**Experiment No. 02**

**Aim**: Generation of various signals and sequences (Periodic and Aperiodic)

S**oftware used:** MATLAB.

  MATLAB® is a programming language and numerical computing environment. The name MATLAB® is an acronym for “Matrix Laboratory”. As it names suggests it allows easy manipulation of matrix and vectors. Plotting functions and data is made easy with MATLAB®. It has a good Graphic User Interface and conversion of MATLAB files to C/C++ is possible. It has several toolboxes that possess specific functions for specific applications. For example, Image Processing, Neural Networks, CDMA toolboxes are name a few. An additional package, Simulink, adds graphical multidomain simulation and Model‐Based Design for dynamic and embedded systems. Simulink contains Block sets that is analogous to Toolboxes. It was created by MathWorks Incorporation, USA.    Writing MATLAB programs for modulation applications require knowledge on very few functions and operators. The operators mostly used are arithmetic operators and matrix operators. To know more type in the command prompt ‘help ops’. MATLAB will give a list in that to know on specific operator say addition type in the command prompt ‘help plus’. MATLAB will give how to use and other relevant information.

Commonly used graphical functions are plot, figure, subplot, title, and mathematical functions are sin and cos only.  The mathematical functions sin and cos are self‐explanatory. The graphical function figure will create a new window and then subsequent graphical commands can be applied. The plot function usually takes two vectors and plot data points according to given vector data.  Subplot function is used when two or more plots are drawn on the same figure. As title function suggests it helps to write title of the graph in the figure. For further details type ‘help plot’ or ‘help subplot’ in the command prompt and learn the syntax.

**THEORY:**

1. **Unit Impulse: for t=0, impulse is 1 otherwise 0.**
2. **Unit Step: for t>=0, step is 1 otherwise 0.**
3. **Unit Ramp: for t>=0, ramp is function of t otherwise 0.**
4. **Parabola: for t>=0, parabola is a function of t, x(t) = t2/2, otherwise 0**
5. **Rectangle: for |t|>=1/2, rectangle is 1 otherwise 0.**
6. **Triangular: for |t|<=2, triangular is a function of t, x(t) = 1-|t|/2 otherwise 0.**
7. **Signun: for t>=0, signun is 1, otherwise 0.**
8. **Sinc, Sawtooth, Sine, Cosine,Exponential functions are inbuilt.**

**MATLAB CODE**:

Continuous Time Signal [CTS]:

close all;

clear all;

%time range

t=-5:0.1:5;

%Unit Impulse Signal

impulse=1.\*(t==0)+0.\*(t~=0);

subplot(4,3,1);

plot(t,impulse)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Impulse(4D4)");

axis([min(t),max(t), min(impulse)-0.5,max(impulse)+0.5]);

%Unit Step Signal

step=1.\*(t>=0)+0.\*(t<0);

subplot(4,3,2);

plot(t,step)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Step(4D4)");

axis([min(t),max(t), min(step)-0.5,max(step)+0.5]);

%Unit Ramp Signal

ramp=t.\*(t>=0)+0.\*(t<0);

subplot(4,3,3);

plot(t,ramp)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Ramp(4D4)");

axis([min(t),max(t), min(ramp)-0.5,max(ramp)+0.5]);

%Unit Parabolic Signal

parabolic=t.^2/2.\*(t>=0)+0.\*(t<0);

subplot(4,3,4);

plot(t,parabolic)

xlabel("Iime");

ylabel("amplitude");

title("Unit Parabolic(4D4)");

axis([min(t),max(t), min(parabolic)-0.5,max(parabolic)+0.5]);

%Unit Rectangular Signal

rectangle=1.\*(abs(t)<=0.5)+0.\*(abs(t)>0.5);

subplot(4,3,5);

plot(t,rectangle)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Rectangle(4D4)");

axis([min(t),max(t), min(rectangle)-0.5,max(rectangle)+0.5]);

%Unit Triangular Signal

triangular=(1-(abs(t)/2)).\*(abs(t)<=2)+0.\*(abs(t)>2);

subplot(4,3,6);

plot(t,triangular)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Triangular(4D4)");

axis([min(t),max(t), min(triangular)-0.5,max(triangular)+0.5]);

%Unit Signum Signal

signum=1.\*(t>=0)-1.\*(t<0);

subplot(4,3,7);

plot(t,signum)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Signum(4D4)");

axis([min(t),max(t), min(signum)-0.5,max(signum)+0.5]);

%Unit Sinc Signal

u\_sinc=sinc(t);

subplot(4,3,8);

plot(t,u\_sinc)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Sinc(4D4)");

axis([min(t),max(t), min(u\_sinc)-0.5,max(u\_sinc)+0.5]);

%Unit Real Exponential Signal

real\_exponential=exp(5.\*t).\*(t>=0)+0.\*(t<0);

subplot(4,3,9);

plot(t,real\_exponential)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Real Exponential(4D4)");

axis([min(t),max(t), min(real\_exponential)-0.5,max(real\_exponential)+0.5]);

%time interval for sin and cos Signals

t1=0:pi/12:2\*pi;

%Unit Sin Signal

u\_sin=sin(t1);

subplot(4,3,10);

plot(t1,u\_sin)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Sinusoidal(4D4)");

axis([min(t1),max(t1), min(u\_sin)-0.5,max(u\_sin)+0.5]);

%Unit Cosine Signal

u\_cos=cos(t1);

subplot(4,3,11);

plot(t1,u\_cos)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Cosinusoidal(4D4)");

axis([min(t1),max(t1), min(u\_cos)-0.5,max(u\_cos)+0.5]);

%time interval for saw tooth Signal

F=input('enter the frequency:');

T=1/F;

t=0:5\*T/100:5\*T;

%Umit Sawtooth Signal

u\_saw\_tooth=sawtooth(2\*pi\*F\*t,1/2);

subplot(4,3,12);

plot(t,u\_saw\_tooth)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Saw Tooth(4D4)");

axis([min(t),max(t), min(u\_saw\_tooth)-0.5,max(u\_saw\_tooth)+0.5]);

Discrete Time Signal [DTS]:

close all;

clear all;

%time range

t=-5:1:5;

%Unit Impulse Signal

impulse=1.\*(t==0)+0.\*(t~=0);

subplot(4,3,1);

stem(t,impulse)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Impulse(4D4)");

axis([min(t),max(t), min(impulse)-0.5,max(impulse)+0.5]);

%Unit Step Signal

step=1.\*(t>=0)+0.\*(t<0);

subplot(4,3,2);

stem(t,step)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Step(4D4)");

axis([min(t),max(t), min(step)-0.5,max(step)+0.5]);

%Unit Ramp Signal

ramp=t.\*(t>=0)+0.\*(t<0);

subplot(4,3,3);

stem(t,ramp)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Ramp(4D4)");

axis([min(t),max(t), min(ramp)-0.5,max(ramp)+0.5]);

%Unit Parabolic Signal

parabolic=t.^2/2.\*(t>=0)+0.\*(t<0);

subplot(4,3,4);

stem(t,parabolic)

xlabel("Iime");

ylabel("amplitude");

title("Unit Parabolic(4D4)");

axis([min(t),max(t), min(parabolic)-0.5,max(parabolic)+0.5]);

%Unit Square Signal

square=1.\*(abs(t)<=0.5)+0.\*(abs(t)>0.5);

subplot(4,3,5);

stem(t,square)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Square(4D4)");

axis([min(t),max(t), min(square)-0.5,max(square)+0.5]);

%Unit Triangular Signal

triangular=(1-(abs(t)/2)).\*(abs(t)<=2)+0.\*(abs(t)>2);

subplot(4,3,6);

stem(t,triangular)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Triangular(4D4)");

axis([min(t),max(t), min(triangular)-0.5,max(triangular)+0.5]);

%Unit Signum Signal

signum=1.\*(t>=0)-1.\*(t<0);

subplot(4,3,7);

stem(t,signum)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Signum(4D4)");

axis([min(t),max(t), min(signum)-0.5,max(signum)+0.5]);

%Unit Sinc Signal

u\_sinc=sinc(t);

subplot(4,3,8);

stem(t,u\_sinc)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Sinc(4D4)");

axis([min(t),max(t), min(u\_sinc)-0.5,max(u\_sinc)+0.5]);

%Unit Real Exponential Signal

real\_exponential=exp(5.\*t).\*(t>=0)+0.\*(t<0);

subplot(4,3,9);

stem(t,real\_exponential)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Real Exponential(4D4)");

axis([min(t),max(t), min(real\_exponential)-0.5,max(real\_exponential)+0.5]);

%time interval for sin and cos Signals

t1=0:pi/12:2\*pi;

%Unit Sin Signal

u\_sin=sin(t1);

subplot(4,3,10);

stem(t1,u\_sin)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Sinusoidal(4D4)");

axis([min(t1),max(t1), min(u\_sin)-0.5,max(u\_sin)+0.5]);

%Unit Cosine Signal

u\_cos=cos(t1);

subplot(4,3,11);

stem(t1,u\_cos)

xlabel("Iime");

ylabel("Amplitude");

title("Unit Cosinusoidal(4D4)");

axis([min(t1),max(t1), min(u\_cos)-0.5,max(u\_cos)+0.5]);

%time interval for saw tooth Signal

F=input('enter the frequency:');

T=1/F;

t=0:5\*T/100:5\*T;

%Umit Sawtooth Signal

u\_saw\_tooth=sawtooth(2\*pi\*F\*t,1/2);

subplot(4,3,12);

stem(t,u\_saw\_tooth)

xlabel("Iime");

ylabel("Amplitude");

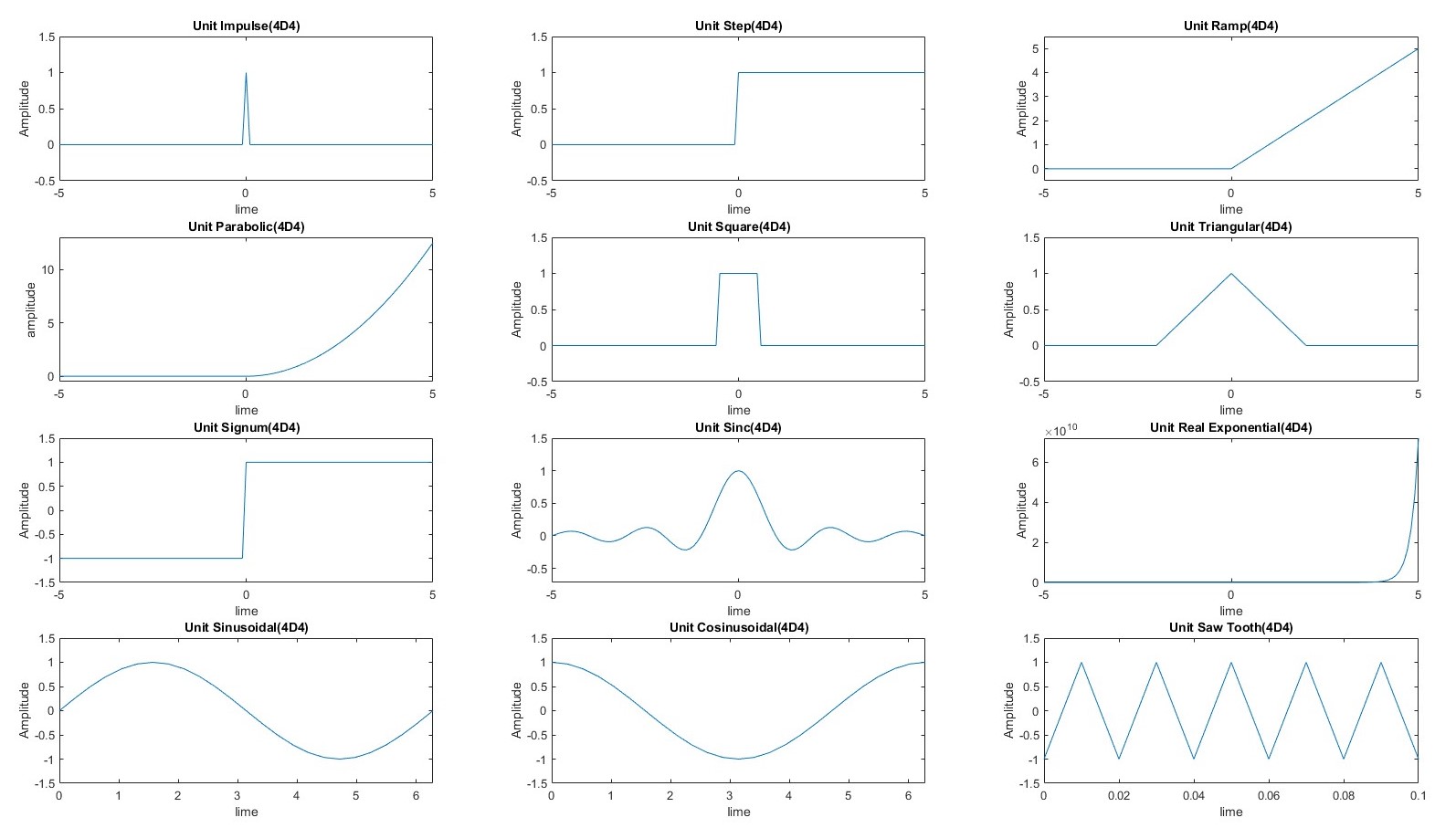
title("Unit Saw Tooth(4D4)");

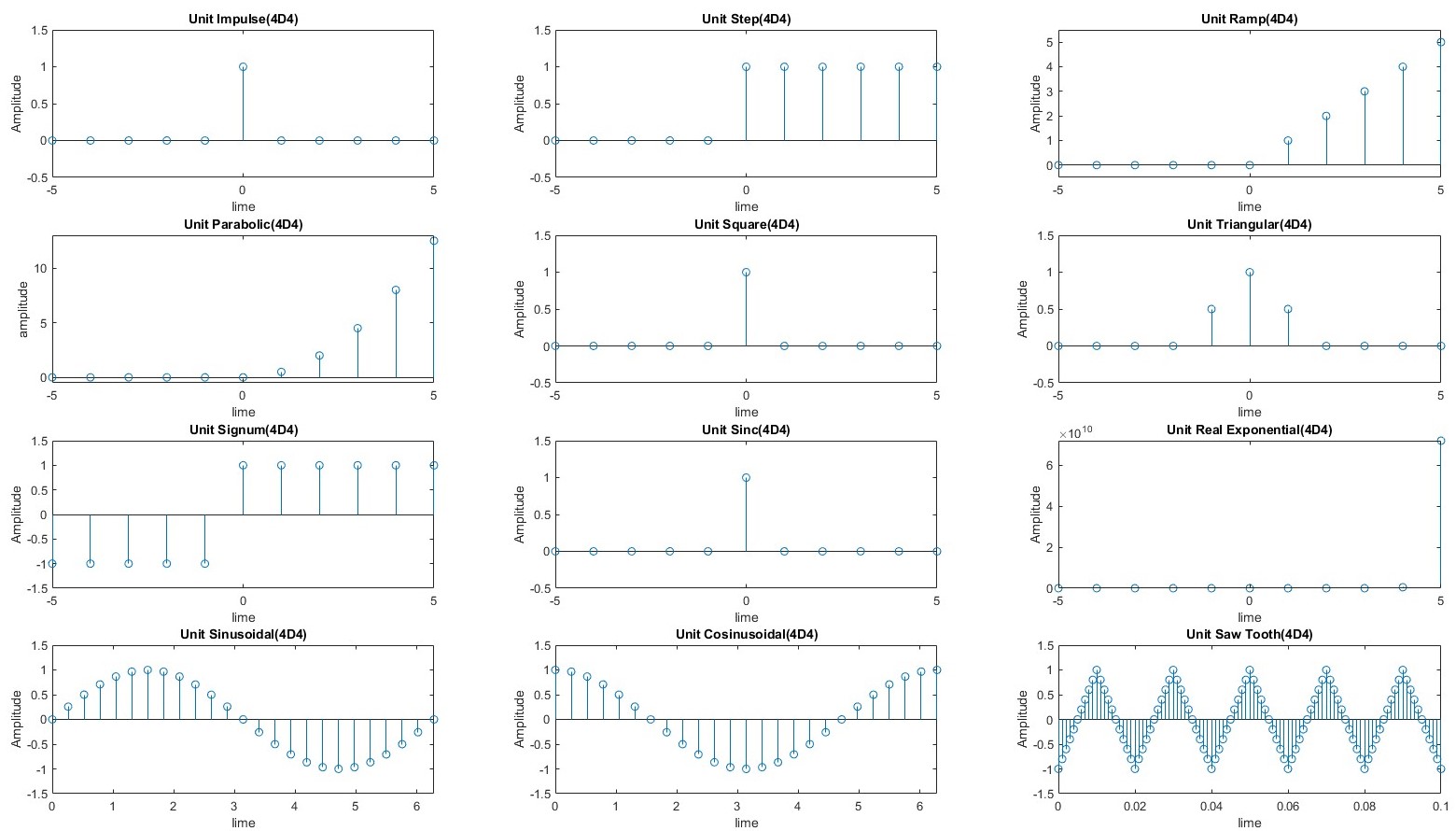
axis([min(t),max(t), min(u\_saw\_tooth)-0.5,max(u\_saw\_tooth)+0.5]);

**LAB PROCEDURE:**

1. Open the MATLAB® software by double clicking its icon.
2. MATLAB®logo will appear and after few moments Command Prompt will appear.
3. Go to the File Menu and select a New M‐file. (File New M‐file) or in the left-hand corner a blank white paper icon will be there. Click it once.
4. A blank M‐file will appear with a title ‘untitled’
5. Now start typing your program. After completing, save the M‐file with appropriate name. To execute the program Press F5 or go to Debug Menu and select Run.
6. After execution output will appear in the Command window. If there is an error then with an alarm, type of error will appear in red color.
7. Rectify the error if any and go to Debug Menu and select Run.

**OUTPUT WAVEFORMS /GRAPHS:**

****Continuous Time Signal

****

Discrete Time Signal

**RESULTS**:

Successfully Implemented Generation of various signals and sequences (Periodic and Aperiodic).

**Experiment No. 03**

**Aim**: Operations on signals and sequences such as Addition, Multiplication, Scaling, Shifting, Folding, Computation of Energy and Average Power.

S**oftware used:** MATLAB.

**THEORY:**

1. **Input Signal 1[x1(t)]**

**t+2, -2<=t<=-1**

**1, -1<t<=0**

**2, 0<t<=1**

**2-t, 1<t<=2**

1. **Input Signal 2[x2(t)]**

**1, -2<=t<=-1**

**2, -1<t<=1**

**1, 1<t<=2**

**3-t, 2<t<=3**

**MATLAB CODE**:

**Continuous Time Signal [CTS]:**

**Arithmetic Operations**

close all;

clear all;

%time range

t=-5:0.01:+5;

%Signal 1

Signal\_1 = (t+2).\*(t>=-2 & t<=-1) + (1).\*(t>-1 & t<=0)+ (2).\*(t>0 & t<=1)+ (2-t).\*(t>1 & t<=2);

subplot(3,3,1);

plot(t,Signal\_1);

xlabel("Time");

ylabel("Amplitude");

title("Signal 1");

axis([min(t),max(t),min(Signal\_1)-0.5,max(Signal\_1)+0.5]);

%Signal 2

Signal\_2 = (1).\*(t>=-2 & t<=-1)+ (2).\*(t>-1 & t<=1) + (1).\*(t>1 & t<=2) + (3-t).\*(t>2 & t<=3);

subplot(3,3,2);

plot(t,Signal\_2);

xlabel("Time");

ylabel("Amplitude");

title("Signal 2");

axis([min(t),max(t),min(Signal\_2)-0.5,max(Signal\_2)+0.5]);

%Signal Addition

Signal\_Addition = Signal\_1 + Signal\_2;

subplot(3,3,3);

plot(t,Signal\_Addition);

xlabel("Time");

ylabel("Amplitude");

title("Signal Addition(4D4)");

axis([min(t),max(t),min(Signal\_Addition)-0.5,max(Signal\_Addition)+0.5]);

%Signal Multiplication

Signal\_Multiplication = Signal\_1 .\* Signal\_2;

subplot(3,3,4);

plot(t,Signal\_Multiplication);

xlabel("Time");

ylabel("Amplitude");

title("Signal Multiplication(4D4)");

axis([min(t),max(t),min(Signal\_Multiplication)-0.5,max(Signal\_Multiplication)+0.5]);

%Signal Subtraction

Signal\_Subtraction = Signal\_1 - Signal\_2;

subplot(3,3,5);

plot(t,Signal\_Subtraction);

xlabel("Time");

ylabel("Amplitude");

title("Signal Subtraction(4D4)");

axis([min(t),max(t),min(Signal\_Subtraction)-0.5,max(Signal\_Subtraction)+0.5]);

%Signal Division

Signal\_Division = Signal\_1 ./ Signal\_2;

subplot(3,3,6);

plot(t,Signal\_Division);

xlabel("Time");

ylabel("Amplitude");

title("Signal Division(4D4)");

axis([min(t),max(t),min(Signal\_Division)-0.5,max(Signal\_Division)+0.5]);

%Signal Exponential

Signal\_Exponential = Signal\_1 .^ Signal\_2;

subplot(3,3,7);

plot(t,Signal\_Exponential);

xlabel("Time");

ylabel("Amplitude");

title("Signal Exponential(4D4)");

axis([min(t),max(t),min(Signal\_Exponential)-0.5,max(Signal\_Exponential)+0.5]);

**Other Functions on Signals**

%Time Range

t=-5:0.01:+5;

%Signal 1

Signal\_1=Signal(t);

subplot(4,2,1);

plot(t,Signal\_1);

xlabel("Time");

ylabel("Amplitude");

title("Signal 1");

axis([min(t),max(t),min(Signal\_1)-0.5,max(Signal\_1)+0.5]);

%Signal Left Shifting

Signal\_Left\_Shifting=Signal(t+3);

subplot(4,2,2);

plot(t,Signal\_Left\_Shifting);

xlabel("Time");

ylabel("Amplitude");

title("Signal Left Shifting(4D4)");

axis([min(t),max(t),min(Signal\_Left\_Shifting)-0.5,max(Signal\_Left\_Shifting)+0.5]);

%Signal Right Shifting

Signal\_Right\_Shifting=Signal(t-3);

subplot(4,2,3);

plot(t,Signal\_Right\_Shifting);

xlabel("Time");

ylabel("Amplitude");

title("Signal Right Shifting(4D4)");

axis([min(t),max(t),min(Signal\_Right\_Shifting)-0.5,max(Signal\_Right\_Shifting)+0.5]);

%Signal Amplitude Expansion

Signal\_Amplitude\_Expansion=2.\*Signal(t);

subplot(4,2,4);

plot(t,Signal\_Amplitude\_Expansion);

xlabel("Time");

ylabel("Amplitude");

title("Signal Amplitude Expansion Scaling(4D4)");

axis([min(t),max(t),min(Signal\_Amplitude\_Expansion)-0.5,max(Signal\_Amplitude\_Expansion)+0.5]);

%Signal Amplitude Compression

Signal\_Amplitude\_Compression=0.5.\*Signal(t);

subplot(4,2,5);

plot(t,Signal\_Amplitude\_Compression);

xlabel("Time");

ylabel("Amplitude");

title("Signal Amplitude Compression Scaling(4D4)");

axis([min(t),max(t),min(Signal\_Amplitude\_Compression)-0.5,max(Signal\_Amplitude\_Compression)+0.5]);

%Signal Time Compression

Signal\_Time\_Compression=Signal(2.\*t);

subplot(4,2,6);

plot(t,Signal\_Time\_Compression);

xlabel("Time");

ylabel("Amplitude");

title("Signal Time Compression Scaling(4D4)");

axis([min(t),max(t),min(Signal\_Time\_Compression)-0.5,max(Signal\_Time\_Compression)+0.5]);

%Signal Time Expansion

Signal\_Time\_Expansion=Signal(0.5.\*t);

subplot(4,2,7);

plot(t,Signal\_Time\_Expansion);

xlabel("Time");

ylabel("Amplitude");

title("Signal Time Expansion Scaling(4D4)");

axis([min(t),max(t),min(Signal\_Time\_Expansion)-0.5,max(Signal\_Time\_Expansion)+0.5]);

%Signal Folding

Signal\_Reverse=Signal(-t);

subplot(4,2,8);

plot(t,Signal\_Reverse);

xlabel("Time");

ylabel("Amplitude");

title("Signal Folding(4D4)");

axis([min(t),max(t),min(Signal\_Reverse)-0.5,max(Signal\_Reverse)+0.5]);

**Discrete Time Signal [DTS]:**

**Arithmetic Operations**

close all;

clear all;

%time range

t=-5:1:+5;

%Signal 1

Signal\_1 = (t+2).\*(t>=-2 & t<=-1) + (1).\*(t>-1 & t<=0)+ (2).\*(t>0 & t<=1)+ (2-t).\*(t>1 & t<=2);

subplot(3,3,1);

stem(t,Signal\_1);

xlabel("Time");

ylabel("Amplitude");

title("Signal 1");

axis([min(t),max(t),min(Signal\_1)-0.5,max(Signal\_1)+0.5]);

%Signal 1

Signal\_2 = (1).\*(t>=-2 & t<=-1)+ (2).\*(t>-1 & t<=1) + (1).\*(t>1 & t<=2) + (3-t).\*(t>2 & t<=3);

subplot(3,3,2);

stem(t,Signal\_2);

xlabel("Time");

ylabel("Amplitude");

title("Signal 2");

axis([min(t),max(t),min(Signal\_2)-0.5,max(Signal\_2)+0.5]);

%Signal Addition

Signal\_Addition = Signal\_1 + Signal\_2;

subplot(3,3,3);

stem(t,Signal\_Addition);

xlabel("Time");

ylabel("Amplitude");

title("Signal Addition(4D4)");

axis([min(t),max(t),min(Signal\_Addition)-0.5,max(Signal\_Addition)+0.5]);

%Signal Multiplication

Signal\_Multiplication = Signal\_1 .\* Signal\_2;

subplot(3,3,4);

stem(t,Signal\_Multiplication);

xlabel("Time");

ylabel("Amplitude");

title("Signal Multiplication(4D4)");

axis([min(t),max(t),min(Signal\_Multiplication)-0.5,max(Signal\_Multiplication)+0.5]);

%Signal Subtraction

Signal\_Subtraction = Signal\_1 - Signal\_2;

subplot(3,3,5);

stem(t,Signal\_Subtraction);

xlabel("Time");

ylabel("Amplitude");

title("Signal Subtraction(4D4)");

axis([min(t),max(t),min(Signal\_Subtraction)-0.5,max(Signal\_Subtraction)+0.5]);

%Signal Division

Signal\_Division = Signal\_1 ./ Signal\_2;

subplot(3,3,6);

stem(t,Signal\_Division);

xlabel("Time");

ylabel("Amplitude");

title("Signal Division(4D4)");

axis([min(t),max(t),min(Signal\_Division)-0.5,max(Signal\_Division)+0.5]);

%Signal Exponential

Signal\_Exponential = Signal\_1 .^ Signal\_2;

subplot(3,3,7);

stem(t,Signal\_Exponential);

xlabel("Time");

ylabel("Amplitude");

title("Signal Exponential(4D4)");

axis([min(t),max(t),min(Signal\_Exponential)-0.5,max(Signal\_Exponential)+0.5]);

**Other Functions on Signals**

%Time Range

t=-5:1:+5;

%Signal 1

Signal\_1=Signal(t);

subplot(4,2,1);

stem(t,Signal\_1);

xlabel("Time");

ylabel("Amplitude");

title("Signal 1");

axis([min(t),max(t),min(Signal\_1)-0.5,max(Signal\_1)+0.5]);

%Signal Left Shifting

Signal\_Left\_Shifting=Signal(t+3);

subplot(4,2,2);

stem(t,Signal\_Left\_Shifting);

xlabel("Time");

ylabel("Amplitude");

title("Signal Left Shifting(4D4)");

axis([min(t),max(t),min(Signal\_Left\_Shifting)-0.5,max(Signal\_Left\_Shifting)+0.5]);

%Signal Right Shifting

Signal\_Right\_Shifting=Signal(t-3);

subplot(4,2,3);

stem(t,Signal\_Right\_Shifting);

xlabel("Time");

ylabel("Amplitude");

title("Signal Right Shifting(4D4)");

axis([min(t),max(t),min(Signal\_Right\_Shifting)-0.5,max(Signal\_Right\_Shifting)+0.5]);

%Signal Amplitude Expansion

Signal\_Amplitude\_Expansion=2.\*Signal(t);

subplot(4,2,4);

stem(t,Signal\_Amplitude\_Expansion);

xlabel("Time");

ylabel("Amplitude");

title("Signal Amplitude Expansion Scaling(4D4)");

axis([min(t),max(t),min(Signal\_Amplitude\_Expansion)-0.5,max(Signal\_Amplitude\_Expansion)+0.5]);

%Signal Amplitude Compression

Signal\_Amplitude\_Compression=0.5.\*Signal(t);

subplot(4,2,5);

stem(t,Signal\_Amplitude\_Compression);

xlabel("Time");

ylabel("Amplitude");

title("Signal Amplitude Compression Scaling(4D4)");

axis([min(t),max(t),min(Signal\_Amplitude\_Compression)-0.5,max(Signal\_Amplitude\_Compression)+0.5]);

%Signal Time Compression

Signal\_Time\_Compression=Signal(2.\*t);

subplot(4,2,6);

stem(t,Signal\_Time\_Compression);

xlabel("Time");

ylabel("Amplitude");

title("Signal Time Compression Scaling(4D4)");

axis([min(t),max(t),min(Signal\_Time\_Compression)-0.5,max(Signal\_Time\_Compression)+0.5]);

%Signal Time Expansion

Signal\_Time\_Expansion=Signal(0.5.\*t);

subplot(4,2,7);

stem(t,Signal\_Time\_Expansion);

xlabel("Time");

ylabel("Amplitude");

title("Signal Time Expansion Scaling(4D4)");

axis([min(t),max(t),min(Signal\_Time\_Expansion)-0.5,max(Signal\_Time\_Expansion)+0.5]);

%Signal Folding

Signal\_Reverse=Signal(-t);

subplot(4,2,8);

stem(t,Signal\_Reverse);

xlabel("Time");

ylabel("Amplitude");

title("Signal Folding(4D4)");

axis([min(t),max(t),min(Signal\_Reverse)-0.5,max(Signal\_Reverse)+0.5]);

**Energy and Average Power Computation**

%TimeStamp

T = 10;

%Time Range

t=-T:0.01:T;

Signal\_1=T.\*sin(2\*pi.\*t);

Signal\_2 = Signal\_1.^2;

Energy = trapz(t,Signal\_2);

Power = Energy/(2\*T);

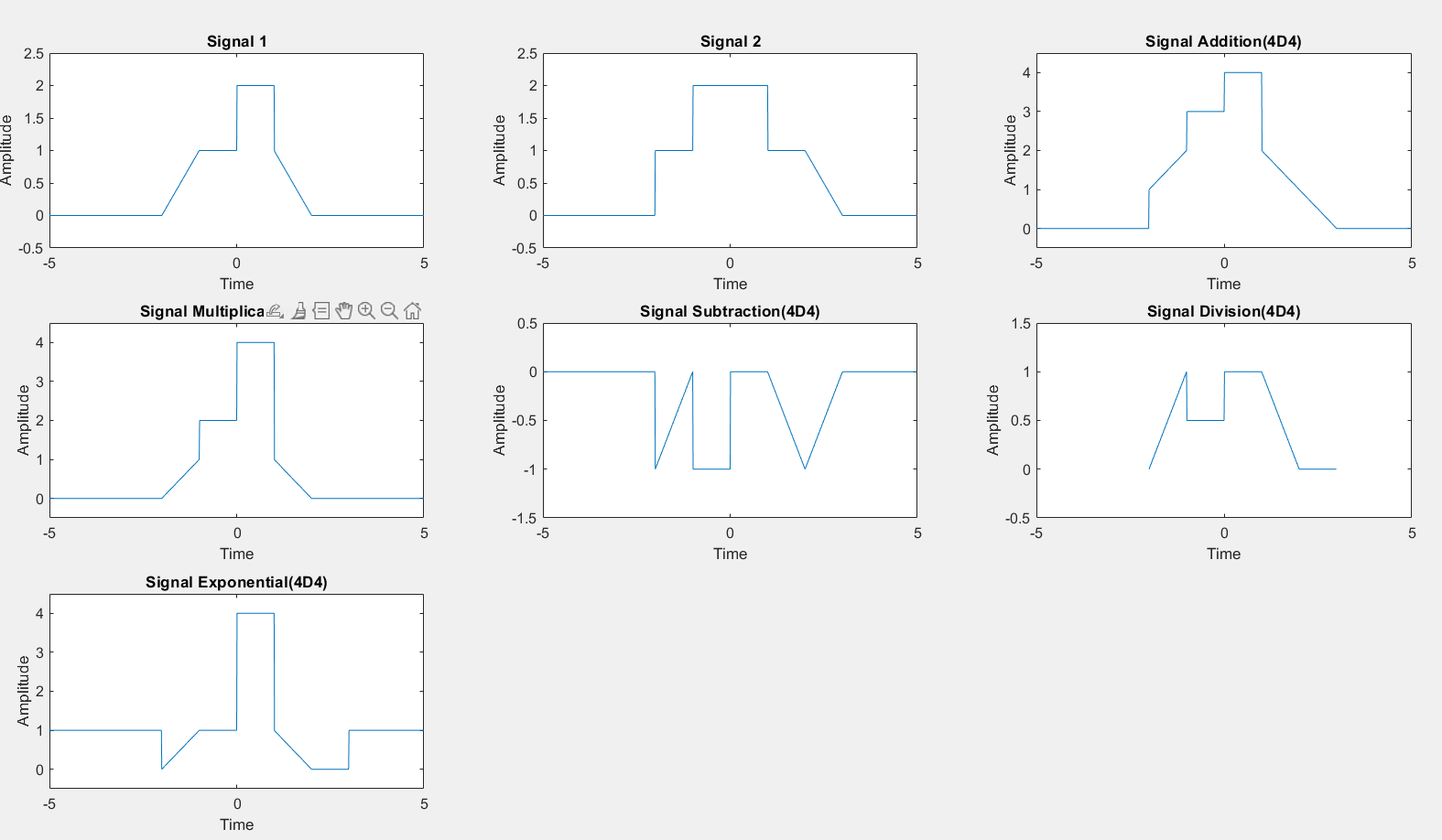
disp(['Energy:',num2str(Energy),'Joule']);

disp(['Power:',num2str(Power),'Watts']);

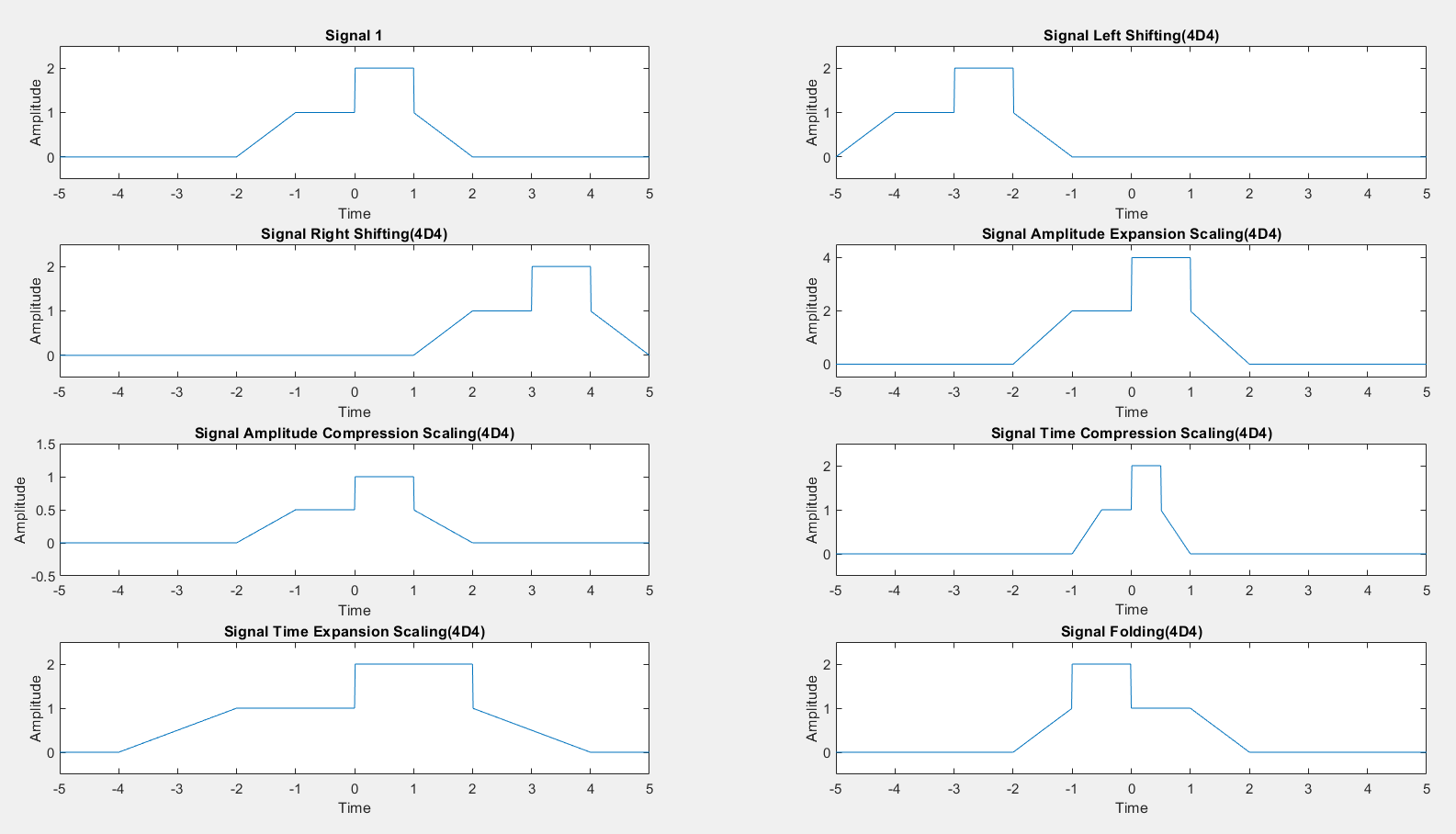
**LAB PROCEDURE:**

1. Open the MATLAB® software by double clicking its icon.
2. MATLAB®logo will appear and after few moments Command Prompt will appear.
3. Go to the File Menu and select a New M‐file. (File New M‐file) or in the left-hand corner a blank white paper icon will be there. Click it once.
4. A blank M‐file will appear with a title ‘untitled’
5. Now start typing your program. After completing, save the M‐file with appropriate name. To execute the program Press F5 or go to Debug Menu and select Run.
6. After execution output will appear in the Command window. If there is an error then with an alarm, type of error will appear in red color.
7. Rectify the error if any and go to Debug Menu and select Run.

**OUTPUT WAVEFORMS /GRAPHS:**

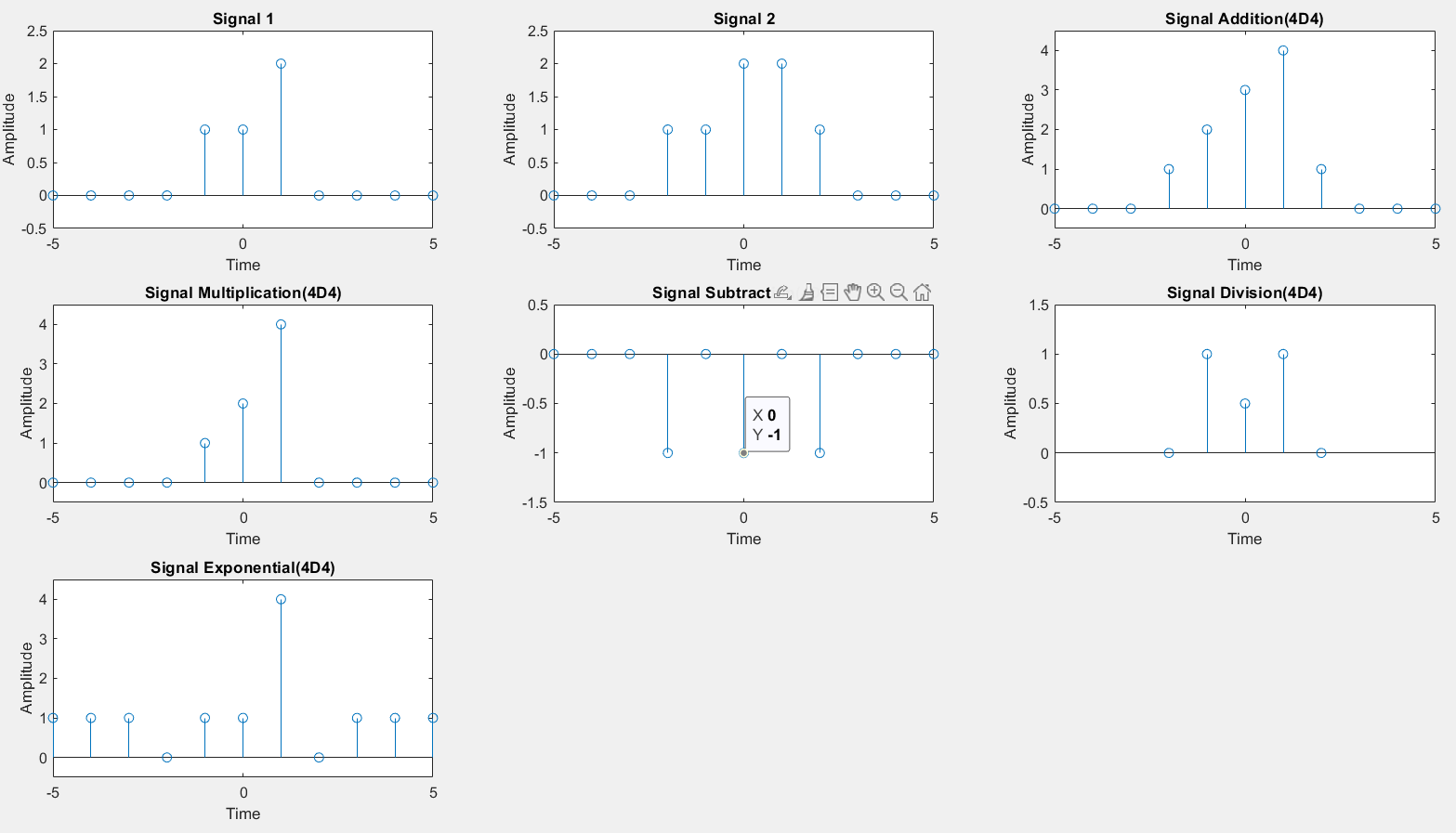
****

**Arithmetic Operations on Signals**

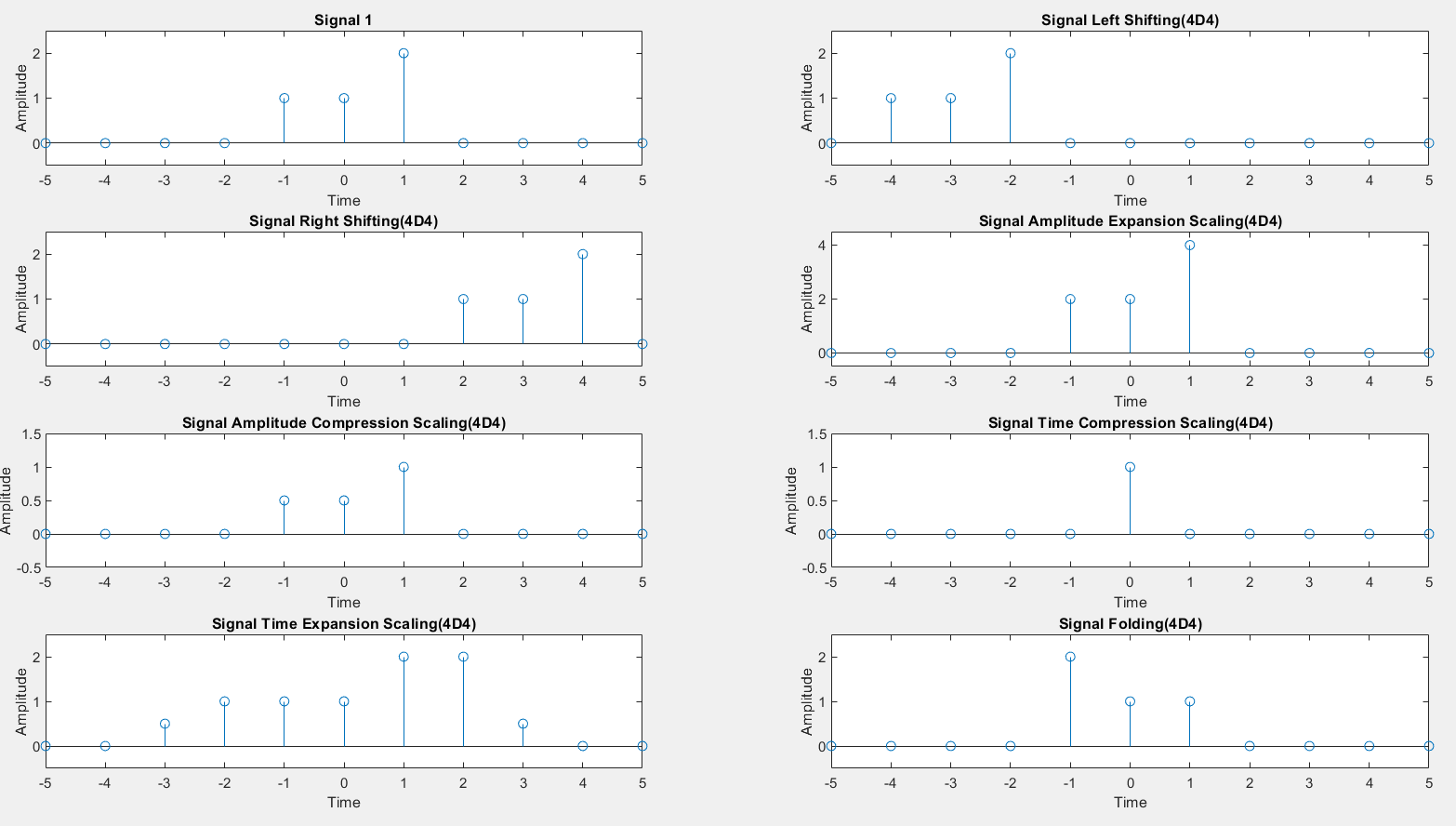
****

**Other Operations on Signals**

Continuous Time Signal

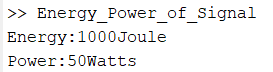
****

**Arithmetic Operations on Signals**

****

**Other Operations on Signals**

Discrete Time Signal



Computation of Energy and Power

**RESULTS**:

Successfully Implemented Operations on signals and sequences such as Addition, Multiplication, Scaling, Shifting, Folding, Computation of Energy and Average Power.

**Experiment No. 04**

**Aim**: Finding the Even and Odd parts of Signal / Sequence and Real and imaginary parts of Signal.

S**oftware used:** MATLAB.

**THEORY:**

**Continuous Time Signal[CTS]:**

x(t)= xe(t)+x0(t) ----(1)

x(-t)= xe(-t)+x0(-t) = xe(t)-x0(t) ----(2)

(1) + (2)

2xe(t) = x(t)+ x(-t)

xe(t) = [x(t)+ x(-t)]/2

2xo(t) = x(t)-x(-t)

xo(t) = [x(t)-x(-t)]/2

**Discrete Time Signal[DTS]:**

x(n)= xe(n)+x0(n) ----(3)

x(-n)= xe(-t)+x0(-n) = xe(n)-x0(n) ----(4)

(3) + (4)

2xe(n) = x(n)+ x(-n)

xe(n) = [x(n)+ x(-n)]/2

2xo(n) = x(n)-x(-n)

xo(n) = [x(n)-x(-n)]/2

let x(t) = ej2πt

**MATLAB CODE**:

**Continuous Time Signal [CTS]:**

clear all

close all

t = -5:0.001:5;

x1 = exp(1i\*2\*pi.\*t); %Exponential Signal

%x1 = cos(t); Cosine Signal

%x1 = sin(t); Sine Signal

subplot(3,2,1);

plot(t,x1);

xlabel("Time");

ylabel("Amplitude");

title("Signal 1(4D4)");

grid;

x2 = exp((-1i)\*2\*pi.\*t); %Exponential Signal

%x2 = cos(-t); Cosine Signal

%x2 = sin(-t); Sine Signal

subplot(3,2,2);

plot(t,x2);

xlabel("Time");

ylabel("Amplitude");

title("Signal 2(4D4)");

grid;

if(x1==x2)

disp("The given Signal is Even");

else if(x1==-x2)

disp("The given Signal is Odd");

else

disp("The given Signal is Neither Even Nor Odd");

end

end

xe = (x1+x2)/2;

subplot(3,2,3);

plot(t,xe);

xlabel("Time");

ylabel("Amplitude");

title("Even Signal(4D4)");

grid;

xo = (x1-x2)/2;

subplot(3,2,4);

plot(t,xo);

xlabel("Time");

ylabel("Amplitude");

title("Odd Signal(4D4)");

grid;

Real = real(x1);

subplot(3,2,5);

plot(t,Real);

xlabel("Time");

ylabel("Amplitude");

title("Real Part of Signal(4D4)");

grid;

Imagimary = imag(x2);

subplot(3,2,6);

plot(t,Imagimary);

xlabel("Time");

ylabel("Amplitude");

title("Imagimary Part of Signal(4D4)");

grid;

**Discrete Time Signal [DTS]:**

clear all

close all

t = -5:1:5;

x1 = exp(1i\*2\*pi.\*t); %Exponential Signal

%x1 = cos(t); Cosine Signal

%x1 = sin(t); Sine Signal

subplot(2,2,1);

stem(t,x1);

xlabel("Time");

ylabel("Amplitude");

title("Signal 1(4D4)");

grid;

%x2 = exp((-1i)\*2\*pi.\*t); Exponential Signal

%x2 = cos(-t); Cosine Signal

x2 = sin(-t); %Sine Signal

subplot(2,2,2);

stem(t,x2);

xlabel("Time");

ylabel("Amplitude");

title("Signal 2(4D4)");

grid;

if(x1==x2)

disp("The given Signal is Even");

else if(x1==-x2)

disp("The given Signal is Odd");

else

disp("The given Signal is Neither Even Nor Odd");

end

end

xe = (x1+x2)/2;

subplot(2,2,3);

stem(t,xe);

xlabel("Time");

ylabel("Amplitude");

title("Even Signal(4D4)");

grid;

xo = (x1-x2)/2;

subplot(2,2,4);

stem(t,xo);

xlabel("Time");

ylabel("Amplitude");

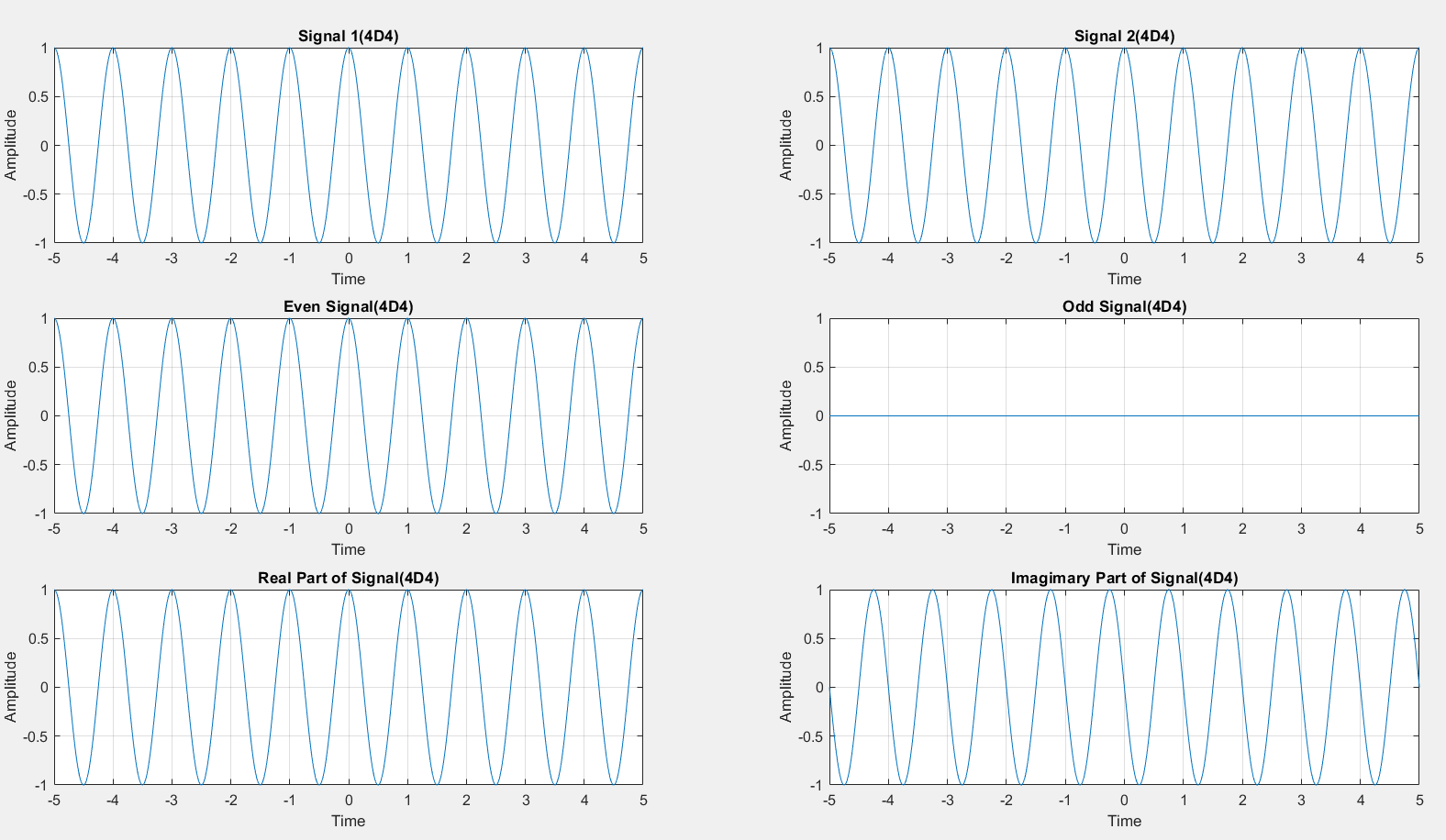
title("Odd Signal(4D4)");

grid;

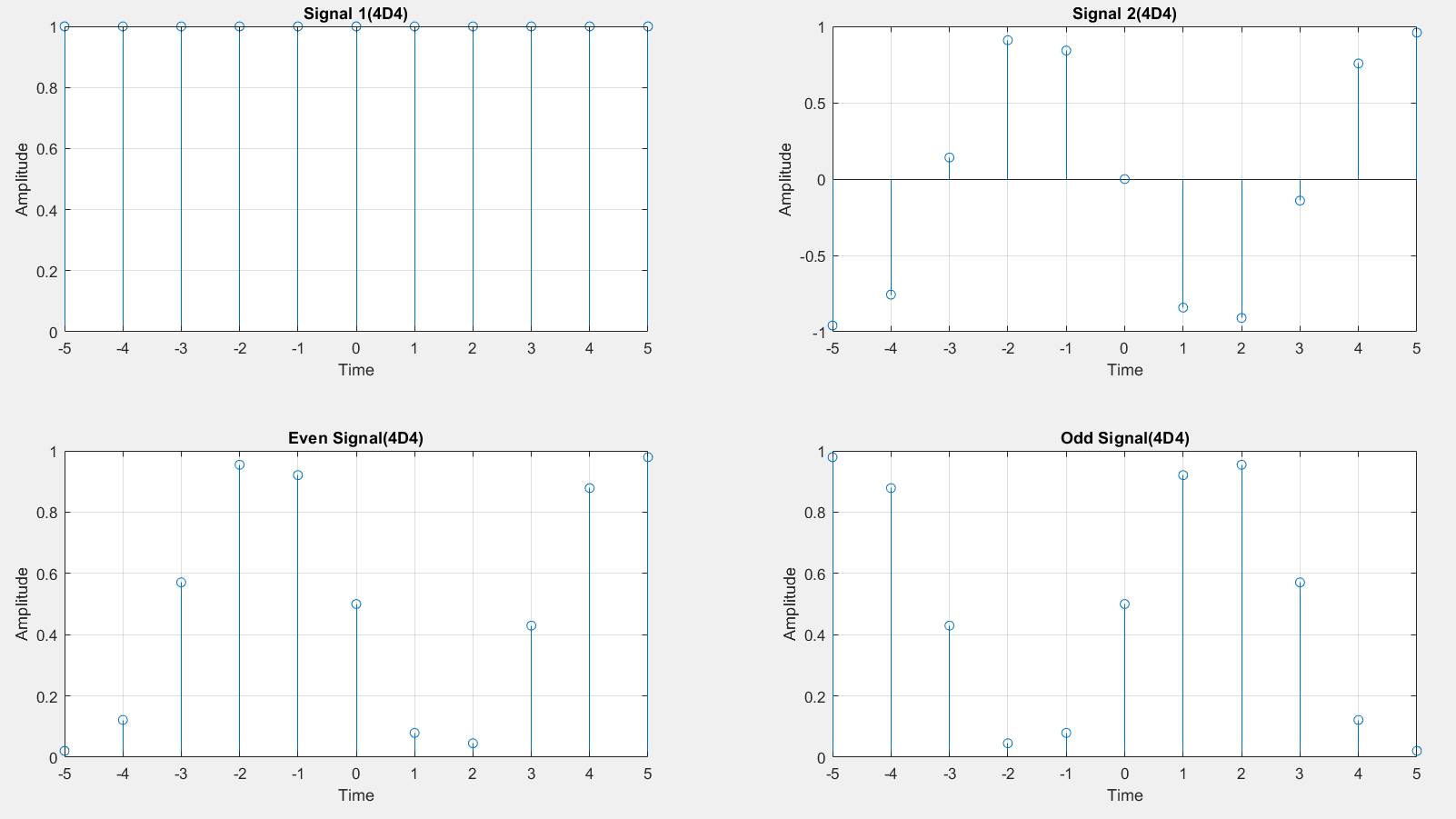
**LAB PROCEDURE:**

1. Open the MATLAB® software by double clicking its icon.
2. MATLAB®logo will appear and after few moments Command Prompt will appear.
3. Go to the File Menu and select a New M‐file. (File New M‐file) or in the left-hand corner a blank white paper icon will be there. Click it once.
4. A blank M‐file will appear with a title ‘untitled’
5. Now start typing your program. After completing, save the M‐file with appropriate name. To execute the program Press F5 or go to Debug Menu and select Run.
6. After execution output will appear in the Command window. If there is an error then with an alarm, type of error will appear in red color.
7. Rectify the error if any and go to Debug Menu and select Run.

**OUTPUT WAVEFORMS /GRAPHS:**

****

Continuous Time Signal

****

Discrete Time Signal

**RESULTS**:

Successfully Implemented Finding the Even and Odd parts of Signal / Sequence and Real and imaginary parts of Signal.

**Experiment No. 05**

**Aim**: Verification of Gibb’s Phenomenon.

S**oftware used:** MATLAB.

**MATLAB CODE**:

t=-3:0.01:3;

w=2\*pi;

c0=zeros(1,length(t));

for i = 1:4

x=c0;

N=input("Enter no of Oscillations:");

for n=1:N

cn=(2/n\*pi)\*sin(n\*pi/2);

cnn=cn;

x=x+(cn)\*exp(1i\*n\*w.\*t)+(cnn)\*exp(-1i\*n\*w.\*t);

end

subplot(2,2,i);

plot(t,x);

xlabel("time");

ylabel("Amplitude");

title("Gibbs Phenomenon for N=",num2str(N));

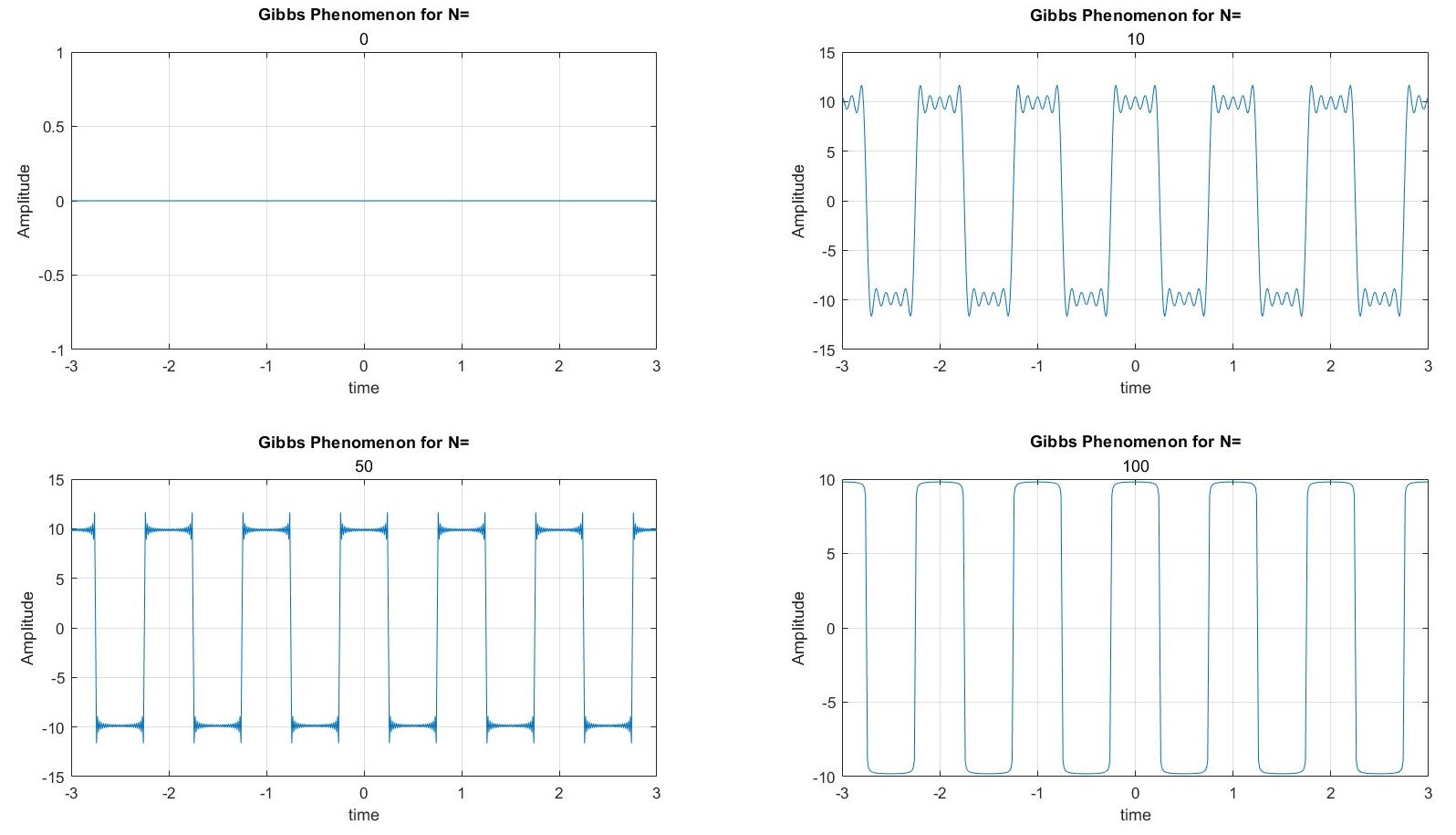
grid

end

**LAB PROCEDURE:**

1. Open the MATLAB® software by double clicking its icon.
2. MATLAB®logo will appear and after few moments Command Prompt will appear.
3. Go to the File Menu and select a New M‐file. (File New M‐file) or in the left-hand corner a blank white paper icon will be there. Click it once.
4. A blank M‐file will appear with a title ‘untitled’
5. Now start typing your program. After completing, save the M‐file with appropriate name. To execute the program Press F5 or go to Debug Menu and select Run.
6. After execution output will appear in the Command window. If there is an error then with an alarm, type of error will appear in red color.
7. Rectify the error if any and go to Debug Menu and select Run.

**OUTPUT WAVEFORMS /GRAPHS:**

****

Gibb’s Phenomenon

**RESULTS**:

Successfully Verified Gibb’s Phenomenon using Matlab.

**Experiment No. 06**

**Aim**: Finding the Fourier Transform of a given signal and plotting its magnitude and phase spectrum.

S**oftware used:** MATLAB.

**MATLAB CODE**:

t = -10:0.1:10;

syms t w;

x = int(t\*exp(-1i\*w\*t),t,[0,8]);

w1 = -10:0.07:10;

x1 = subs(x,w,w1);

mgx = abs(x1);

phx = angle(x1);

subplot(3,1,1);

plot(w1,x1);

xlabel("Frequency");

ylabel("Amplitude");

title("Given Signal(4D4)");

subplot(3,1,2);

plot(w1,mgx);

xlabel("Frequency");

ylabel("Magnitude");

title("Magnitude Spectrum(4D4)");

subplot(3,1,3);

plot(w1,phx);

xlabel("Frequency");

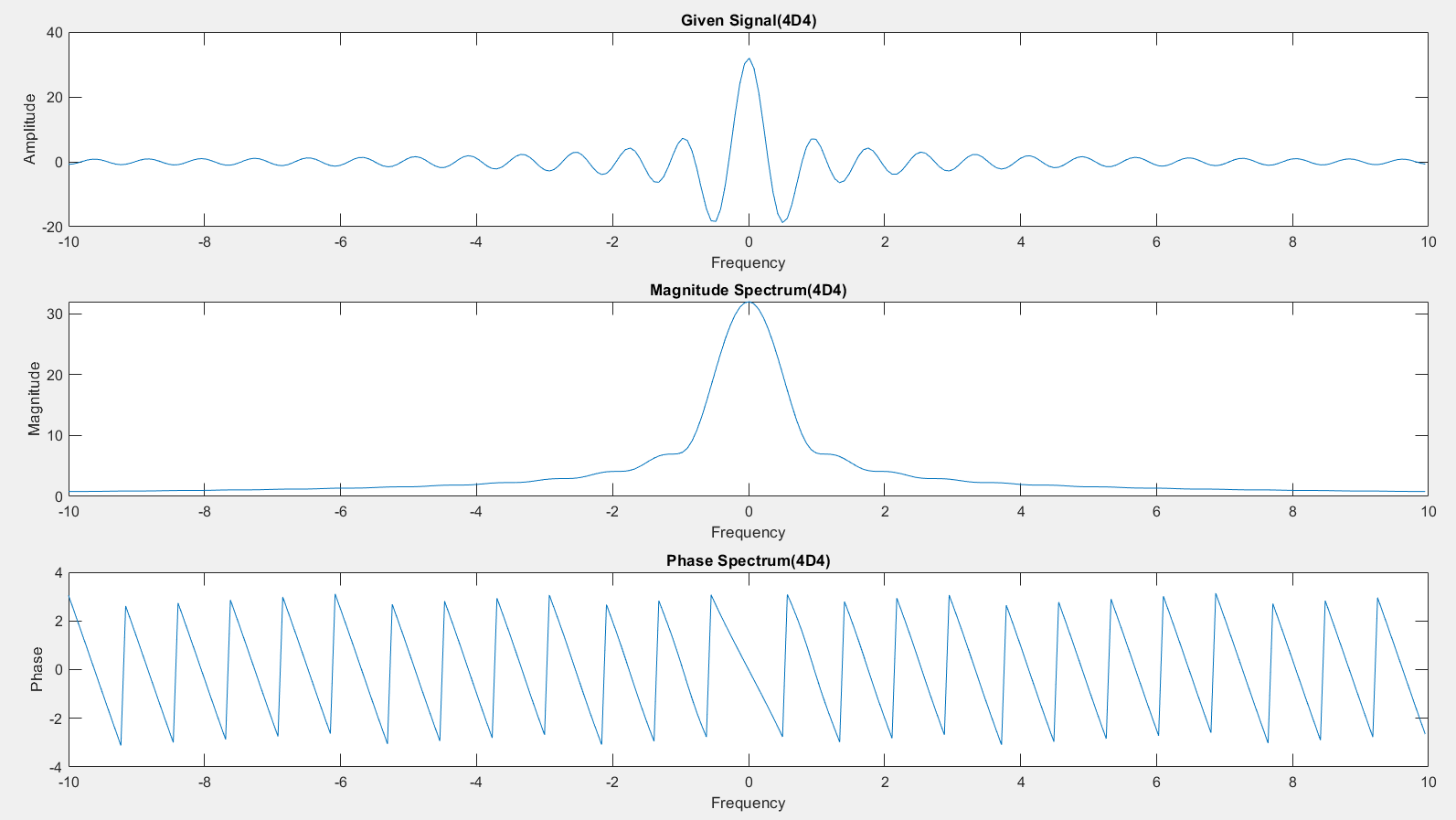
ylabel("Phase");

title("Phase Spectrum(4D4)");

**LAB PROCEDURE:**

1. Open the MATLAB® software by double clicking its icon.
2. MATLAB®logo will appear and after few moments Command Prompt will appear.
3. Go to the File Menu and select a New M‐file. (File New M‐file) or in the left-hand corner a blank white paper icon will be there. Click it once.
4. A blank M‐file will appear with a title ‘untitled’
5. Now start typing your program. After completing, save the M‐file with appropriate name. To execute the program Press F5 or go to Debug Menu and select Run.
6. After execution output will appear in the Command window. If there is an error then with an alarm, type of error will appear in red color.
7. Rectify the error if any and go to Debug Menu and select Run.

**OUTPUT WAVEFORMS /GRAPHS:**

****

Fourier Transform of Signal

**RESULTS**:

Successfully Implemented Finding the Fourier Transform of a given signal and plotting its magnitude and phase spectrum.

**Experiment No. 07**

**Aim**: Finding the Convolution between (i) Signals (ii) Sequences

S**oftware used:** MATLAB.

**MATLAB CODE**:

**Continuous Time Signal [CTS]:**

clear all;

close all;

t1=-2:0.01:2;

x=exp(-2.\*t1).\*(t1>=0) + 0.\*(t1<0);

t2=-1:0.01:3;

h=1.\*(t2>=0) + 0.\*(t2<0);

y=conv(x,h);

a=min(t1)+min(t2);

b=max(t1)+max(t2);

t3=a:0.01:b;

subplot(3,1,1);

plot(t1,x);

xlabel("time t");

ylabel("amplitude x");

title("input signal(4D4)");

subplot(3,1,2);

plot(t2,h);

xlabel("time t");

ylabel("amplitude x");

title("system response(4D4)");

subplot(3,1,3);

plot(t3,y)

xlabel("time t");

ylabel("amplitude x");

title("convloution(4D4)");

**Discrete Time Signal [DTS]:**

clear all;

close all;

n1=-1:1:2;

x=[1 2 3 -1];

n2=0:1:4;

h=[2 -1 -2 4 3];

y=conv(x,h);

a=min(n1)+min(n2);

b=max(n1)+max(n2);

n3=a:1:b;

subplot(3,1,1);

stem(n1,x);

xlabel("time t");

ylabel("amplitude x");

title("input signal(4D4)");

subplot(3,1,2);

stem(n2,h);

xlabel("time t");

ylabel("amplitude x");

title("system response(4D4)");

subplot(3,1,3);

stem(n3,y)

xlabel("time t");

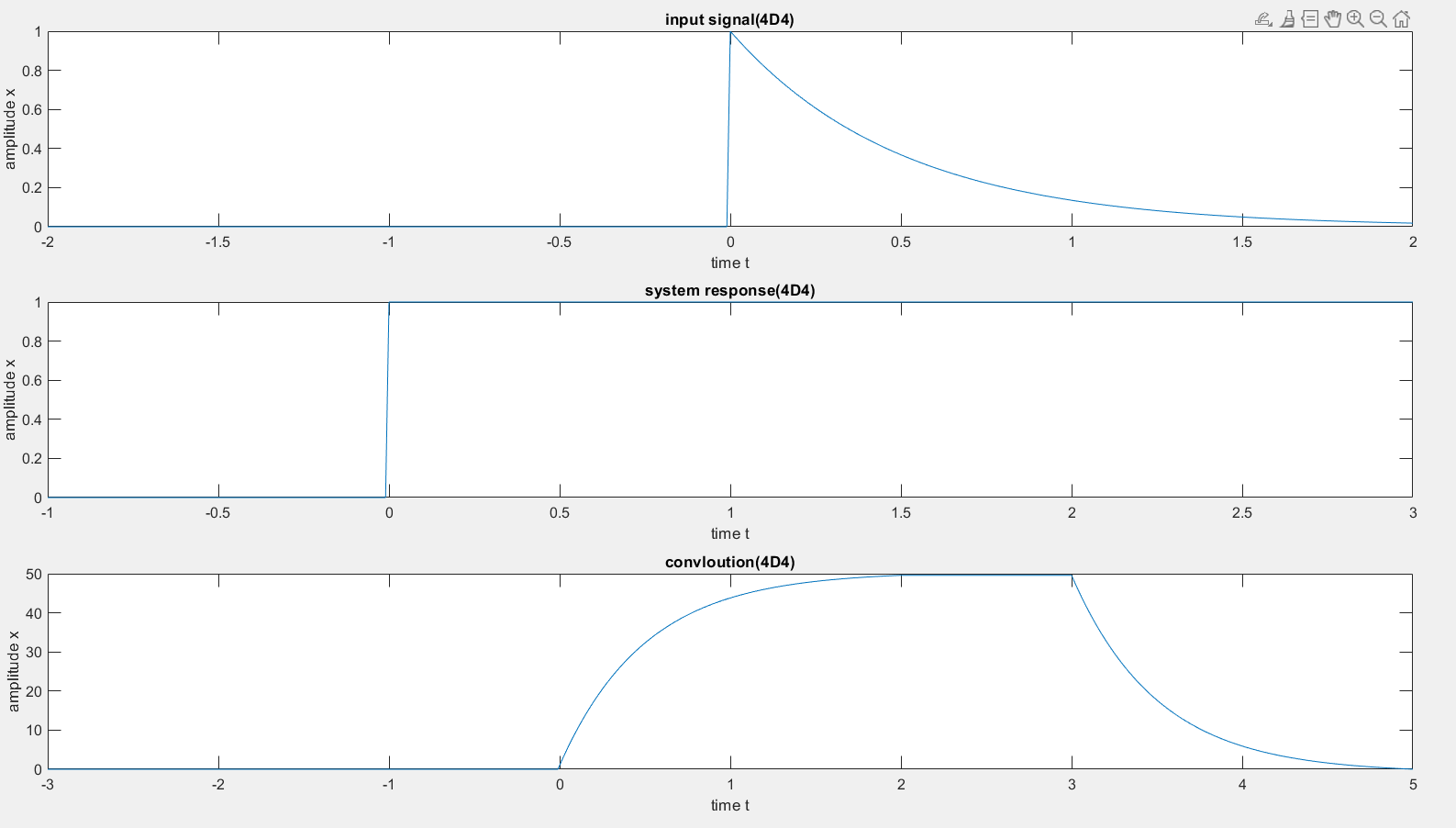
ylabel("amplitude x");

title("convloution(4D4)");

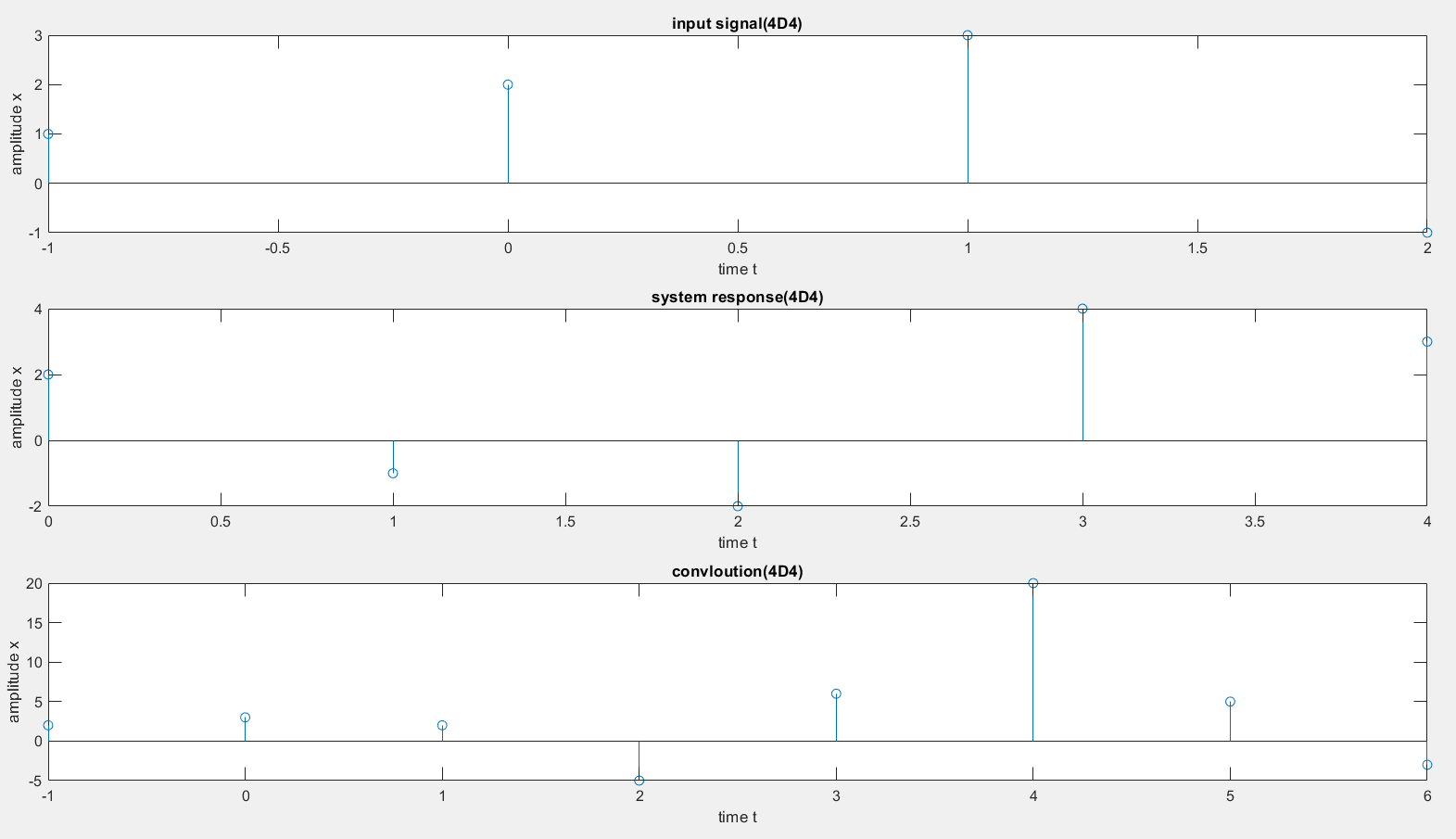
**LAB PROCEDURE:**

1. Open the MATLAB® software by double clicking its icon.
2. MATLAB®logo will appear and after few moments Command Prompt will appear.
3. Go to the File Menu and select a New M‐file. (File New M‐file) or in the left-hand corner a blank white paper icon will be there. Click it once.
4. A blank M‐file will appear with a title ‘untitled’
5. Now start typing your program. After completing, save the M‐file with appropriate name. To execute the program Press F5 or go to Debug Menu and select Run.
6. After execution output will appear in the Command window. If there is an error then with an alarm, type of error will appear in red color.
7. Rectify the error if any and go to Debug Menu and select Run.

**OUTPUT WAVEFORMS /GRAPHS:**

****

Continuous Time Signal



Discrete Time Signal

**RESULTS**:

Successfully Implemented Finding the Convolution between (i) Signals (ii) Sequences

**Experiment No. 08**

**Aim**: Finding the Auto Correlation and Cross Correlation of (i) Signals (ii) Sequences

S**oftware used:** MATLAB.

**MATLAB CODE**:

**Continuous Time Signal [CTS]:**

close all

clear all

t1 = -3:0.01:3;

x = 1.\*((t1>=0)&(t1<=2))+0.\*((t1<0)&(t1>2));

t2 = -t1;

a = min(t1)+min(t2);

b = max(t1)+max(t2);

t = a:0.01:b;

RXX = xcorr(x,x);

subplot(2,3,1);

plot(t1,x);

xlabel("Time");

ylabel("Amplitude");

title("Input Signal(4D4)");

subplot(2,3,2);

plot(t2,x);

xlabel("Time");

ylabel("Amplitude");

title("Received Signal(4D4)");

subplot(2,3,3);

plot(t,RXX);

xlabel("Time");

ylabel("Amplitude");

title("Auto Correlation(4D4)");

%Cross Correlation

t1 = -3:0.01:3;

x = 1.\*((t1>=0)&(t1<=2))+0.\*((t1<0)&(t1>2));

t2 = -3:0.01:3;

y = 1.\*((t2>=0)&(t2<=3))+0.\*((t2<0)&(t2>3));

t3 = -t2;

a = min(t1)+min(t3);

b = max(t1)+max(t3);

t = a:0.01:b;

RXY = xcorr(x,y);

subplot(2,3,4);

plot(t1,x);

xlabel("Time");

ylabel("Amplitude");

title("Input Signal(4D4)");

subplot(2,3,5);

plot(t3,y);

xlabel("Time");

ylabel("Amplitude");

title("Received Signal(4D4)");

subplot(2,3,6);

plot(t,RXY);

xlabel("Time");

ylabel("Amplitude");

title("Cross Correlation(4D4)");

**Discrete Time Signal [DTS]:**

close all

clear all

n1 = -3:1:3;

x = [1,2,3,4,5,6,7];

n2 = -n1;

a = min(n1)+min(n2);

b = max(n1)+max(n2);

n = a:1:b;

RXX = xcorr(x,x);

subplot(2,3,1);

stem(n1,x);

xlabel("Time");

ylabel("Amplitude");

title("Input Signal(4D4)");

subplot(2,3,2);

stem(n2,x);

xlabel("Time");

ylabel("Amplitude");

title("Received Signal");

subplot(2,3,3);

stem(n,RXX);

xlabel("Time");

ylabel("Amplitude");

title("Auto Correlation(4D4)");

%Cross Correlation

n1 = -3:1:3;

x = [1,2,3,4,5,6,7];

n2 = -3:1:3;

y = [2,5,8,7,9,3,6];

n3 = -n2;

a = min(n1)+min(n3);

b = max(n1)+max(n3);

n = a:1:b;

RXY = xcorr(x,y);

subplot(2,3,4);

stem(n1,x);

xlabel("Time");

ylabel("Amplitude");

title("Input Signal(4D4)");

subplot(2,3,5);

stem(n3,y);

xlabel("Time");

ylabel("Amplitude");

title("Received Signal(4D4)");

subplot(2,3,6);

stem(n,RXY);

xlabel("Time");

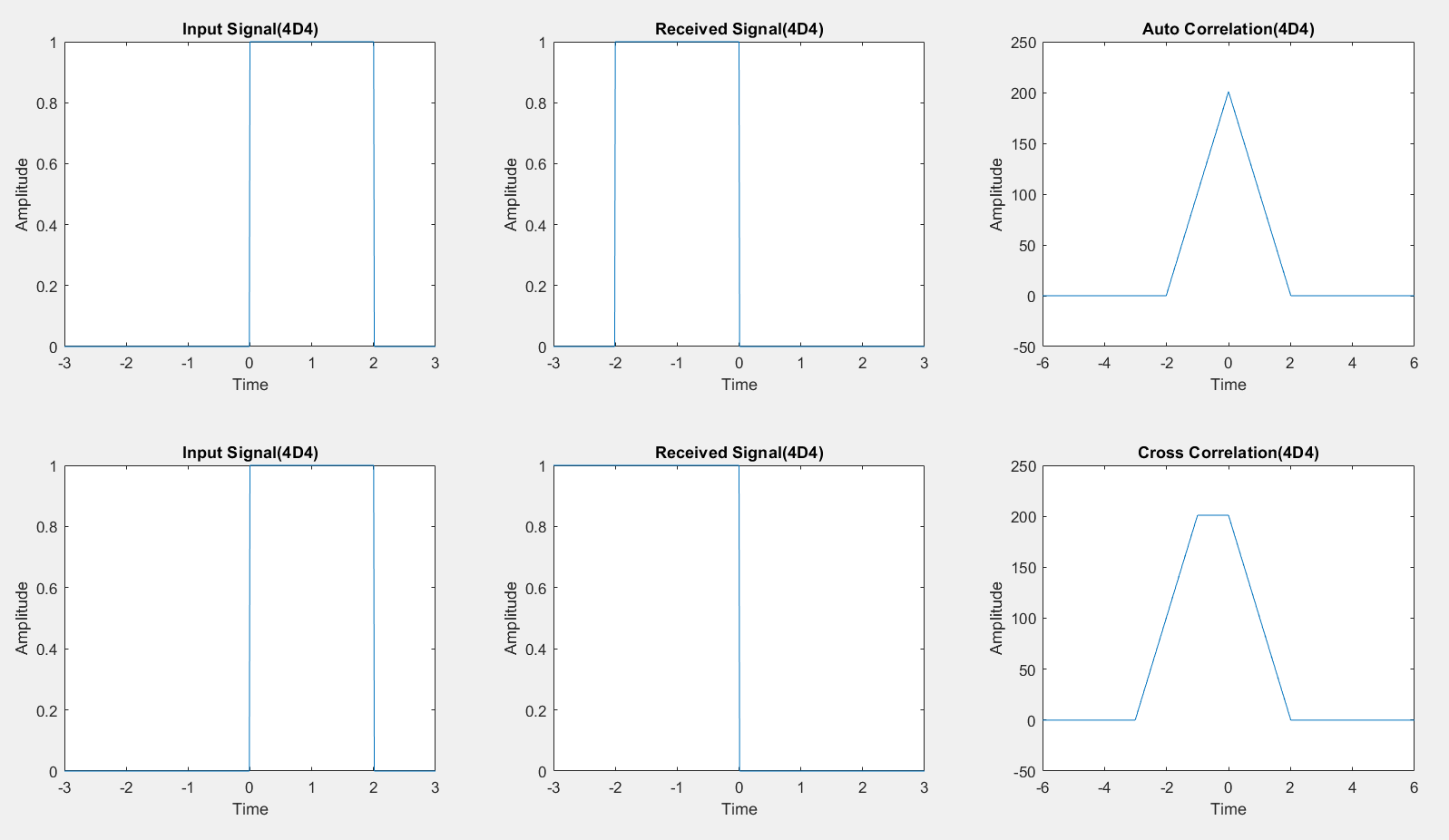
ylabel("Amplitude");

title("Cross Correlation(4D4)");

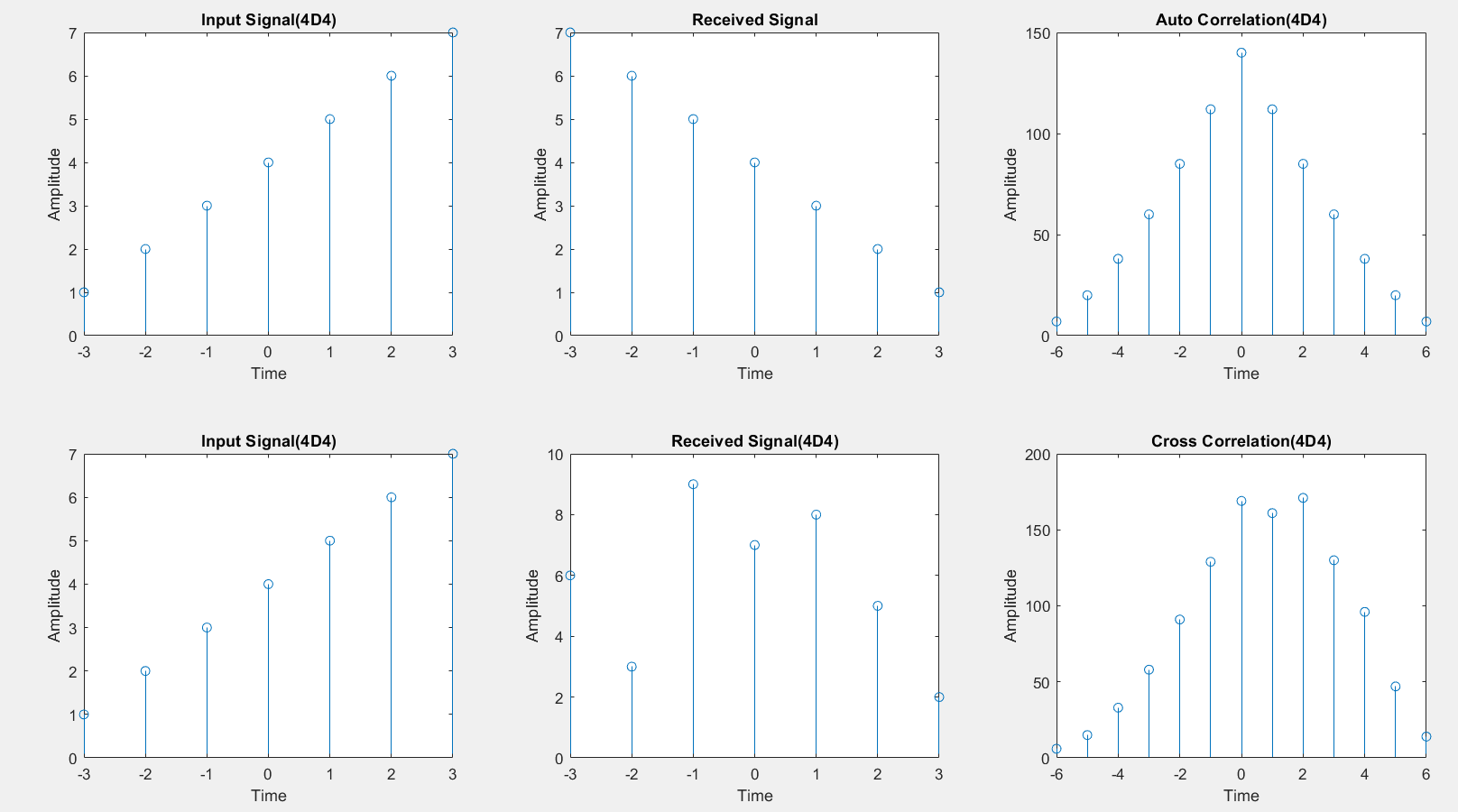
**LAB PROCEDURE:**

1. Open the MATLAB® software by double clicking its icon.
2. MATLAB®logo will appear and after few moments Command Prompt will appear.
3. Go to the File Menu and select a New M‐file. (File New M‐file) or in the left-hand corner a blank white paper icon will be there. Click it once.
4. A blank M‐file will appear with a title ‘untitled’
5. Now start typing your program. After completing, save the M‐file with appropriate name. To execute the program Press F5 or go to Debug Menu and select Run.
6. After execution output will appear in the Command window. If there is an error then with an alarm, type of error will appear in red color.
7. Rectify the error if any and go to Debug Menu and select Run.

**OUTPUT WAVEFORMS /GRAPHS:**

****

Continuous Time Signal



Discrete Time Signal

**RESULTS**:

Successfully Implemented Finding the Convolution between (i) Signals (ii) Sequences

**Experiment No. 09**

**Aim**: Verification of Linearity and Time Invariance Properties of a given Continuous/Discrete System

S**oftware used:** MATLAB.

**MATLAB CODE**:

**Linearity:**

N=5;

x1=[1,2,-1,-3,2];

x2=[2,-1,3,-4,3];

a1=2;

a2=3;

n=0:1:N-1;

x3=a1\*x1+a2\*x2;

y01=n.\*(x3);

y1=n.\*(x1);

y2=n.\*(x2);

y02=a1\*y1+a2\*y2;

disp(' the otput sequence y01 is :');

disp(y01);

disp('the output sequence y02 is:');

disp(y02);

if(y01==y02)

disp('y01==y02 .hence the system is linear');

else

disp('y01~=y02 .hence the system is non linear');

end

subplot(3,2,1);

stem(n,x1);

xlabel("time");

ylabel("amplitude");

title(" 1st input signal(4D4)");

subplot(3,2,2);

stem(n,x2);

xlabel("time");

ylabel("amplitude");

title(" 2nd input signal(4D4)");

subplot(3,2,3);

stem(n,y1);

xlabel("time");

ylabel("amplitude");

title(" 1s input response(4D4)");

subplot(3,2,4);

stem(n,y1);

xlabel("time");

ylabel("amplitude");

title(" 2nd input signal(4D4)");

subplot(3,2,5);

stem(n,y01);

xlabel("time");

ylabel("amplitude");

title(" total weighted sum of response(4D4)");

subplot(3,2,6);

stem(n,y02);

xlabel("time");

ylabel("amplitude");

title(" weighted sum of individual response(4D4)");

**Time Invariance:**

x=[1,2,-1,3,-2];

n=0:length(x)-1;

d=2;

y=n.\*(x.^2);

xd=[zeros(1,d),x];

disp(length(n));

nd=0:length(xd)-1;

xp=[x,zeros(1,d)];

yp=nd.\*(xd.^2);

disp(' the otput sequence yp is :');

disp(yp);

yd=[zeros(1,d),y];

disp(' the otput sequence yd is :');

disp(yd);

if(yp==yd)

disp('y1==y2. hence the system is time inveriant');

else

disp('y1~=y2 . hence the system is time veriant');

end

subplot(3,2,1);

stem(n,x);

xlabel("time");

ylabel("amplitude");

title(" input sequence(4D4)");

subplot(3,2,2);

stem(n,y);

xlabel("time");

ylabel("amplitude");

title(" output sequence(4D4)");

subplot(3,2,3);

stem(nd,xd);

xlabel("time");

ylabel("amplitude");

title(" input delay by 2 units(4D4)");

subplot(3,2,4);

stem(nd,xp);

xlabel("time");

ylabel("amplitude");

title(" dummy");

subplot(3,2,5);

stem(nd,yp);

xlabel("time");

ylabel("amplitude");

title(" response to be input delay(4D4)");

subplot(3,2,6);

stem(nd,yd);

xlabel("time");

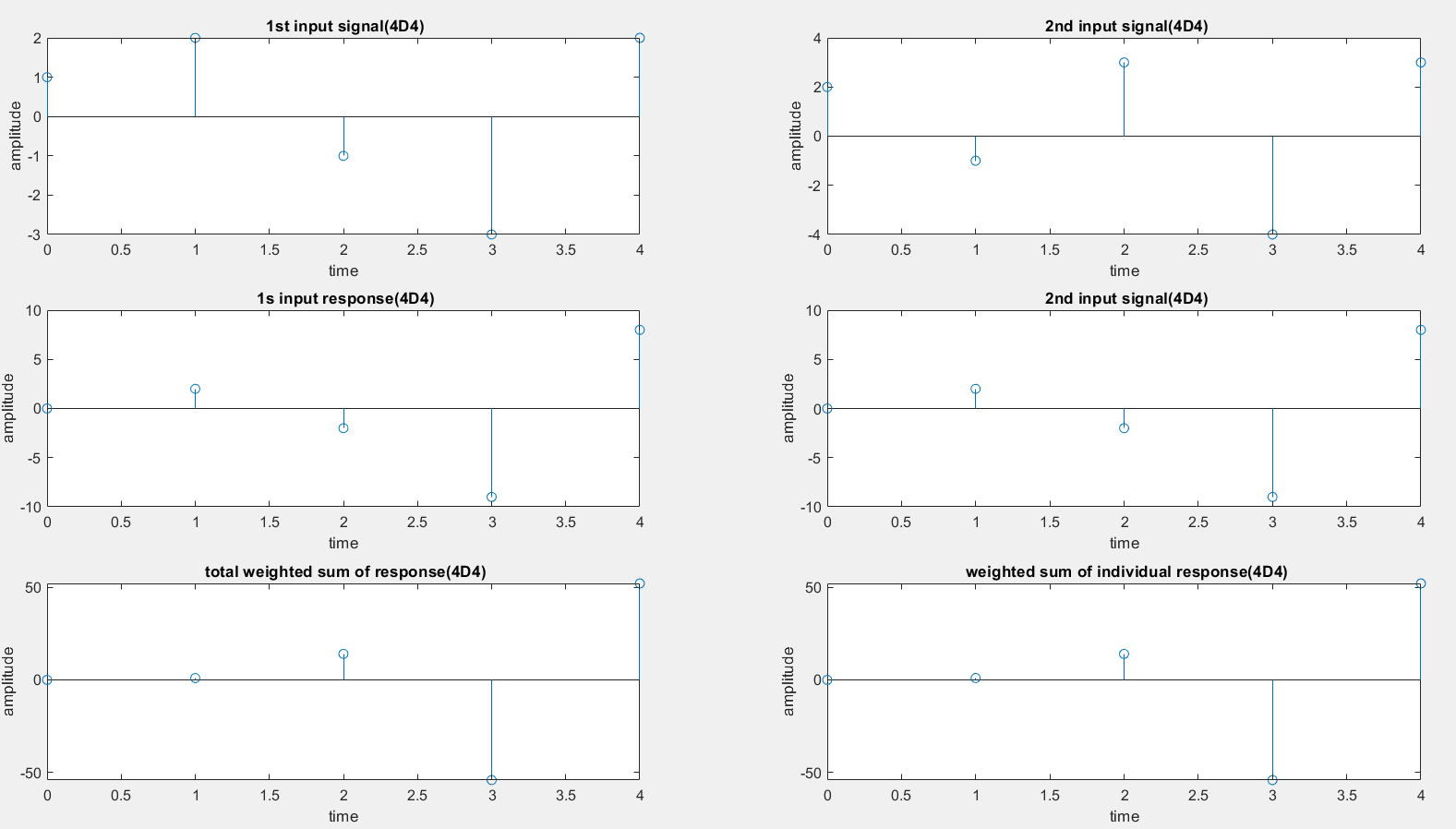
ylabel("amplitude");

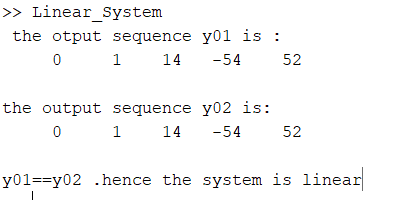
title(" delayed output response(4D4)");

**LAB PROCEDURE:**

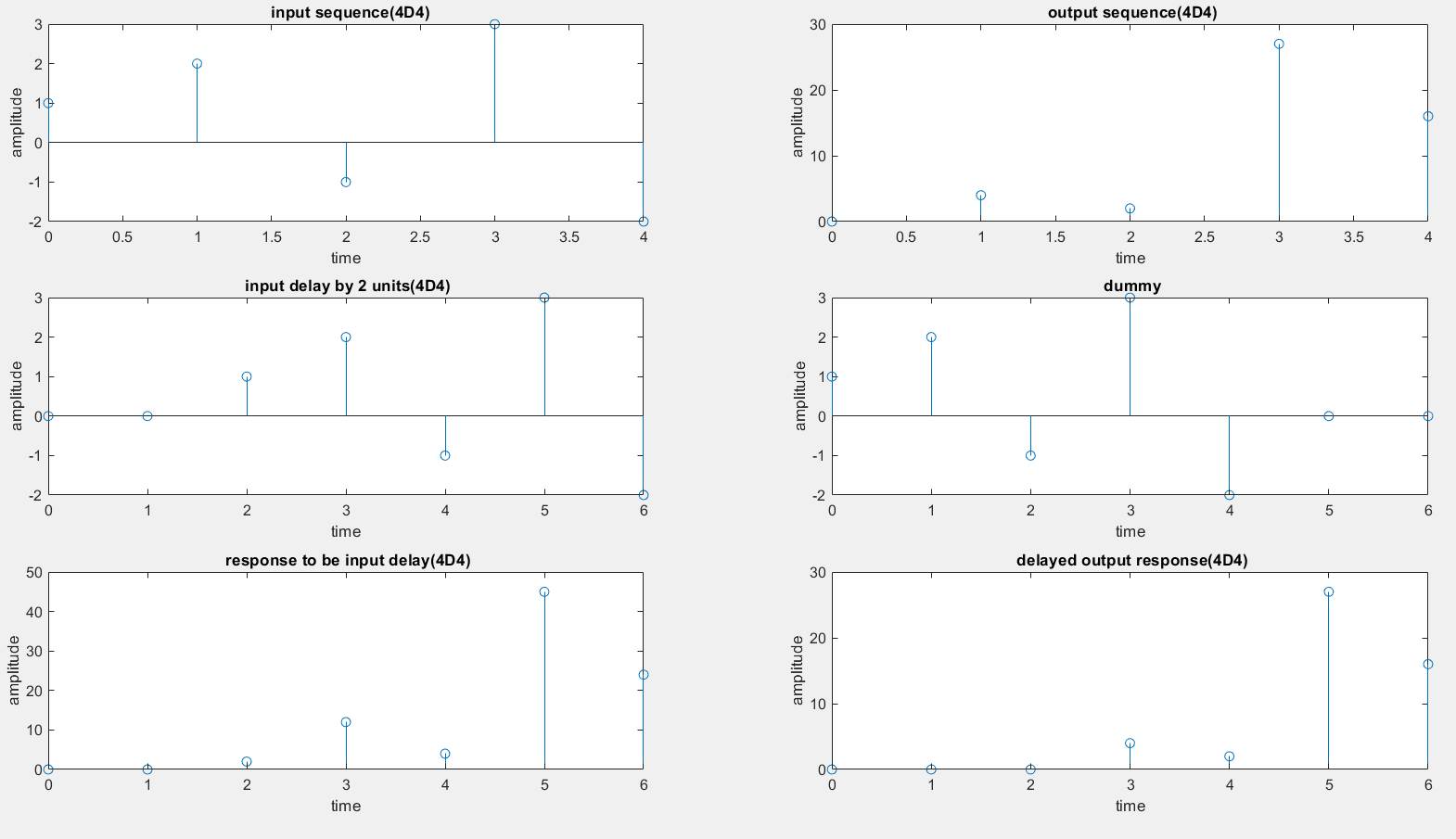
1. Open the MATLAB® software by double clicking its icon.
2. MATLAB®logo will appear and after few moments Command Prompt will appear.
3. Go to the File Menu and select a New M‐file. (File New M‐file) or in the left-hand corner a blank white paper icon will be there. Click it once.
4. A blank M‐file will appear with a title ‘untitled’
5. Now start typing your program. After completing, save the M‐file with appropriate name. To execute the program Press F5 or go to Debug Menu and select Run.
6. After execution output will appear in the Command window. If there is an error then with an alarm, type of error will appear in red color.
7. Rectify the error if any and go to Debug Menu and select Run.

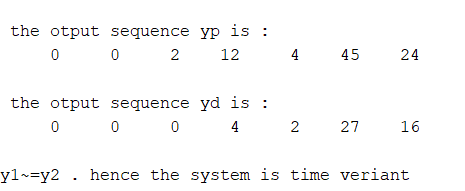
**OUTPUT WAVEFORMS /GRAPHS:**

****

****

Linearity of a System





Time Invariance of a System

**RESULTS**:

Successfully Verified the Linearity and Time Invariance Properties of a given Continuous/Discrete System

**Experiment No. 10**

**Aim**: Computation of Unit sample, Unit step and sinusoidal responses of the given LTI system and Verifying its Physical realizability and stability properties.

S**oftware used:** MATLAB.

**MATLAB CODE**:

close all

clear all

b = [1];

a = [1 -1 0.9];

n = 0:3:100;

x1 = 1.\*(n==0)+0.\*(n~=0);

y1 = filter(b,a,x1);

subplot(3,2,1);

stem(n,x1);

xlabel("Time");

ylabel("Amplitude");

title("Unit Impulse(4D4)");

subplot(3,2,2);

stem(n,y1);

xlabel("Time");

ylabel("Amplitude");

title("Response of Unit Impulse(4D4)");

x2 = 1.\*(n>=0)+0.\*(n<0);

y2 = filter(b,a,x2);

subplot(3,2,3);

stem(n,x2);

xlabel("Time");

ylabel("Amplitude");

title("Unit Step(4D4)");

subplot(3,2,4);

stem(n,y2);

xlabel("Time");

ylabel("Amplitude");

title("Response of Unit Step(4D4)");

n = 0:1:8\*pi;

x3 = sin(n);

y3 = filter(b,a,x3);

subplot(3,2,5);

stem(n,x3);

xlabel("Time");

ylabel("Amplitude");

title("Sinusoidal(4D4)");

subplot(3,2,6);

stem(n,y3);

xlabel("Time");

ylabel("Amplitude");

title("Response of Sinusoidal(4D4)");

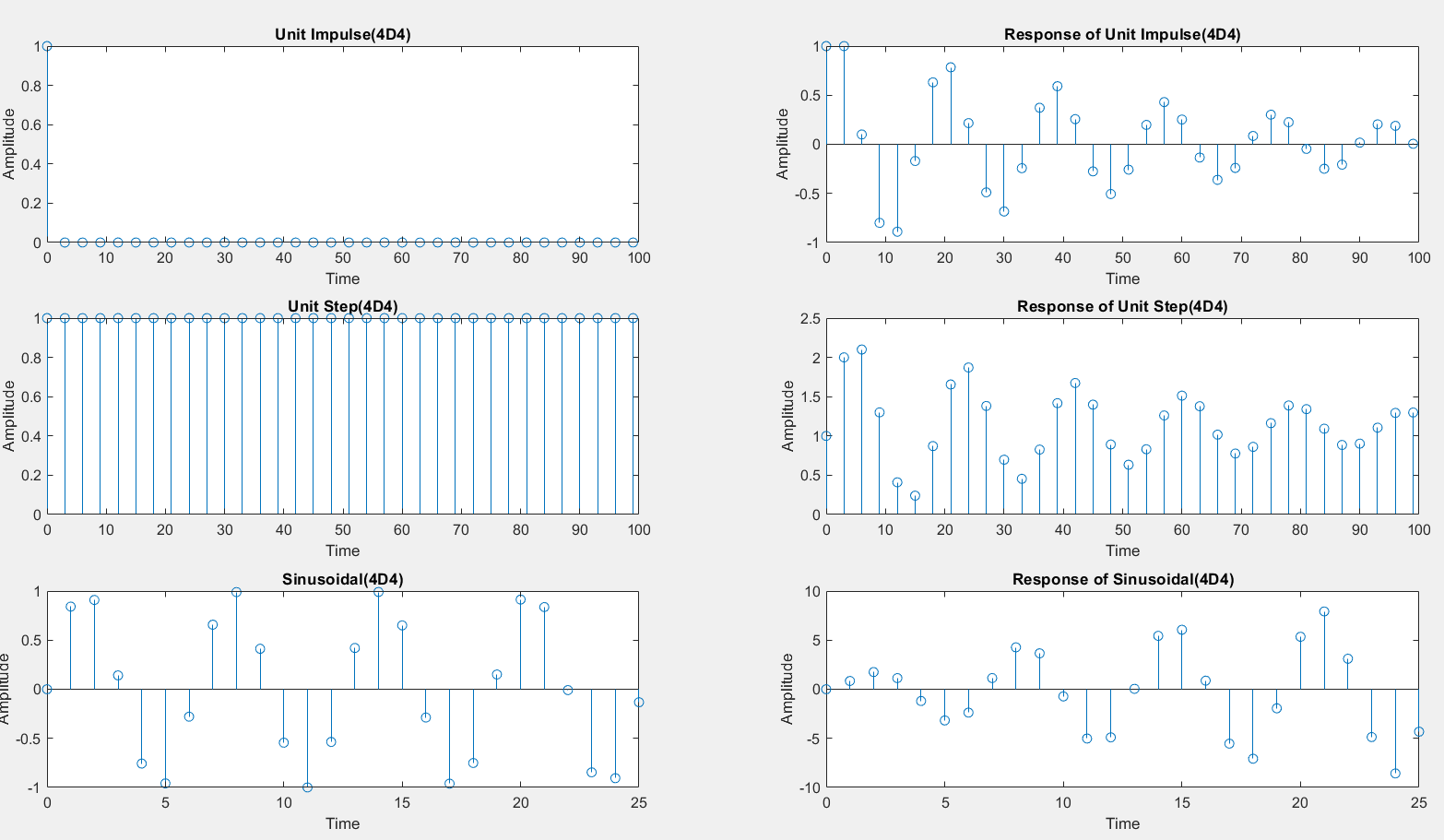
figure;

zplane(b,a);

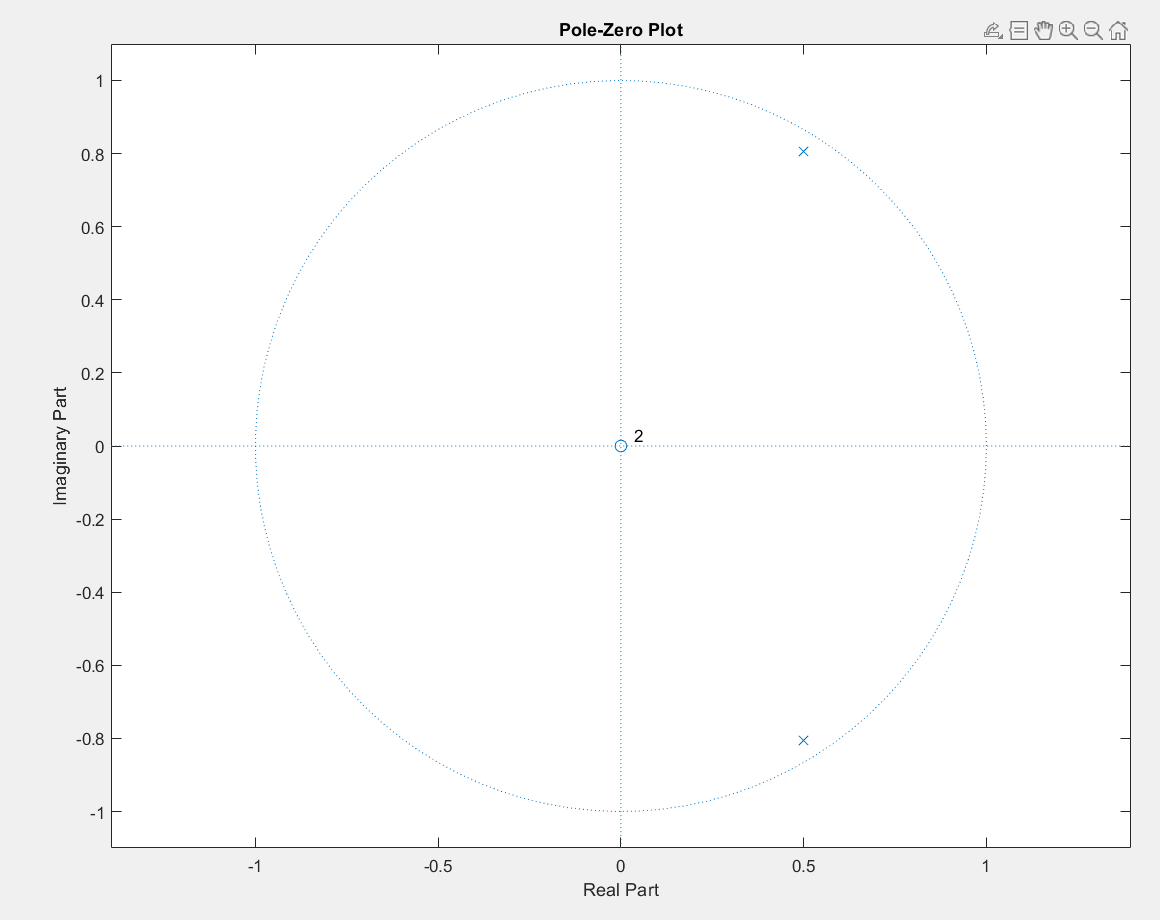
**LAB PROCEDURE:**

1. Open the MATLAB® software by double clicking its icon.
2. MATLAB®logo will appear and after few moments Command Prompt will appear.
3. Go to the File Menu and select a New M‐file. (File New M‐file) or in the left-hand corner a blank white paper icon will be there. Click it once.
4. A blank M‐file will appear with a title ‘untitled’
5. Now start typing your program. After completing, save the M‐file with appropriate name. To execute the program Press F5 or go to Debug Menu and select Run.
6. After execution output will appear in the Command window. If there is an error then with an alarm, type of error will appear in red color.
7. Rectify the error if any and go to Debug Menu and select Run.

**OUTPUT WAVEFORMS /GRAPHS:**

****

Responses of a LTI System for various I/P Signals



Z-Plane

**RESULTS**:

Successfully Implemented Computation of Unit sample, Unit step and sinusoidal responses of the given LTI system and Verifying its Physical realizability and stability properties.

**Experiment No. 11**

**Aim**: Verifying the applications of Correlation: Removal of noise by Autocorrelation & Cross correlation.

S**oftware used:** MATLAB.

**MATLAB CODE**:

**Auto Correlation:**

close all

clear all

t = 0:0.1:8\*pi;

s = sin(t);

n = randn([1 252]);

f = s+n;

subplot(3,2,1);

plot(s);

xlabel("Time");

ylabel("Amplitude");

title("Input Signal(4D4)");

subplot(3,2,2);

plot(f);

xlabel("Time");

ylabel("Amplitude");

title("Combined Signal(4D4)");

Rs = xcorr(s,s);

Rn = xcorr(n,n);

Rf = xcorr(f,f);

R = Rs+Rn;

subplot(3,2,3);

plot(Rs);

xlabel("Time");

ylabel("Amplitude");

title("Auto Correlation Rsc(4D4)");

subplot(3,2,4);

plot(Rn);

xlabel("Time");

ylabel("Amplitude");

title("Auto Correlation Rnc(4D4)");

subplot(3,2,5);

plot(Rf);

xlabel("Time");

ylabel("Amplitude");

title("Auto Correlation Rfc(4D4)");

subplot(3,2,6);

plot(R);

xlabel("Time");

ylabel("Amplitude");

title("Auto Correlation R(4D4)");

**Cross Correlation:**

close all

clear all

t = 0:0.1:8\*pi;

s = sin(t);

n = randn([1 252]);

f = s+n;

c = cos(t);

subplot(4,2,1);

plot(s);

xlabel("Time");

ylabel("Amplitude");

title("Input Signal(4D4)");

subplot(4,2,2);

plot(f);

xlabel("Time");

ylabel("Amplitude");

title("Combined Signal(4D4)");

subplot(4,2,3);

plot(c);

xlabel("Time");

ylabel("Amplitude");

title("Constant Signal(4D4)");

Rsc = xcorr(s,c);

Rnc = xcorr(n,c);

Rfc = xcorr(f,c);

R = Rsc+Rnc;

subplot(4,2,4);

plot(Rsc);

xlabel("Time");

ylabel("Amplitude");

title("Cross Correlation Rsc(4D4)");

subplot(4,2,5);

plot(Rnc);

xlabel("Time");

ylabel("Amplitude");

title("Cross Correlation Rnc(4D4)");

subplot(4,2,6);

plot(Rfc);

xlabel("Time");

ylabel("Amplitude");

title("Cross Correlation Rfc(4D4)");

subplot(4,2,7);

plot(R);

xlabel("Time");

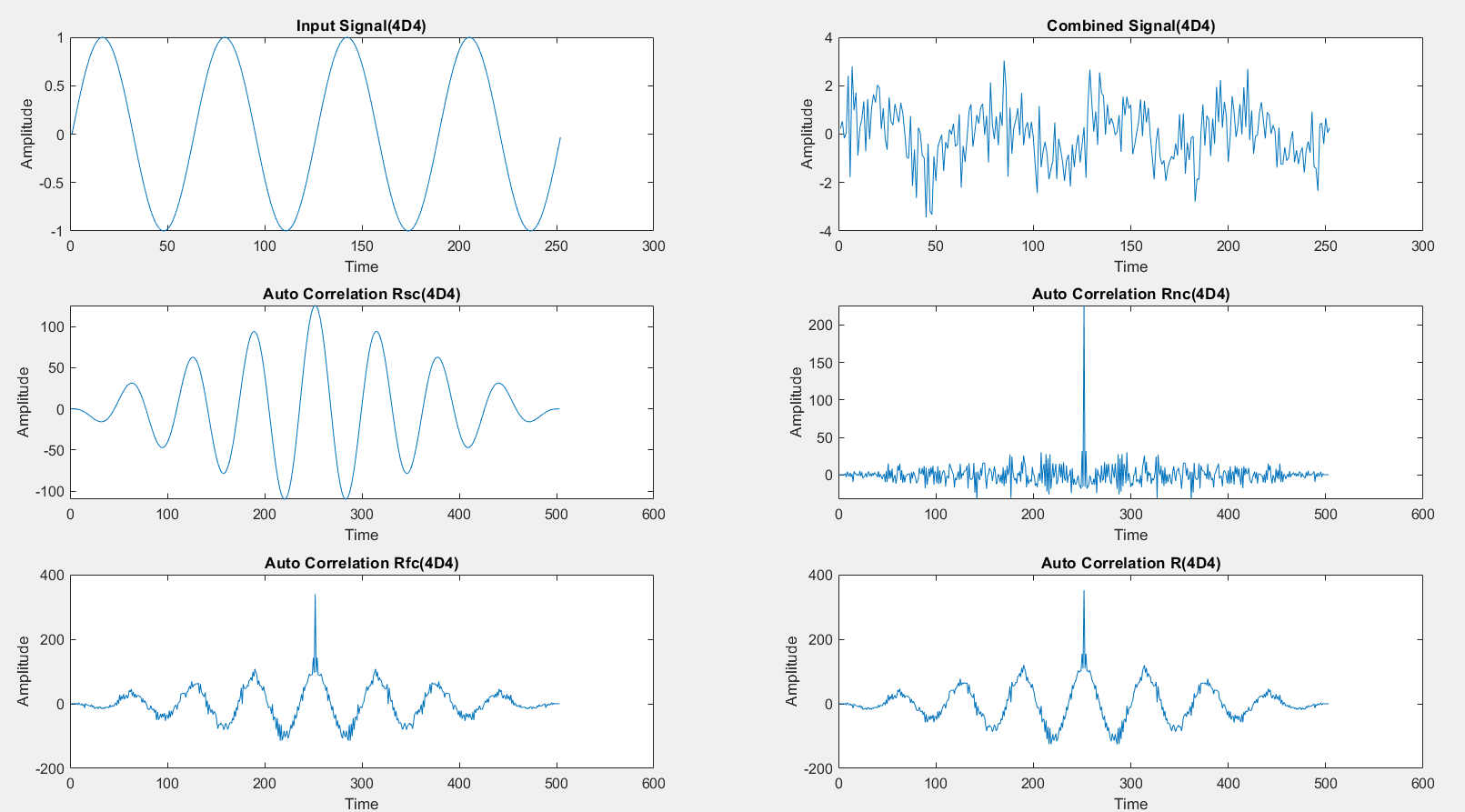
ylabel("Amplitude");

title("Cross Correlation R(4D4)");

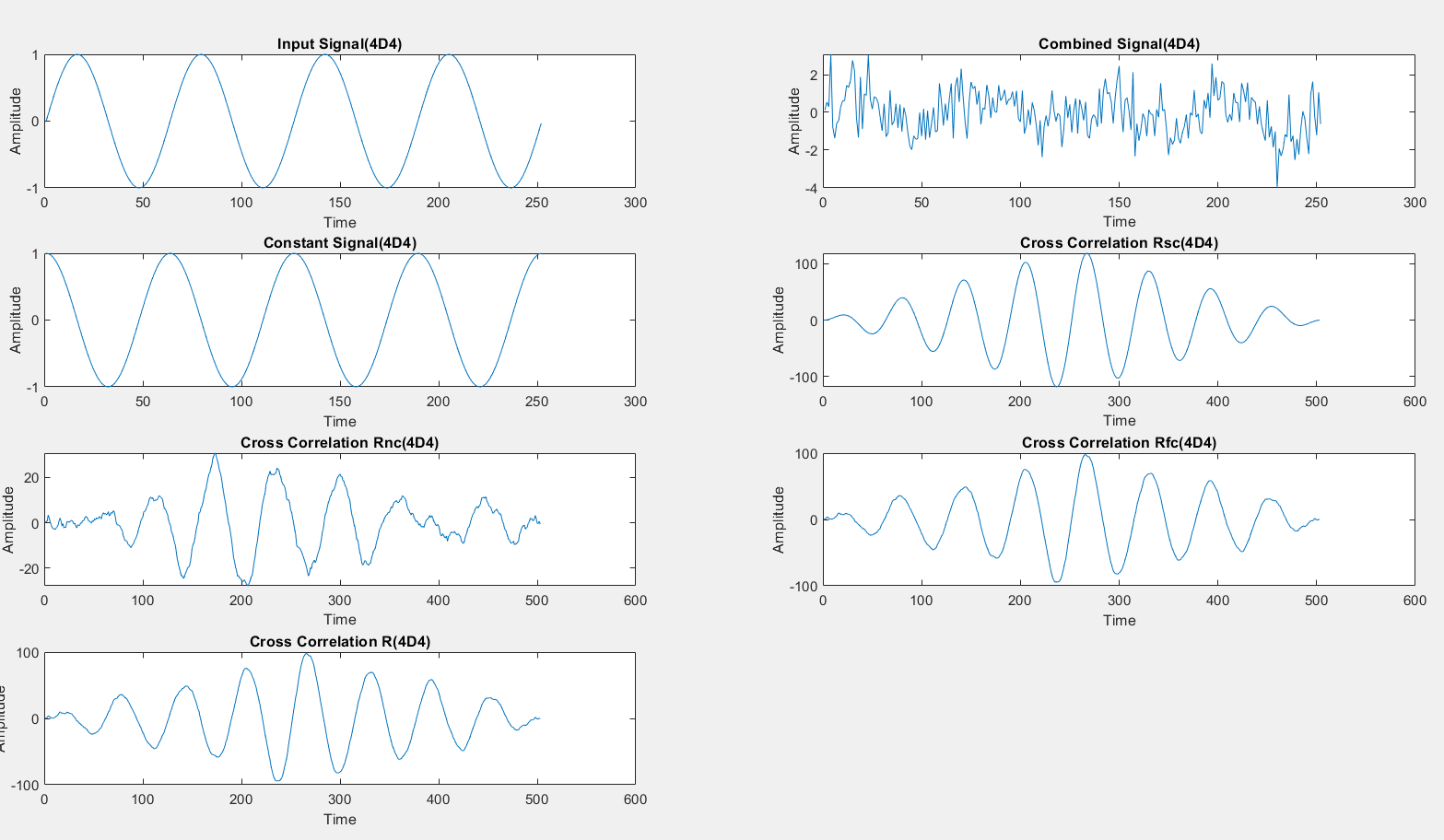
**LAB PROCEDURE:**

1. Open the MATLAB® software by double clicking its icon.
2. MATLAB®logo will appear and after few moments Command Prompt will appear.
3. Go to the File Menu and select a New M‐file. (File New M‐file) or in the left-hand corner a blank white paper icon will be there. Click it once.
4. A blank M‐file will appear with a title ‘untitled’
5. Now start typing your program. After completing, save the M‐file with appropriate name. To execute the program Press F5 or go to Debug Menu and select Run.
6. After execution output will appear in the Command window. If there is an error then with an alarm, type of error will appear in red color.
7. Rectify the error if any and go to Debug Menu and select Run.

**OUTPUT WAVEFORMS /GRAPHS:**

****

Auto Correlation



Cross Correlation

**RESULTS**:

Successfully verified the applications of Correlation: Removal of noise by Autocorrelation and Cross correlation.