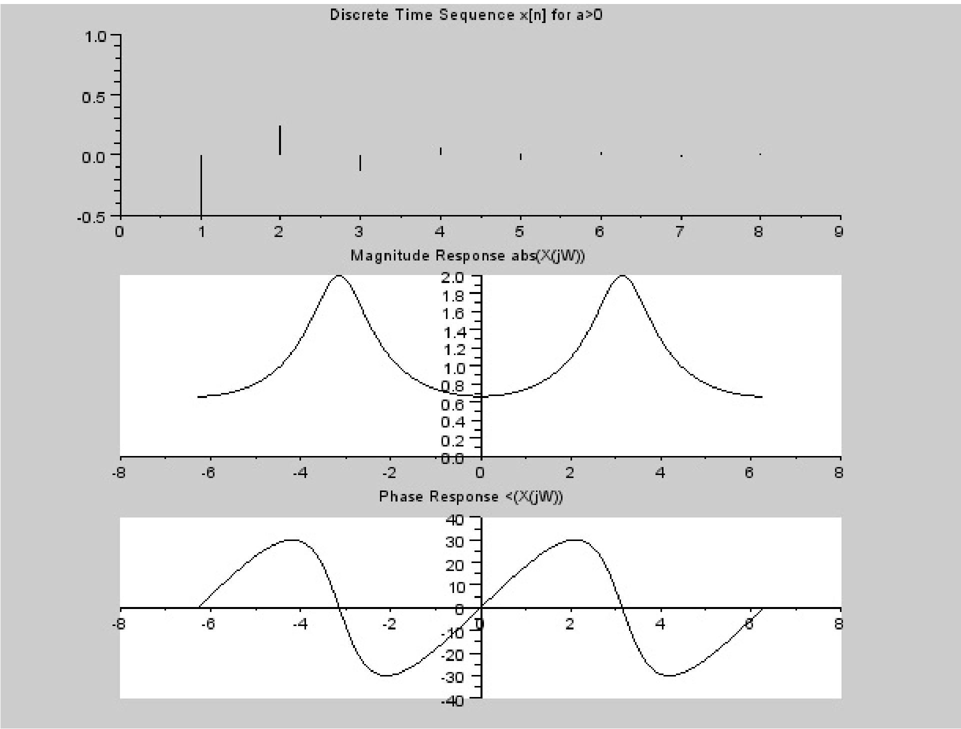

Figure 9.3: discrete time fourier series

```
      /32;
end
subplot(2,1,1); r=n; plot2d3(r,real(X_r));
xlabel('r'); ylabel('X_r');
X_r=fft(x_n)/N_0;
subplot(2,1,2);
plot2d3(r,phasemag(X_r));
xlabel('r'); ylabel('phase of X_r');
disp(N_0,'period=')
disp(2*%pi/N_0,'omega=')
```

Scilab code Exa 9.3 discrete time fourier series

```
//signals and systems
//Discrete Time Fourier Transform of discrete sequence
//x[n]= (a^n).u[n], a>0 and a<0
clear;
clc;
close;
// DTS Signal
a1 = 0.5;
a2 = -0.5;
max_limit = 10;
for n = 0:max_limit-1
  x1(n+1) = (a1^n);
  x2(n+1) = (a2^n);
end
n = 0:max_limit-1;
// Discrete-time Fourier Transform
Wmax = 2*%pi;
K = 4;
k = 0:(K/1000):K;
W = k*Wmax/K;
x1 = x1';
x2 = x2';
XW1 = x1* exp(-sqrt(-1)*n'*W);
XW2 = x2* exp(-sqrt(-1)*n'*W);
XW1_Mag = abs(XW1);
XW2_Mag = abs(XW2);
W = [-mtlb_fliplr(W), W(2:1001)]; // Omega from -Wmax to Wmax
XW1_Mag = [mtlb_fliplr(XW1_Mag), XW1_Mag(2:1001)];
XW2_Mag = [mtlb_fliplr(XW2_Mag), XW2_Mag(2:1001)];
[XW1_Phase,db] = phasemag(XW1);
[XW2_Phase,db] = phasemag(XW2);
XW1_Phase = [-mtlb_fliplr(XW1_Phase),XW1_Phase(2:1001)];
XW2_Phase = [-mtlb_fliplr(XW2_Phase),XW2_Phase
```

```
   (2:1001)];
//plot for a>0
figure
subplot(3,1,1);
plot2d3(’gnn’,n,x1);
xtitle(’Discrete Time Sequence x[n] for a>0’)
subplot(3,1,2);
a = gca();
a.y_location =”origin”;
a.x_location =”origin”;
plot2d(W,XW1_Mag);
title(’Magnitude Response abs(X(jW))’)
subplot(3,1,3);
a = gca();
a.y_location =”origin”;
a.x_location =”origin”;
plot2d(W,XW1_Phase);
title(’Phase Response <(X(jW))’)
//plot for a<0
figure
subplot(3,1,1);
plot2d3(’gnn’,n,x2);
xtitle(’Discrete Time Sequence x[n] for a>0’)
subplot(3,1,2);
a = gca();
a.y_location =”origin”;
a.x_location =”origin”;
plot2d(W,XW2_Mag);
title(’Magnitude Response abs(X(jW))’)
subplot(3,1,3);
a = gca();
a.y_location =”origin”;
a.x_location =”origin”;
plot2d(W,XW2_Phase);
title(’Phase Response <(X(jW))’)
```

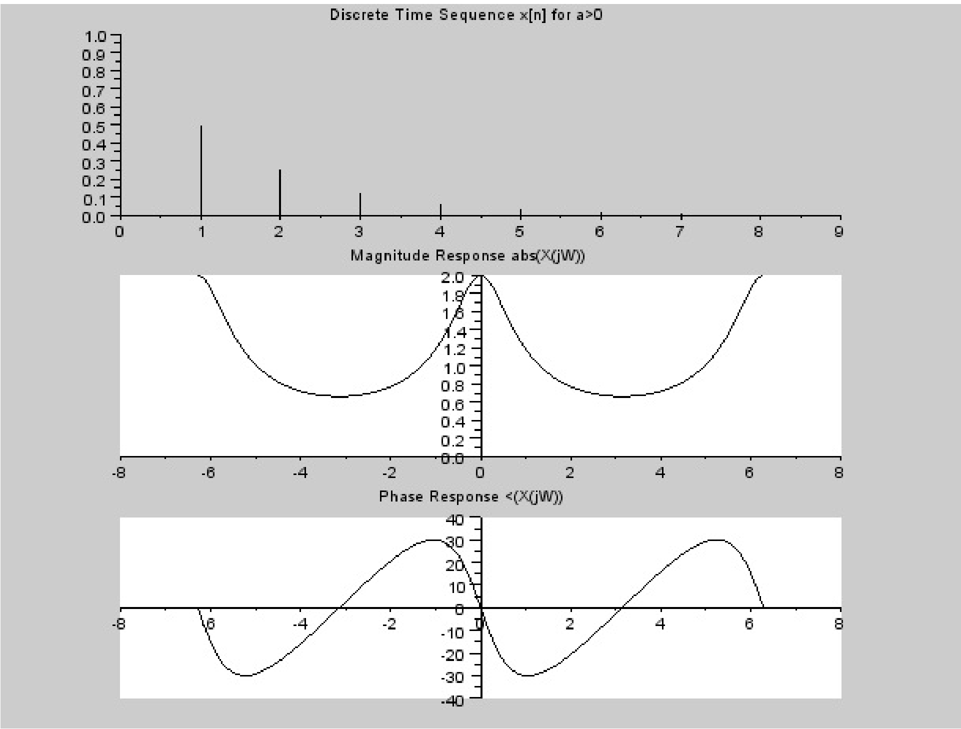

Figure 9.4: discrete time fourier series

Scilab code Exa 9.4 discrete time fourier series

```
//signals and systems
//Discrete Time Fourier Transform of discrete sequence
//x[n]= (a^n).u[-n], a>0 and a<0
clear;
clc;
```

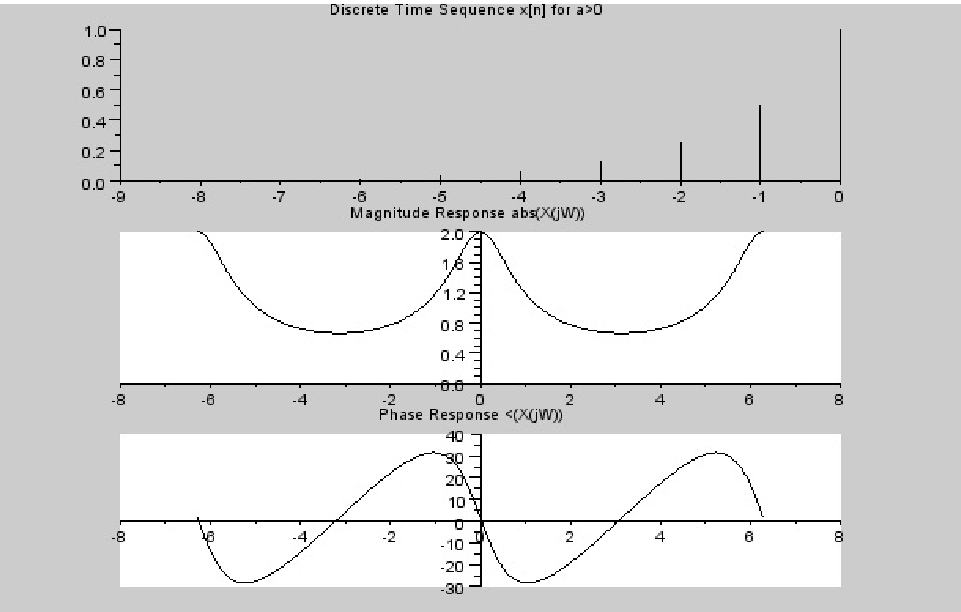

Figure 9.5: discrete time fourier series

```
close;
// DTS Signal
a = 0.5;
max_limit = 10;
for n = 0:max_limit-1
  x1(n+1) = (a^n);
end
n = 0:max_limit-1;
// Discrete-time Fourier Transform
Wmax = 2*%pi;
K = 4;
k = 0:(K/1000):K;
W = k*Wmax/K;
x1 = x1';
XW1 = x1* exp(-sqrt(-1)*n'*W);

XW1_Mag = abs(XW1);
W = [-mtlb_fliplr(W), W(2:1001)]; // Omega from -
```

```
   Wmax to Wmax
XW1_Mag = [mtlb_fliplr(XW1_Mag), XW1_Mag(2:1001)];
[XW1_Phase,db] = phasemag(XW1);
XW1_Phase = [-mtlb_fliplr(XW1_Phase),XW1_Phase(2:1001)];
//plot for a>0
figure
subplot(3,1,1);
plot2d3('gnn',-n,x1);
xtitle('Discrete Time Sequence x[n] for a>0')
subplot(3,1,2);
a = gca();
a.y_location ="origin";
a.x_location ="origin";
plot2d(W,XW1_Mag);
title('Magnitude Response abs(X(jW))')
subplot(3,1,3);
a = gca();
a.y_location ="origin";
a.x_location ="origin";
plot2d(W,XW1_Phase+%pi/2);
title('Phase Response <(X(jW))')
```

Scilab code Exa 9.5 DTFT for rectangular pulse

```
//signals and systems
//Discrete Time Fourier Transform
//x[n]= 1 , abs(n)<=N1
clear;
clc;
close;
// DTS Signal
N1 = 2;
```

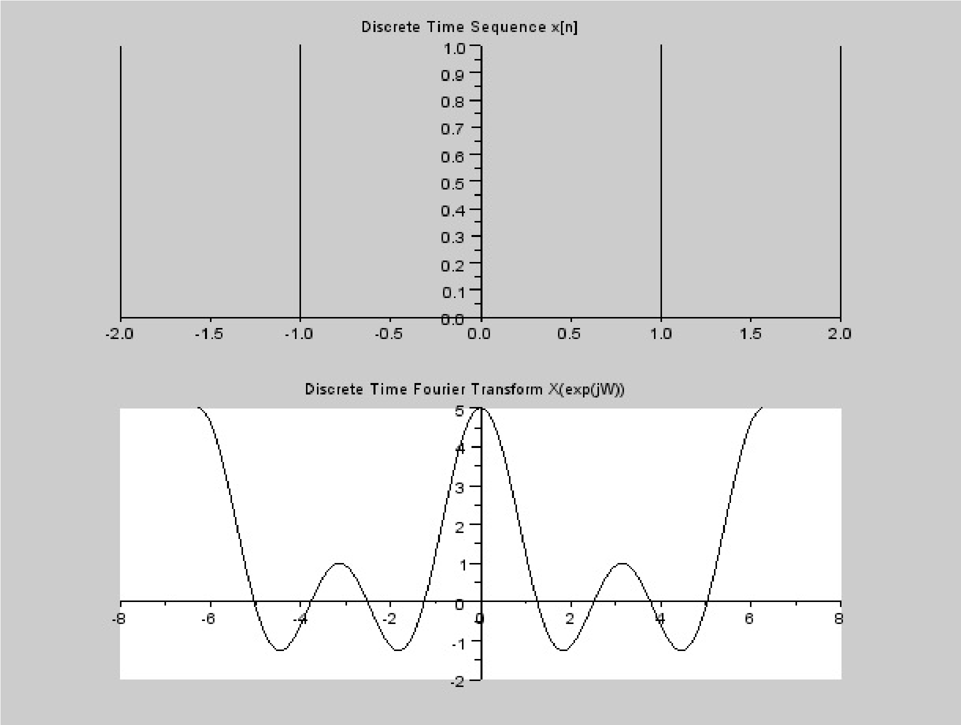

Figure 9.6: DTFT for rectangular pulse

```
n = -N1:N1;
x = ones(1,length(n));
// Discrete-time Fourier Transform
Wmax = 2*%pi;
K = 4;
k = 0:(K/1000):K;
W = k*Wmax/K;
XW = x* exp(-sqrt(-1)*n'*W);
XW_Mag = real(XW);
W = [-mtlb_fliplr(W), W(2:1001)]; // Omega from -Wmax to Wmax
XW_Mag = [mtlb_fliplr(XW_Mag), XW_Mag(2:1001)];
//plot for abs(a)<1
figure
subplot(2,1,1);
a = gca();
a.y_location ="origin";
a.x_location ="origin";
plot2d3('gnn',n,x);
xtitle('Discrete Time Sequence x[n]')
subplot(2,1,2);
a = gca();
a.y_location ="origin";
a.x_location ="origin";
plot2d(W,XW_Mag);
title('Discrete Time Fourier Transform X(exp(jW))')
```

Scilab code Exa 9.6 DTFT for rectangular pulse spectrum

```
//signals and systems
//discreet time fourier series
//IDTFT:Impulse Response of Ideal Low pass Filter
clear;
```

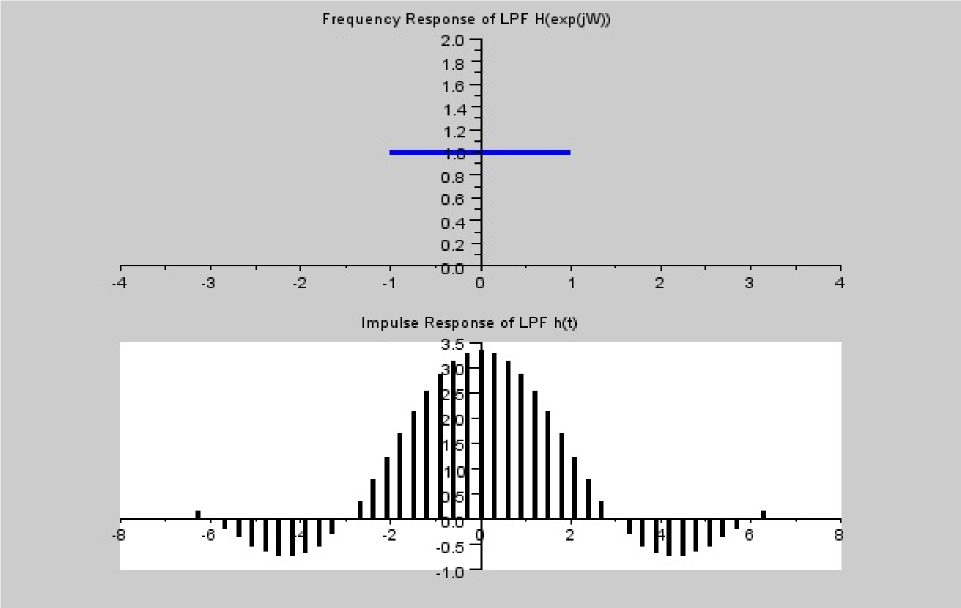

Figure 9.7: DTFT for rectangular pulse spectrum

```
clc;
close;
Wc = 1;     //1 rad/sec
W  = -Wc:0.1:Wc; //Passband of filter
H0 = 1; //Magnitude of Filter
HlpW = H0*ones(1,length(W));
//Inverse Discrete-time Fourier Transform
t = -2*%pi:2*%pi/length(W):2*%pi;
ht =(1/(2*%pi))*HlpW *exp(sqrt(-1)*W’*t);
ht = real(ht);
figure
subplot(2,1,1)
a = gca();
a.y_location =”origin”;
a.x_location =”origin”;
a.data_bounds=[-%pi,0;%pi,2];
plot2d(W,HlpW,2);
poly1 = a.children(1).children(1);
```

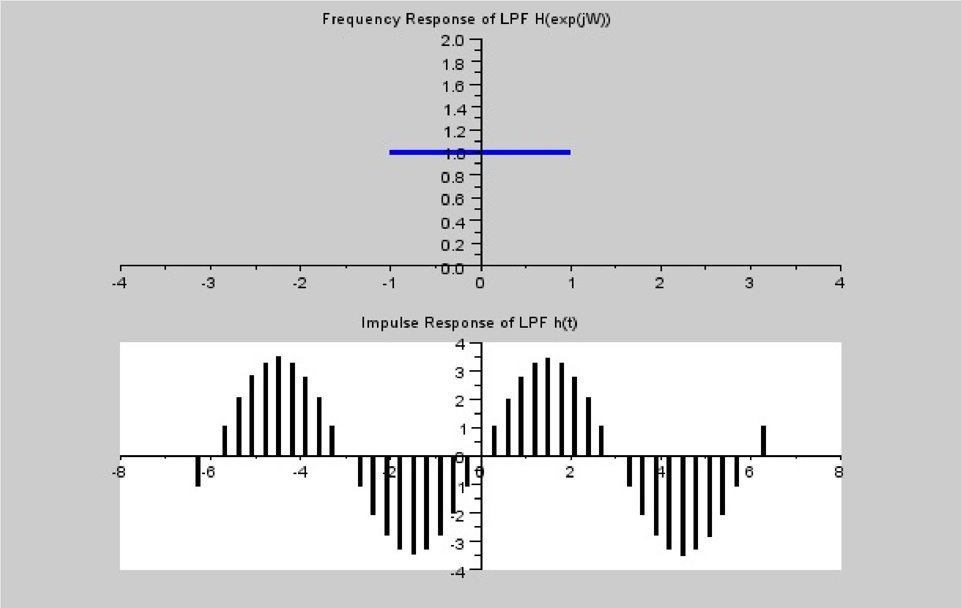

Figure 9.8: DTFT of sinc function

```
poly1.thickness = 3;
xtitle(’Frequency Response of LPF H(exp(jW))’)
subplot(2,1,2)
a = gca();
a.y_location =”origin”;
a.x_location =”origin”;
a.data_bounds=[-2*%pi,-1;2*%pi,2];
plot2d3(’gnn’,t,ht);
poly1 = a.children(1).children(1);
poly1.thickness = 3;
xtitle(’Impulse Response of LPF h(t)’)
```

Scilab code Exa 9.9 DTFT of sinc function

```
//signals and systems
//discreet time fourier series
//IDTFT:Impulse Response of Ideal Low pass Filter
clear;
clc;
close;
Wc = 1;    //1 rad/sec
W  = -Wc:0.1:Wc; //Passband of filter
H0 = 1; //Magnitude of Filter
HlpW = H0*ones(1,length(W));
//Inverse Discrete-time Fourier Transform
t = -2*%pi:2*%pi/length(W):2*%pi;
ht1 =(1/(2*%pi))*HlpW *exp(sqrt(-1)*W'*t);
size(ht1)
n=-21:21;
size(n)
ht=ht1.*(%e^%i*2*t);
ht = real(ht);
figure
subplot(2,1,1)
a = gca();
a.y_location ="origin";
a.x_location ="origin";
a.data_bounds=[-%pi,0;%pi,2];
plot2d(W,HlpW,2);
poly1 = a.children(1).children(1);
poly1.thickness = 3;
xtitle('Frequency Response of LPF H(exp(jW))')
subplot(2,1,2)
a = gca();
a.y_location ="origin";
a.x_location ="origin";
a.data_bounds=[-2*%pi,-1;2*%pi,2];
size(t)
size(ht)
plot2d3('gnn',t,ht);
poly1 = a.children(1).children(1);
poly1.thickness = 3;
```

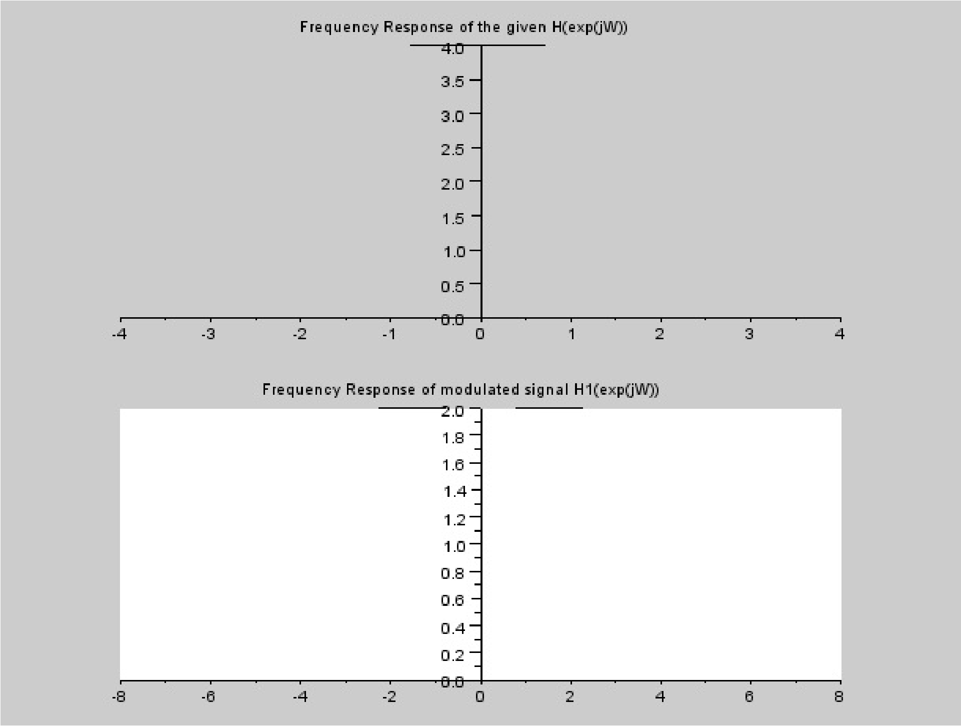

Figure 9.9: sketching the spectrum for a modulated signal

```
xtitle('Impulse Response of LPF h(t)')
```

Scilab code Exa 9.10.a sketching the spectrum for a modulated signal

```
//signals and systems
//discrete fourier transform
//Frequency Shifting Property of DTFT
clear;
clc;
close;
mag = 4;
```

```
W  = -%pi/4:0.1:%pi/4;
H1 = mag*ones(1,length(W));
W1 =W+%pi/2;
W2 = -W-%pi/2;
figure
subplot(2,1,1)
a = gca();
a.y_location =”origin”;
a.x_location =”origin”;
a.data_bounds=[-%pi,0;%pi,2];
plot2d(W,H1);
xtitle('Frequency Response of the given H(exp(jW))')
subplot(2,1,2)
a = gca();
a.y_location =”origin”;
a.x_location =”origin”;
a.data_bounds=[-2*%pi,0;2*%pi,2];
plot2d(W1,0.5*H1);
plot2d(W2,0.5*H1);
xtitle('Frequency Response of modulated signal H1(exp(jW))')
```

Scilab code Exa 9.13 frequency response of LTID

```
//LTi Systems characterized by Linear Constant
//fourier analysis of discrete systems
//Inverse Z Transform
//z = %z;
syms n z;
H1 = (-5/3)/(z-0.5);
H2 = (8/3)/(z-0.8);
F1 = H1*z^(n)*(z-0.5);
F2 = H2*z^(n)*(z-0.8);
h1 = limit(F1,z,0.5);
disp(h1,'h1[n]=')
```

```
h2 = limit(F2,z,0.8);
disp(h2,'h2[n]=')
h = h1-h2;
disp(h,'h[n]=')
```

Chapter 10

state space analysis

Scilab code Exa 10.4 state space descrption by transfer function

```
//signals and systems
//state space analysis
//state space description
clear all;
close;
clc;
s=poly(0,'s');
H=[(4/3)/(1+s),-2/(3+s), (2/3)/(4+s)];
Sys=tf2ss(H)
clean(ss2tf(Sys))
disp(Sys)
```

Scilab code Exa 10.5 finding the state vector

```
syms t s
A=[-12 2/3;-36 -1]; B=[1/3;1]; q0=[2;1]; X=1/s;
size(A)
size(s*eye(2,2))
```

```
Q=inv(s*eye(2,2)-A)*(q0+B*X);
q=[];
q(1)=ilaplace(Q(1));
q(2)=ilaplace(Q(2));
disp(q*’u(t)’,”[q1(t) ; q2(t)]”)
```

Scilab code Exa 10.6 state space descrption by transfer function

```
A=[0 1;-2 -3];
B=[1 0;1 1];
C=[1 0;1 1;0 2];
D=[0 0;1 0; 0 1];
syms s;
H=C*inv(s*eye(2,2)-A)*B+D;
disp(H,”the transfer function matrix H(s)=”)
disp(H(3,2),”the transfer function relating y3 and x2 is H32(s)=”)
```

Scilab code Exa 10.7 time domain method

```
//signals and systems
//state space
//time domain method to find the state vector
clc;
clf;
s=poly(0,’s’);
A=[s+12 -2/3; 36 s+1];
y=roots(det(A))
t=poly(0,’t’);
beta=inv([1 y(1); 1 y(2)])*[%e^-y(1)*t; %e^-y(2)*t];
disp(beta)
size(beta)
W=beta(1)*[1 0;0 1]+ beta(2)*[-12 2/3;-36 -1];
```

```
zir=W*[2;1];
disp(zir);
zsr=W*[1/3;1];
disp(zsr);
total=zir+zsr;
disp(total);
```

Scilab code Exa 10.8 state space descrption by transfer function

```
syms t s;
F1=ilaplace((s+3)/((s+1)*(s+2)))
F2=ilaplace(1/((s+1)*(s+2)))
F3=ilaplace(-2/((s+1)*(s+2)))
F4=ilaplace(s/((s+1)*(s+2)))
F=[F1 F2;F3 F4];
disp(F,"f(t)=")
A=[1 0;1 1;0 2];
B=[0 0;1 0;0 1];
h=A*F*[1 0;1 1]+B*eye(2,2);//here 1 represents del(t
    )
disp(h,"h(t)=")
```

Scilab code Exa 10.9 state equations of a given systems

```
A=[0 1;-2 -3];
B=[1;2];
P=[1 1;1 -1];
Ahat= P*A*inv(P)
Bhat=P*B
disp(Ahat,"A^=")
disp(Bhat,"B^=")
```

Scilab code Exa 10.10 diagonalized form of state equation

```
A=[0 1;-2 -3];
[V,lambda]=spec(A);
B=[1;2];
Bhat=P*B
disp(P,"P=")
disp(Bhat,"Bˆ=")
disp(lambda,"lambda=")
```

Scilab code Exa 10.11 controllability and observability

```
A=[1 0;1 -1];
[V,lambda]=spec(A);
B=[1;0];
C=[1 -2];
P=inv(V);
Bhat=P*B
Chat=C*inv(P)
disp(' (a):')
disp(Bhat,"Bˆ=")
disp(Chat,"Cˆ=")
A=[-1 0;-2 1];
[V,lambda]=spec(A);
B=[1;1];
C=[0 1];
P=inv(V);
Bhat=P*B
Chat=C*inv(P)
disp('Part (b):')
disp(Bhat,"Bˆ=")
disp(Chat,"cˆ=")
```

Scilab code Exa 10.12 state space description of a given description

```
A=[0 1;-1/6 5/6];
B=[0;1];
C=[-1 5];
D=0;
sys=syslin('d',A,B,C,D);
N=25;
x=ones(1,N+1);n=(0:N);
q0=[2;3];
[ y q]=csim('step',n,sys);
y=dsimul(sys,x);
plot2d3(y)
```

Scilab code Exa 10.13 total response using z transform

```
//LTi Systems characterized by Linear Constant
//Inverse Z Transform
//z = %z;
syms n z;
H1 = (-2*z)/(z-(1/3));
H2 = (3*z)/(z-0.5);
H3 = (24*z)/(z-1);
F1 = H1*z^(n-1)*(z-(1/3));
F2 = H2*z^(n-1)*(z-0.5);
F3 = H3*z^(n-1)*(z-1);
h1 = limit(F1,z,(1/3));
disp(h1,'h1[n]=')
h2 = limit(F2,z,0.5);
disp(h2,'h2[n]=')
h3 = limit(F3,z,1);
```

```
disp(h3,'h3[n]=')
h = h1+h2+h3;
disp(h,'h[n]=')
```