Digital Signal Processing

Teacher: Rakhi Narang

Name : Manya Kaushik
Roll No : 1620008
Course : B.Sc. (H) Electronics
Semester : V

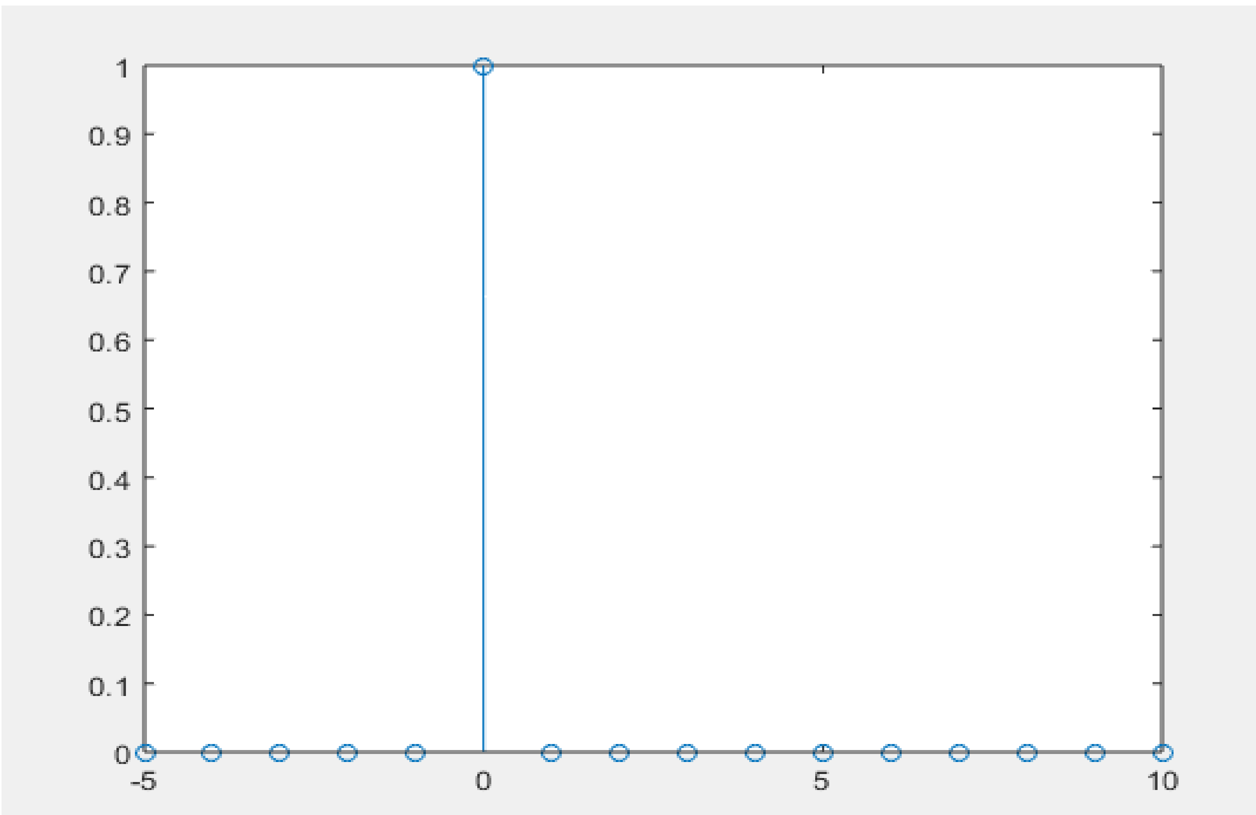
Serial no.	List of experiments
1	Generation of function and sequence
2	Program to generate to 50hz C.T sinusoidal signal
3	Generation of shifting scaling and reversal
4	Generation of odd and even signal
5	Timescaling (downSampling and upsampling)
6	Convolution of DT signal
7	Generate and plot discrete time sequence in a given interval
8	Obtain partial fraction and plot zero pole diagram
9	Z- transform Of various signal
10	Finding inverse of Z-transform
11	Application of convolution and deconvolution in z-transform
12	Deconvolution in z-transform
13	Program on DTFT
14	Program on DFT
15	Program on IDFT
16	Program on circular convolution
17	Program on FFT

<u>Experiment – 01</u>

Aim – Generation of unit impulse unit step, ramp function, discrete time sequence, real exponential ,real sinusoidal sequence.

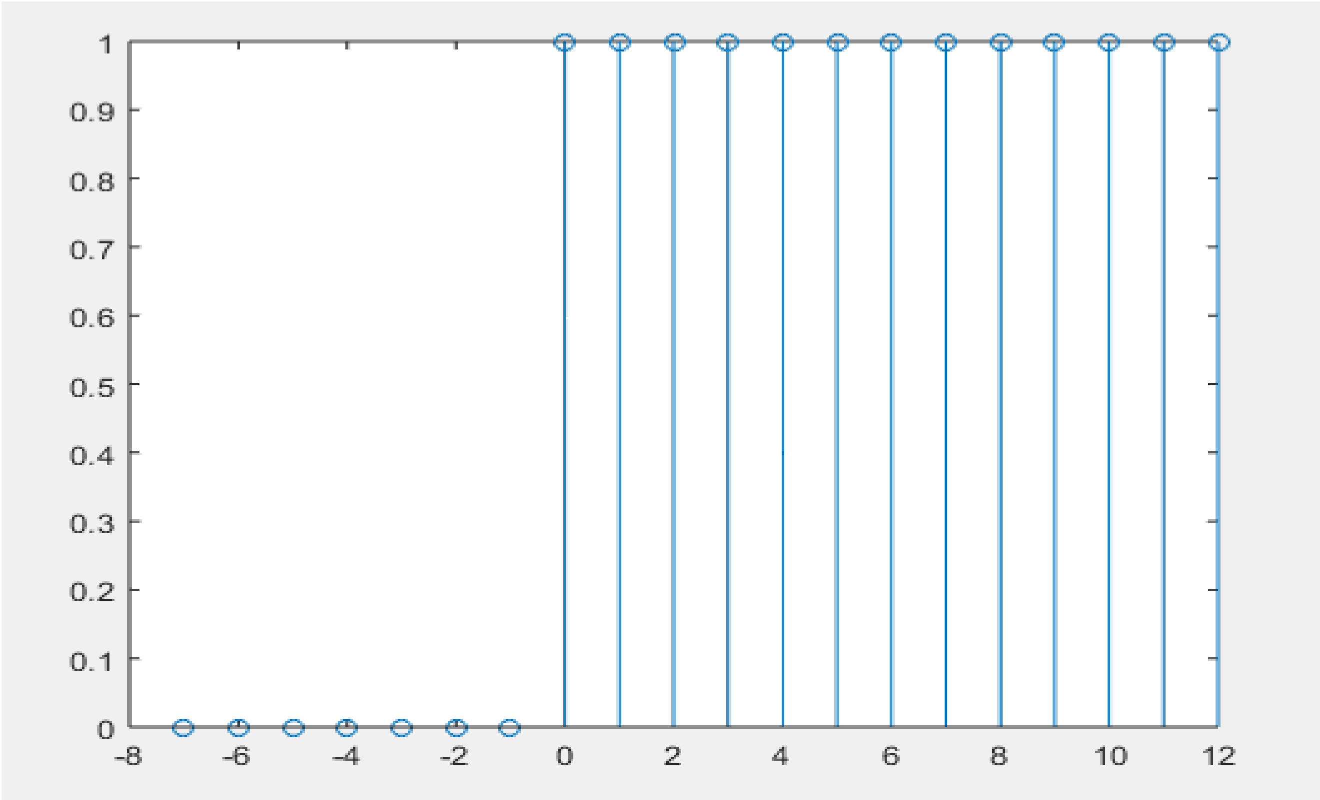
Unit impulse.

```
n0=0
n1=-5
n2=10
n=[n1:n2]
x=[(n-n0)==0]
stem(n,x)
```



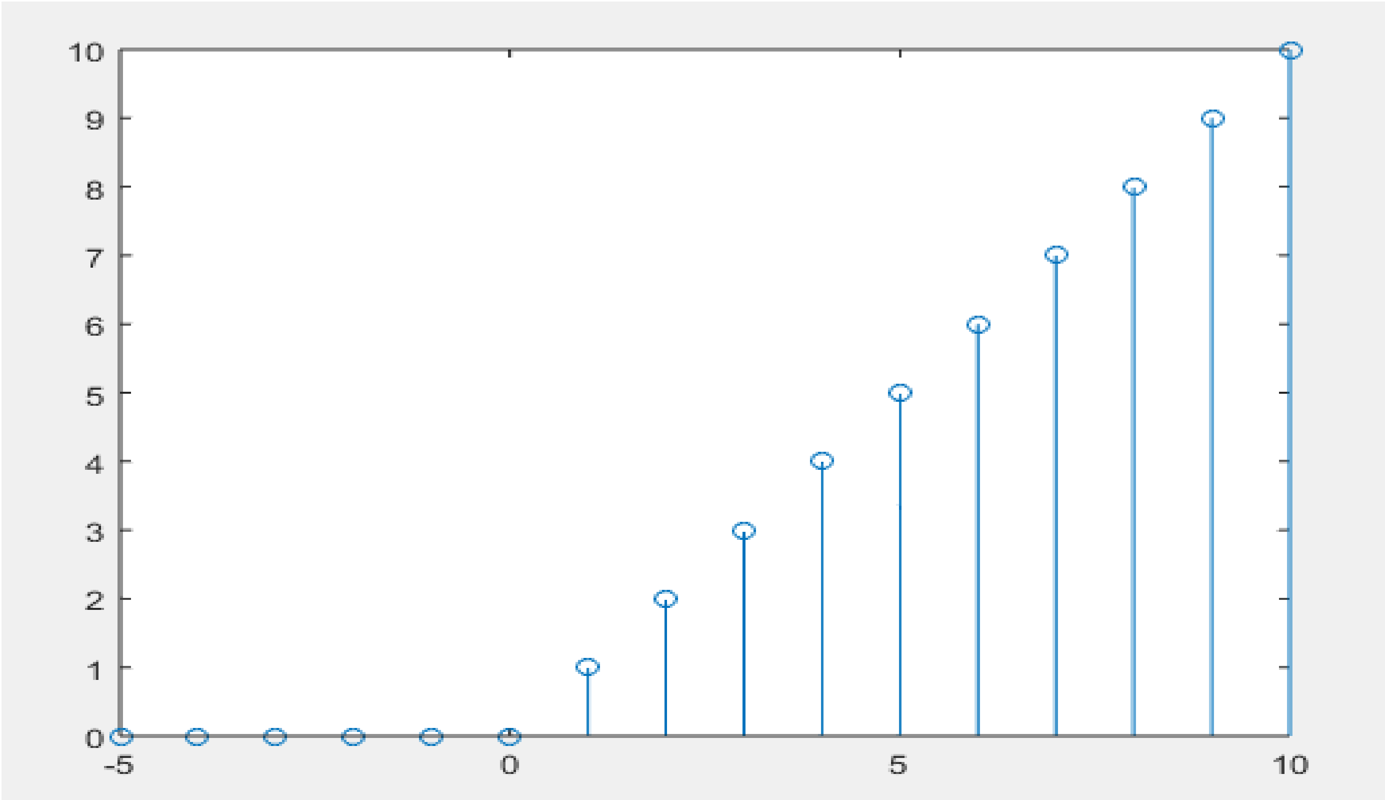
Unit step.

```
n0=0
n1=-7
n2=12
n=[n1:n2]
x=[(n-n0)>=0]
stem(n,x)
```



Unit ramp.

```
n0=0
n1=-5
n2=10
n=[n1:n2]
x=[n>=0]
ramp=n.*x
stem(n,ramp)
```

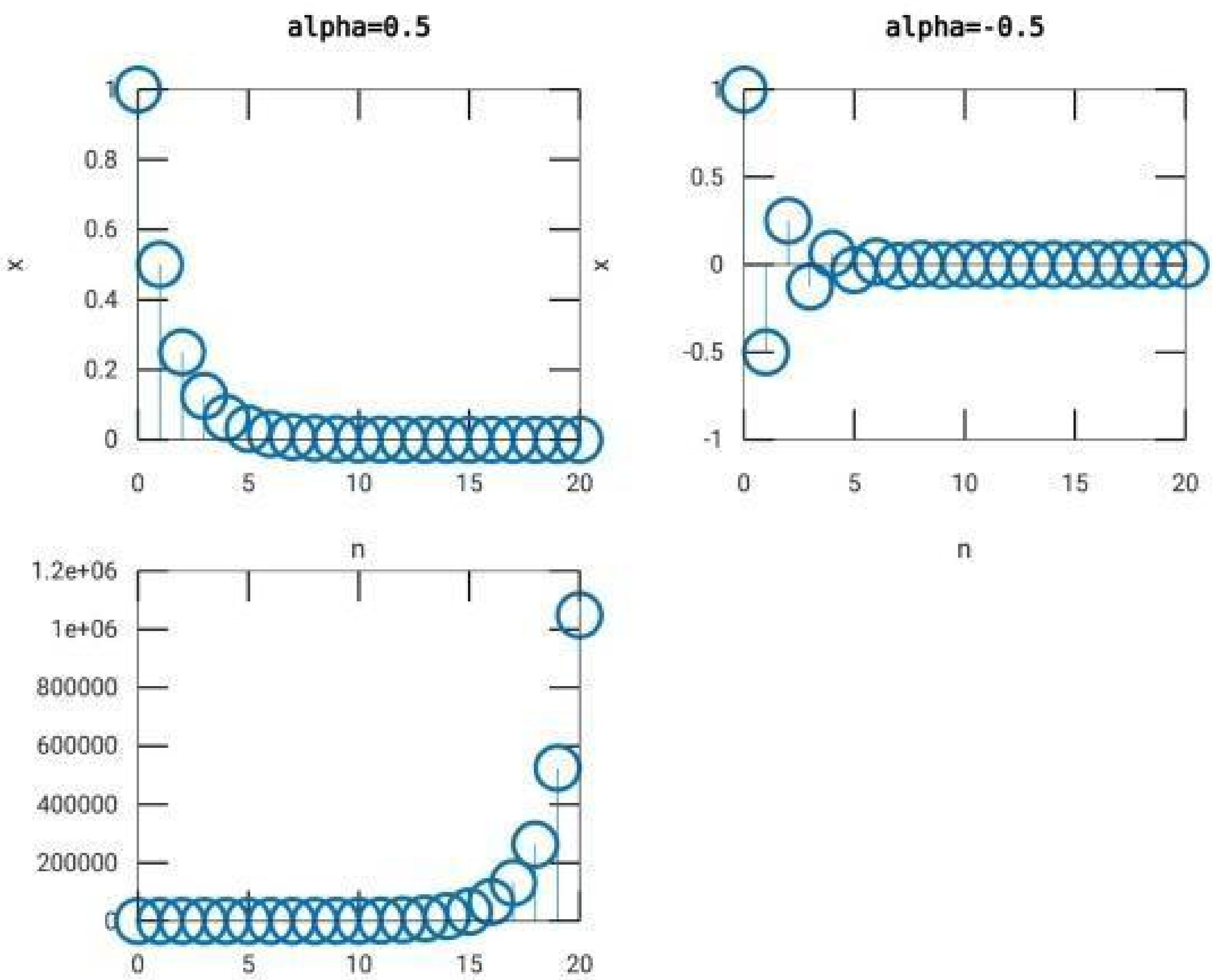


Real Exponential sequence=

Code(DT)-

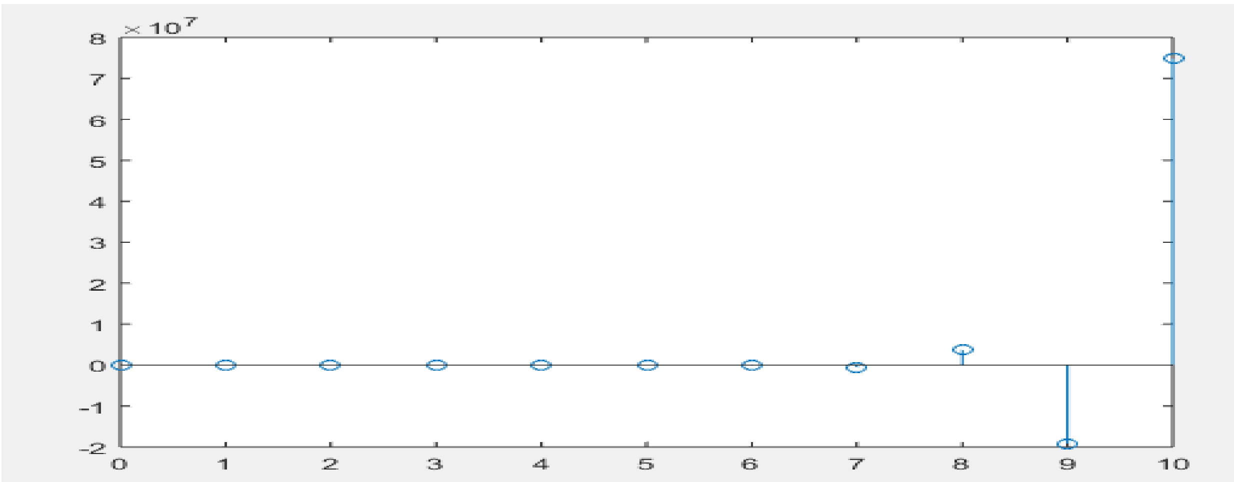
```
function[n,x]=realexp(n1,n2)
n=[n1:n2];
x1=0.5.^n;
subplot(2,2,1);
stem(n,x1);
title('alpha=0.5');
xlabel('n');
ylabel('x');
x2=(-0.5).^n;
subplot(2,2,2);
stem(n,x2);
```

```
title('alpha=-0.5');
xlabel('n');
ylabel('x');
x3=2.^n;
subplot(2,2,3);
stem(n,x3);
title('alpha=2');
xlabel('n');
ylabel('x');
end
```



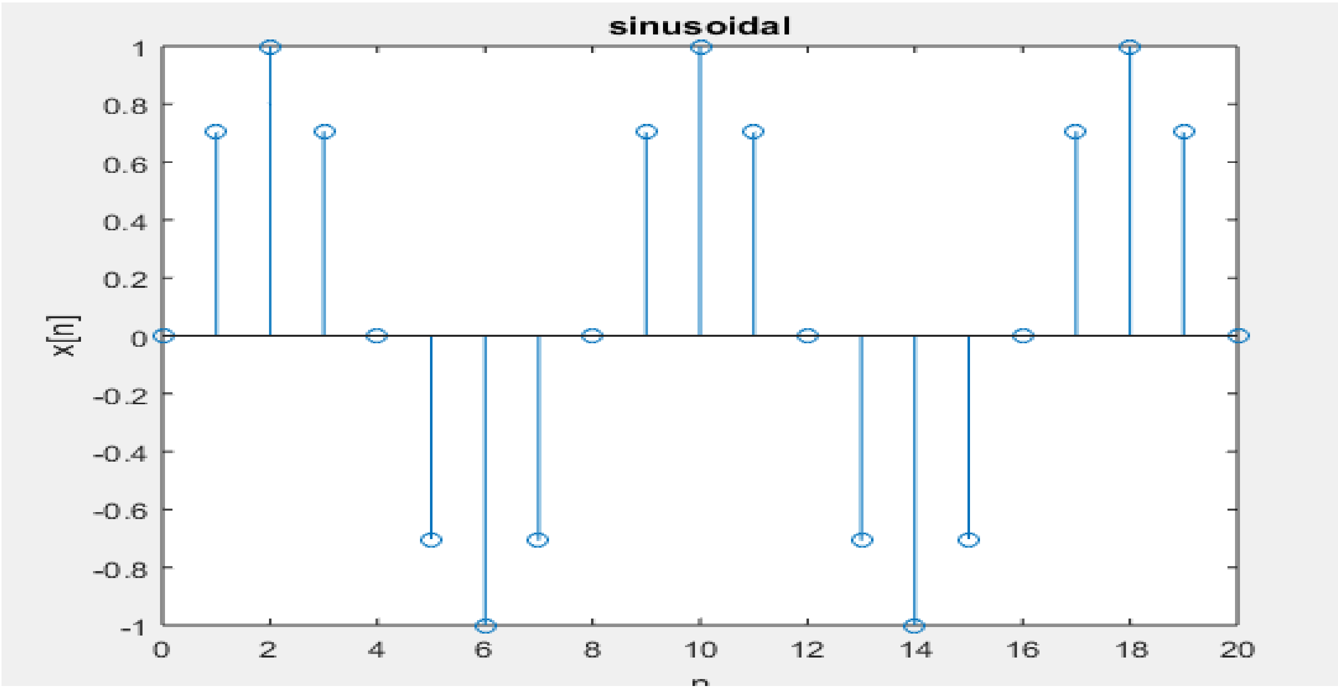
complex valued function.

```
n=[0:10];

x=exp((2+3*j)*n);

stem(n,x);
```



Real sinusoidal sequence.

```
n=[0:20];
x=sin(0.25*pi*n);
stem(n,x);
title('sinusoidal');
xlabel('n');
ylabel('x[n]');
```

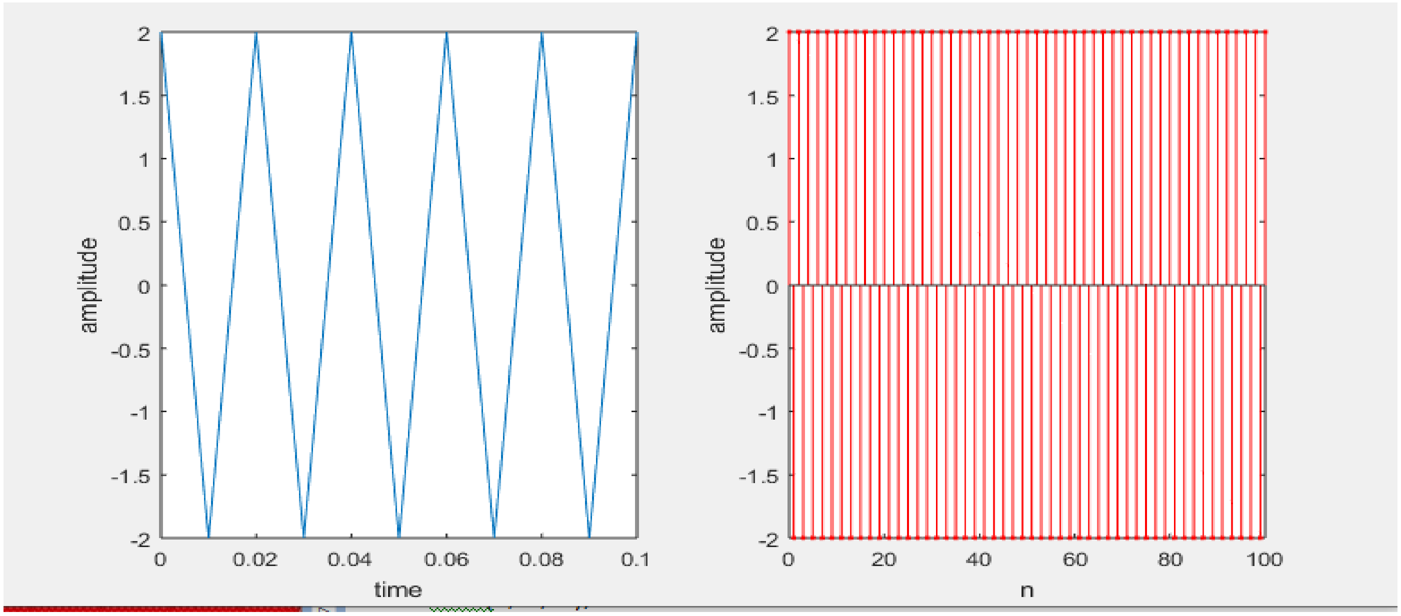

sinusoidal
x[n]
n
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
2
4
6
8
10
12
14
16
18
20

Experiment – 02

Aim – write a program to generate to 50Hz continuous time sinusoidal signal xt=A*cos(2*pi*50*t),and its sampled version (nTs),where f=50 and A= 2 and sampling frequency fs=(100 hz).plot the signal using plot and stem command.

Code –

```
A=2;
fs=100;
T=1/fs;
t=0:T:0.1;
xt=A*cos(2*pi*50*t);
subplot(1,2,1);
plot(t,xt);
xlabel('time');
ylabel('amplitude');
n=0:100;
x1=A*cos(2*pi*50*n*T);
subplot(1,2,2);
stem(n,x1,'.r');
xlabel('n');
ylabel('amplitude');
```

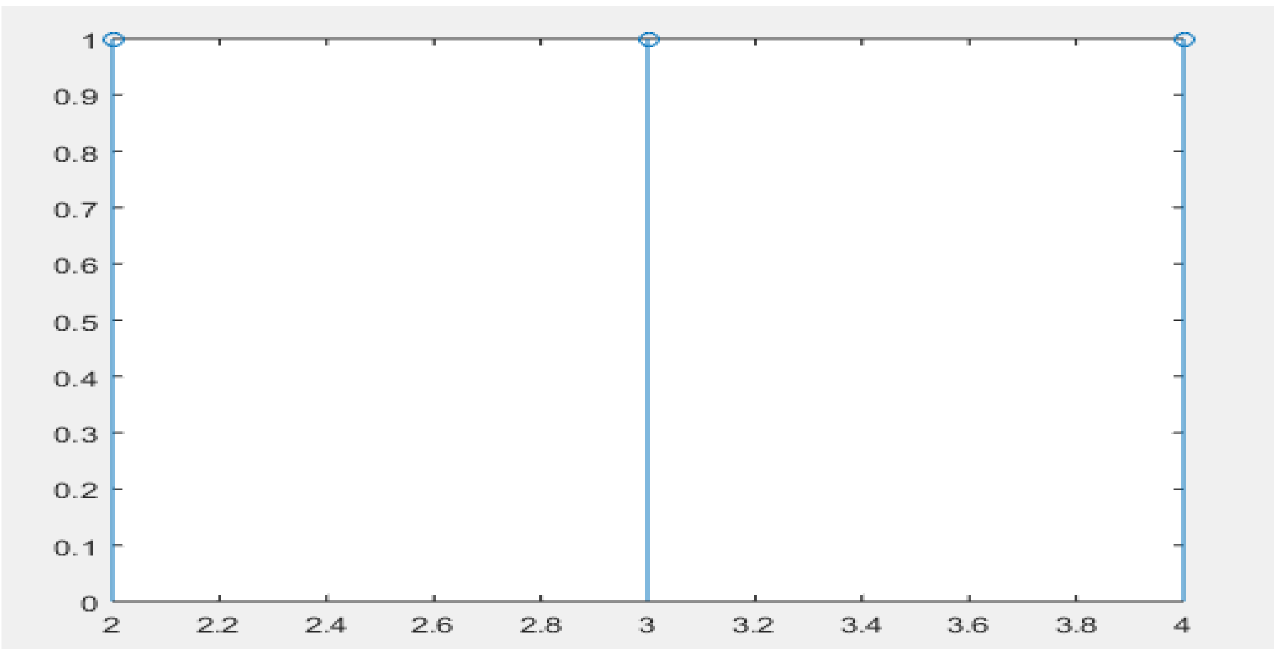

amplitude
time
0
0.02
0.04
0.06
0.08
0.1
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
amplitude
n
0
20
40
60
80
100

Experiment – 03

Aim – Generation of shifting ,scaling and reversal .

Shifting -

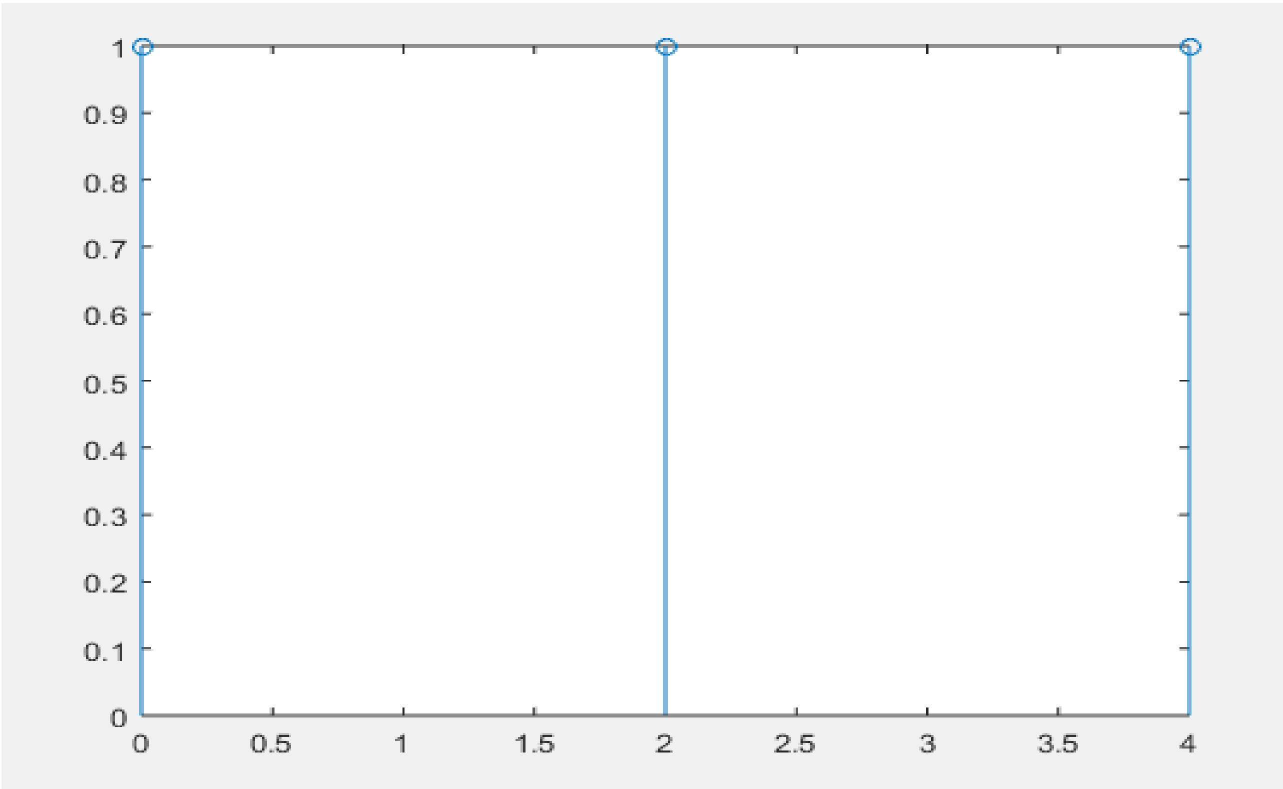
```
function[y,n]=sighift(x,m,k)
n=m+k;
y=x;
stem(n,y)
end
```



Scaling -

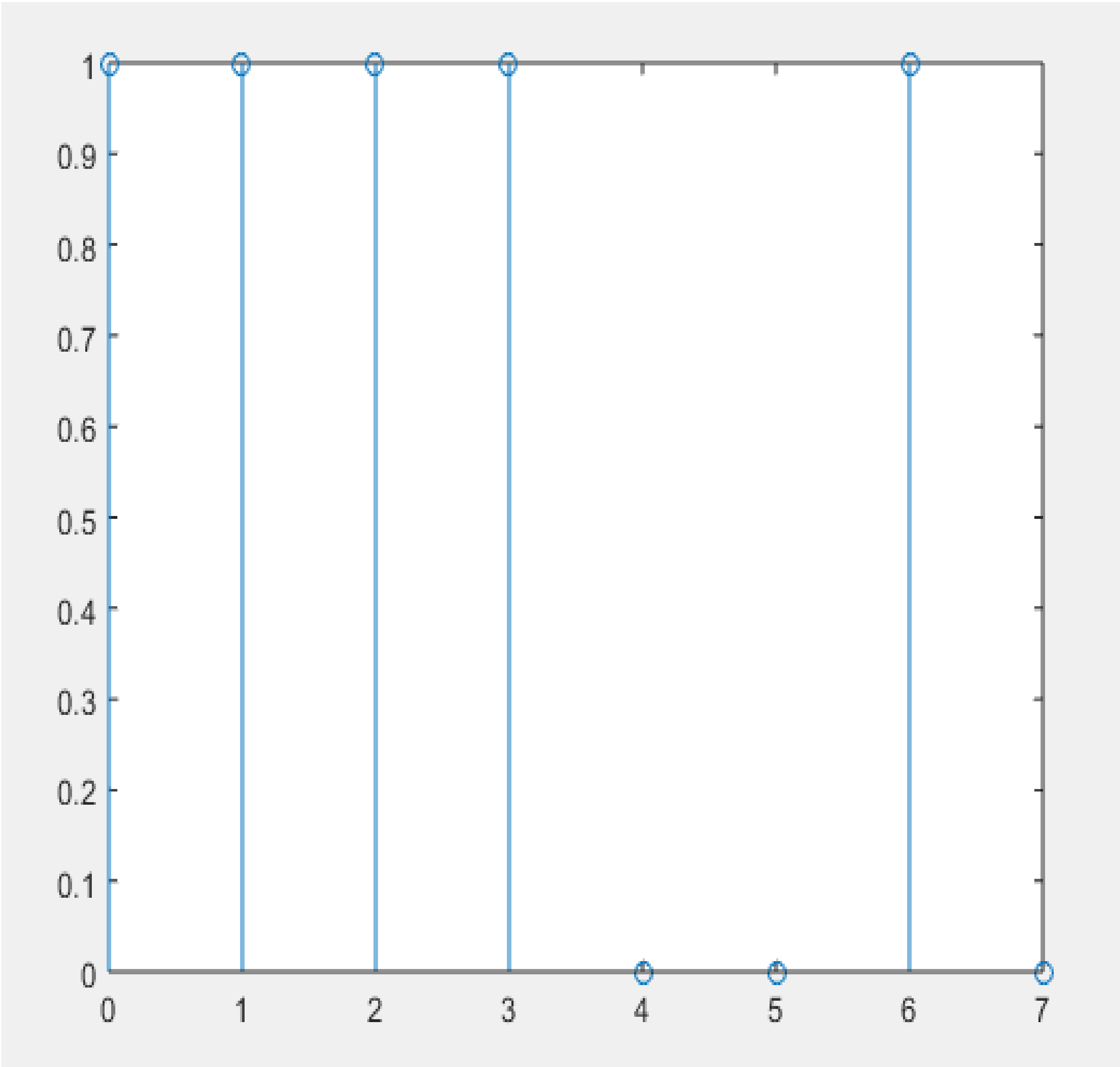
```
function[y,n]=sigscale(x,m,k)
y=x
n=m*k;
stem(n,y);
end


>> sigscale([1,1,1],[0:2],2)

y =

     1     1     1
```

```
ans =

     1     1     1
```



Reversal -

```
function[y,n]=sigfold(x,n)
y=fliplr(x)
n=fliplr(n);
stem(n,y);
end
```



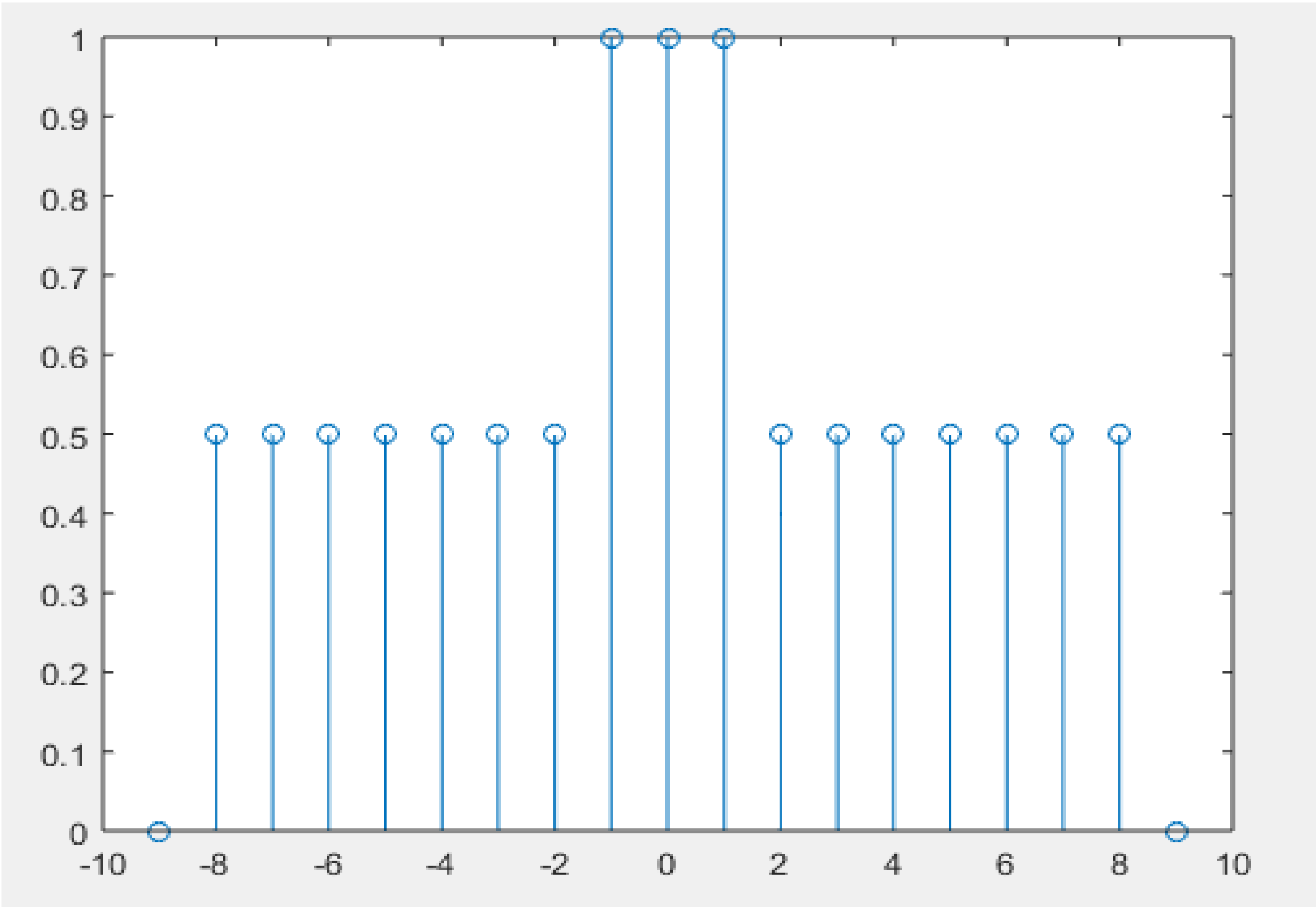
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
1
2
3
4
5
6
7

Experiment – 04

Aim – Generation of odd and even signal.

Odd signal –

```
function[xe,xo,m]=evenodd(x,n)
m=-fliplr(n);
m1=min([m,n]);
m2=max([m,n]);
m=m1:m2;
nm=n(1)-m(1);
n1=1:length(n);
x1=zeros(1,length(m));
x1(n+nm)=x;
x=x1;
xe=0.5*(x+fliplr(x));
xo=0.5*(x-fliplr(x));
stem(m,xe)
end
```



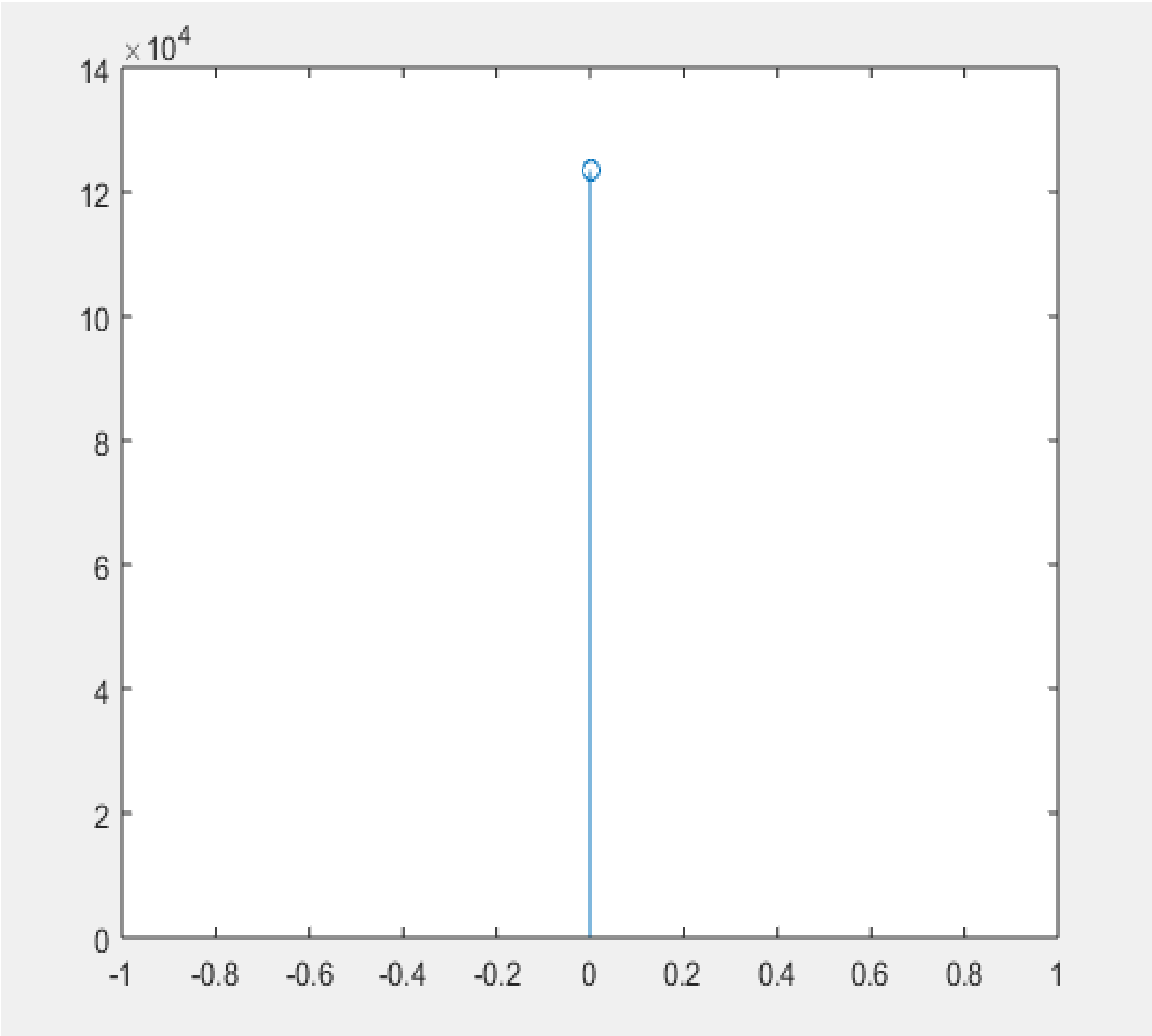
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-10
-8
-6
-4
-2
0
2
4
6
8
10

Experiment – 05

Aim – Time scaling (down sampling and up sampling).

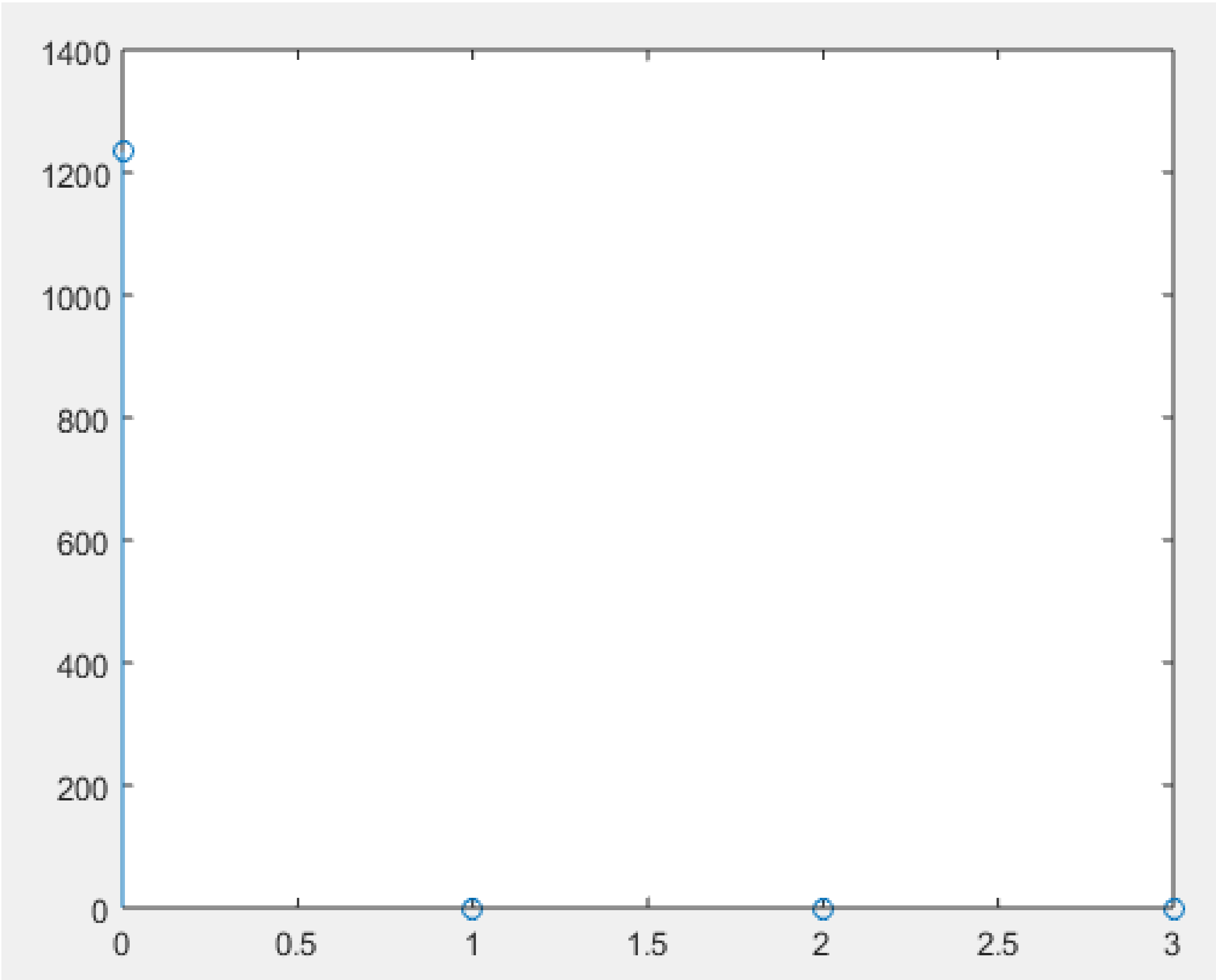
Down sampling –

```
x=[0123456];
n=0:length(x)-1;
new=mod(n,3);
new=(new==0);
x1=x(find(new==1));
n1=0:length(x1)-1;
stem(n1,x1)
```



Up sampling -

```
x=[01234];
scale=4;
xlen=length(x);
new=xlen*scale;
x1=zeros(new,1);
x1(1:scale:new)=x;
n1=0:(new-1);
stem(n1,x1)
```



1400
1200
1000
800
600
400
200
0
0
0.5
1
1.5
2
2.5
3

Experiment – 06

Aim – Convolution of DT signals.

X[n]=[1,0,0,1],[0:3]

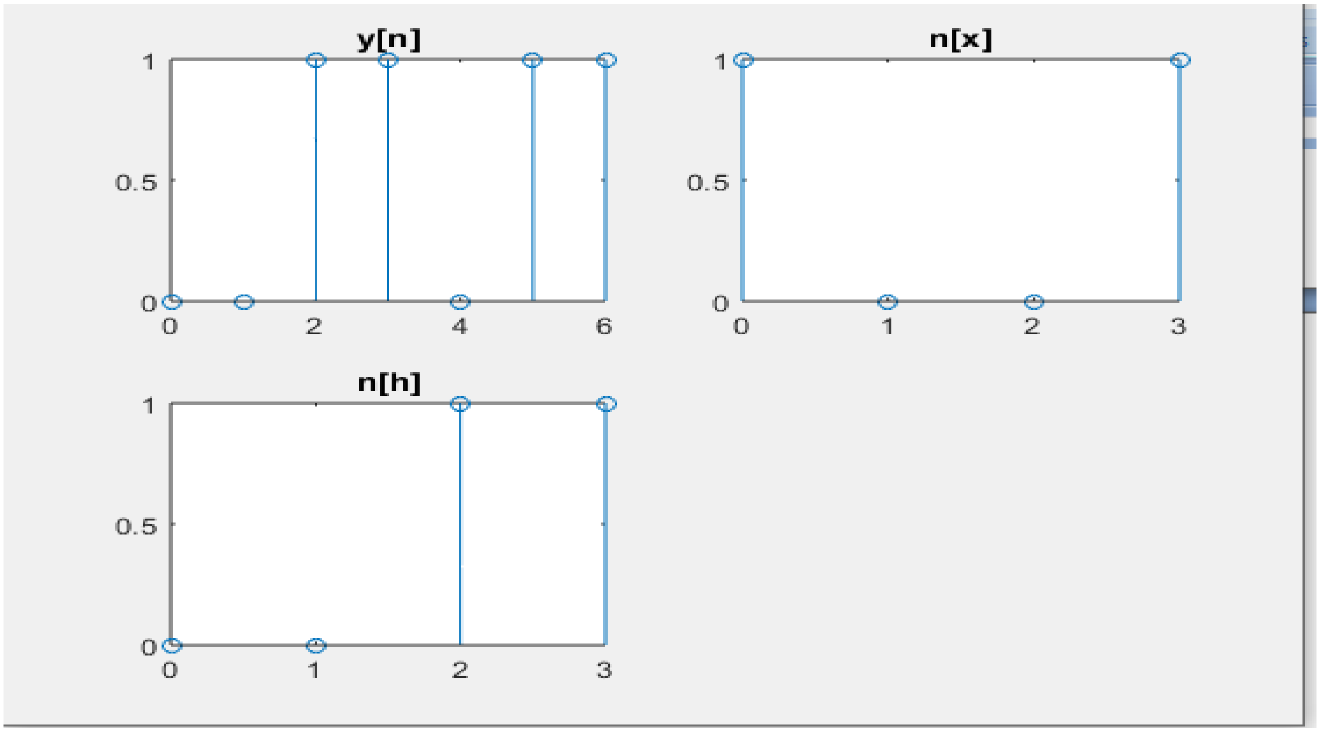
H[n]=[0,0,1,1],[0:3]

Code –

```
function[y,ny]=conv_mod(x,nx,h,nh);
nyl=nx(1)+nh(1)
nyr=nx(length(x))+ nh(length(h));
ny=nyl:nyr;
y=conv(x,h);
subplot(2,2,1);
stem(ny,y)
title('y[n]');
subplot(2,2,2);
stem(nx,x)
title('n[x]');
subplot(2,2,3);
stem(nh,h)
title('n[h]');
end
```

Output –

```
conv_mod([1,0,0,1],[0:3],[0,0,1,1],[0:3])

nyl =

     0

ans =

  Columns 1 through 5

     0     0     1     1     0

  Columns 6 through 7

   1    1
```



y[n]
n[x]
n[h]
1
0.5
0
0
2
4
6
1
2
3

Experiment – 07

Aim – Generate and plot discrete time sequence in a given interval.

X=[0 1 2 3 0 0 1 1]

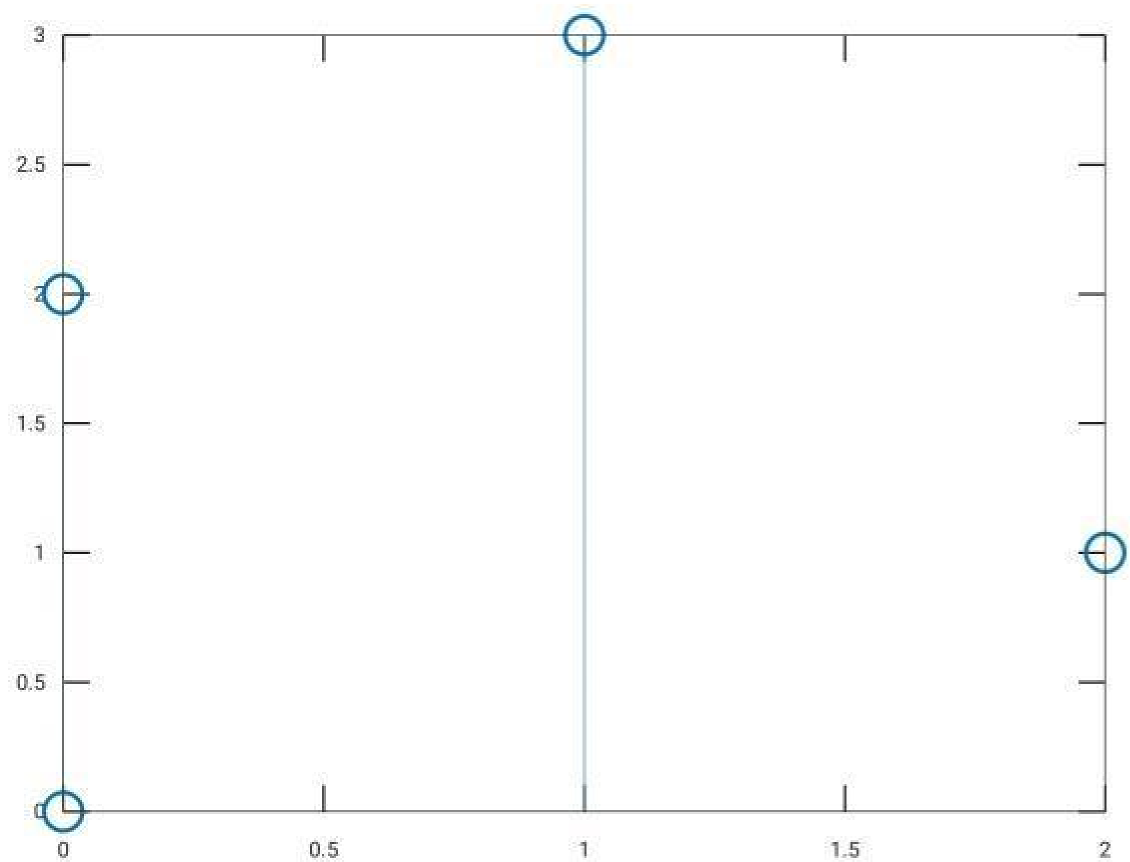
m=-3:4

code –

A) X[2n-3]

Code:

```
x=[01230011];

m=-3:4;

[y,n]=sigshift(x,m,3);

[x2,n2]=sigdownscale(y,n,2);

stem(x2,n2)
```



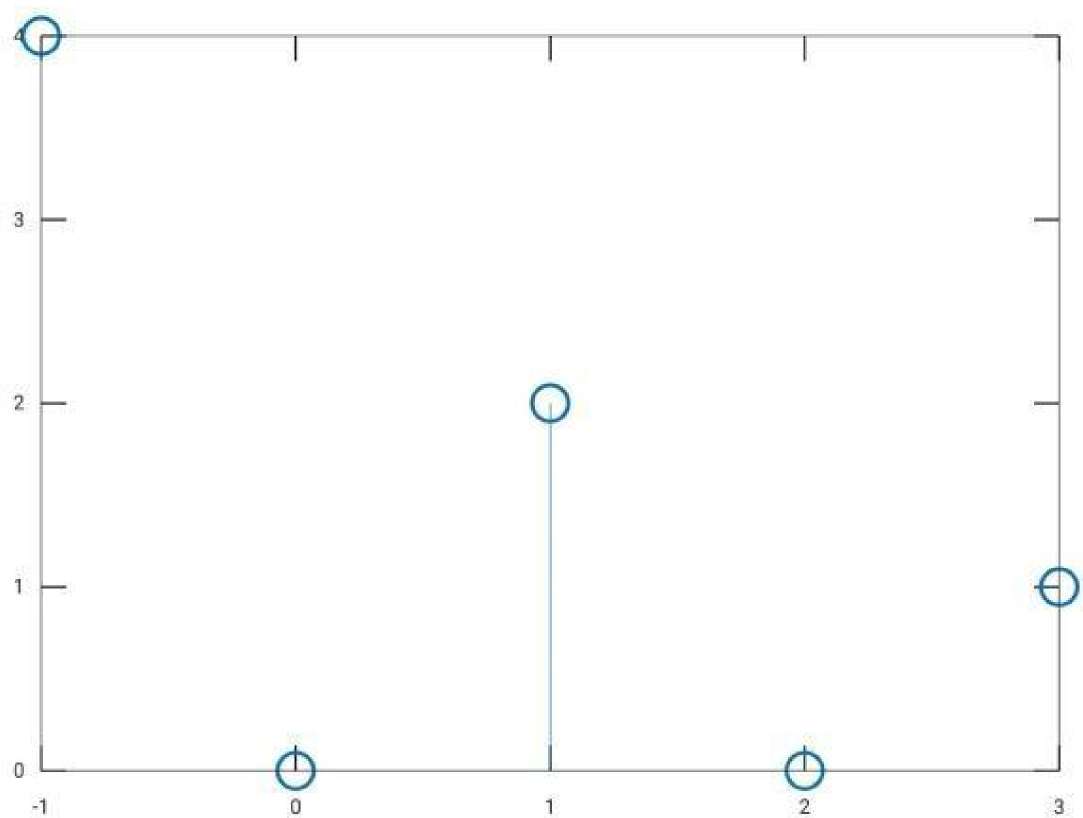
B . X[2n-3]+2*X[3n+2]

Code:

```
x=[01230011];

m=-3:4;

[x1,n1]=sigshift(x,m,3);

[x2,n2]=sigdownscale(x1,n1,2);

[x3,n3]=sigshift(x,m,
-2);

[x4,n4]=sigdownscale(x3,n3,3);

[x5,n5]=sigadd(x2,n2,2.*x4,n4);

stem(n5,x5)
```

Output -

```
y1 =

   0   0   2   0   1

y =

   4   0   2   0   1
```



Experiment – 08

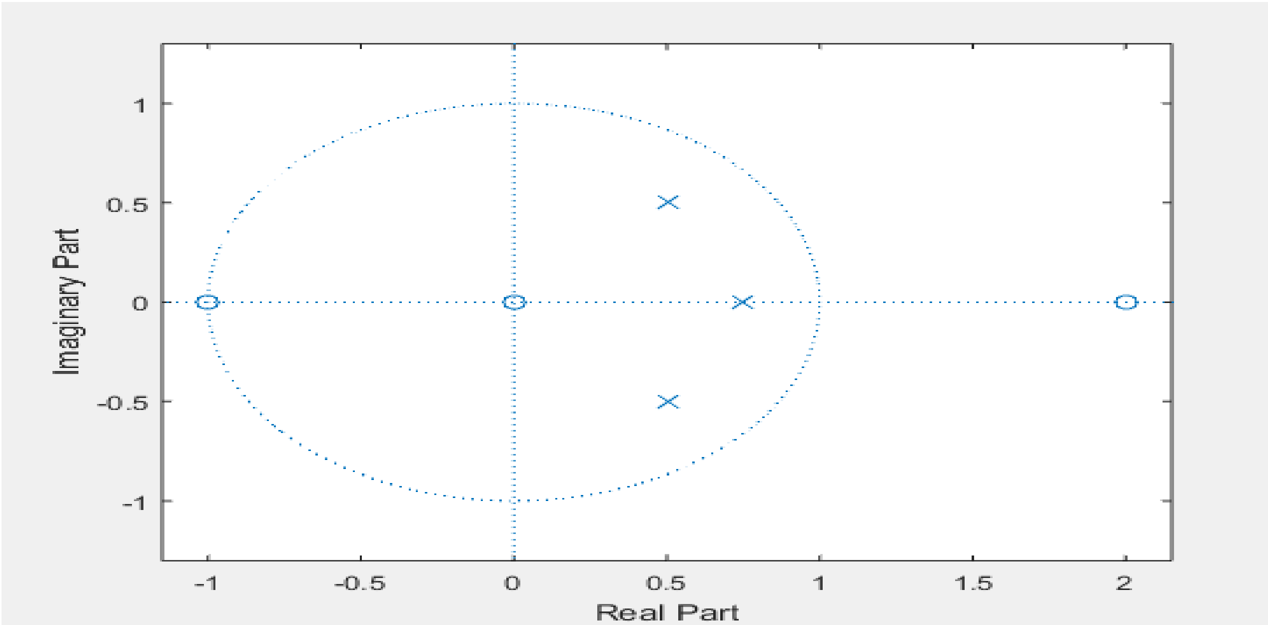
Aim – Obtain partial fraction and plot zero pole diagram

Code –

```
b=[1,-1,-2];
a=[1,-1.75,1.25,-0.375];
[R,p,C]=residuez(b,a);
zplane(b,a);
disp(R);
disp(p);
disp(C);
```

Output –

```
dsp8a

  -7.0000 + 0.0000i

   4.0000 - 2.0000i

   4.0000 + 2.0000i


   0.7500 + 0.0000i

   0.5000 + 0.5000i

   0.5000 - 0.5000i
```

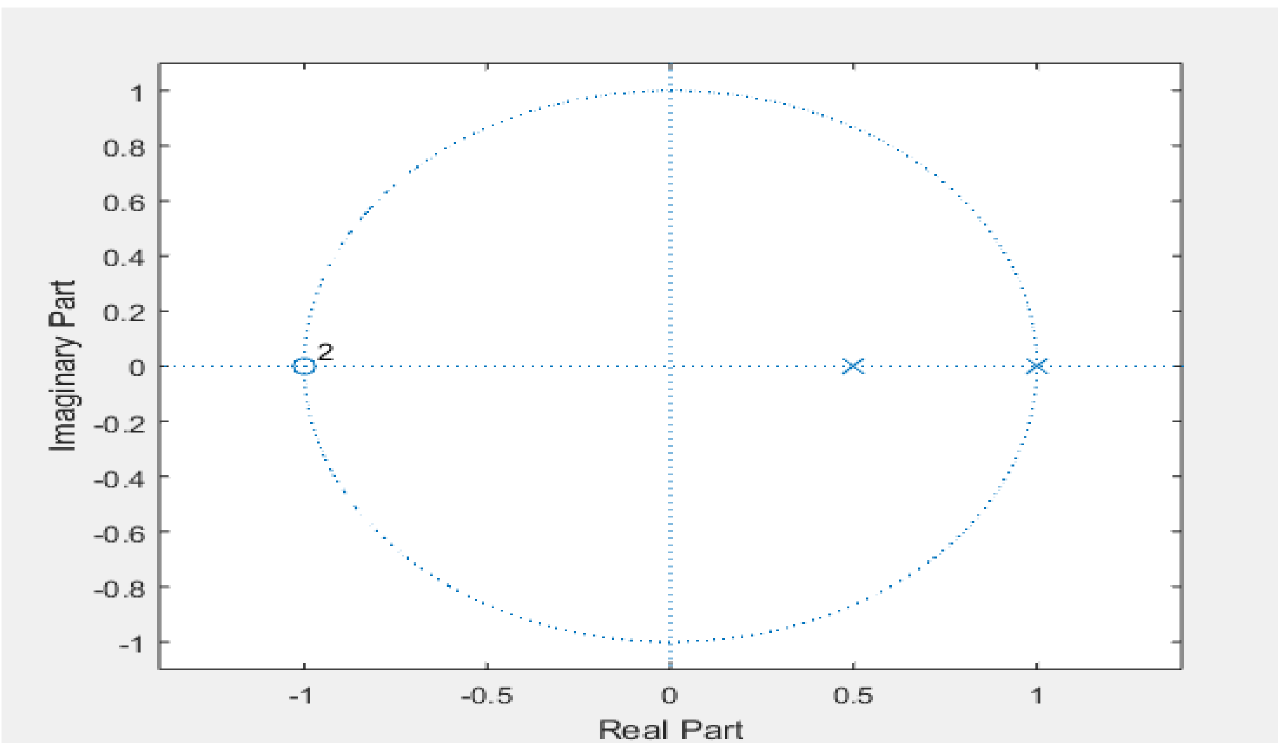


Code –

```
b=[1,2,1];

a=[1,-1.5,0.5];

[R,p,C]=residuez(b,a);

zplane(b,a);

disp(R);

disp(p);

disp(C);
```

Output -

```
dsp8b
      8
     -9


     1.0000
     0.5000


      2
```



Experiment – 09

Aim – : Z transform of various signals .

```
Signal-x[n]=(1/4)^n*u[n]
```

Code-

```
syms z n
ztrans(1/4^n)
```

output -

```
ztrans

 ans =

 z/(z - 1/4)
```

Experiment – 10

Aim – To find inverse z transform.

1)X(z)=2*z/(2*z-1)

Code-

```
syms z n;
iztrans(2*z/(2*z-1))
```

Output –

```
iztrans
```

ans =

(1/2)^n

2) X(z)=1/(z-1)

Code-

```
syms z n
iztrans(2*z/(2*z-1))

Output -
iztransb

ans =

 (1/2)^n
```

Experiment – 11

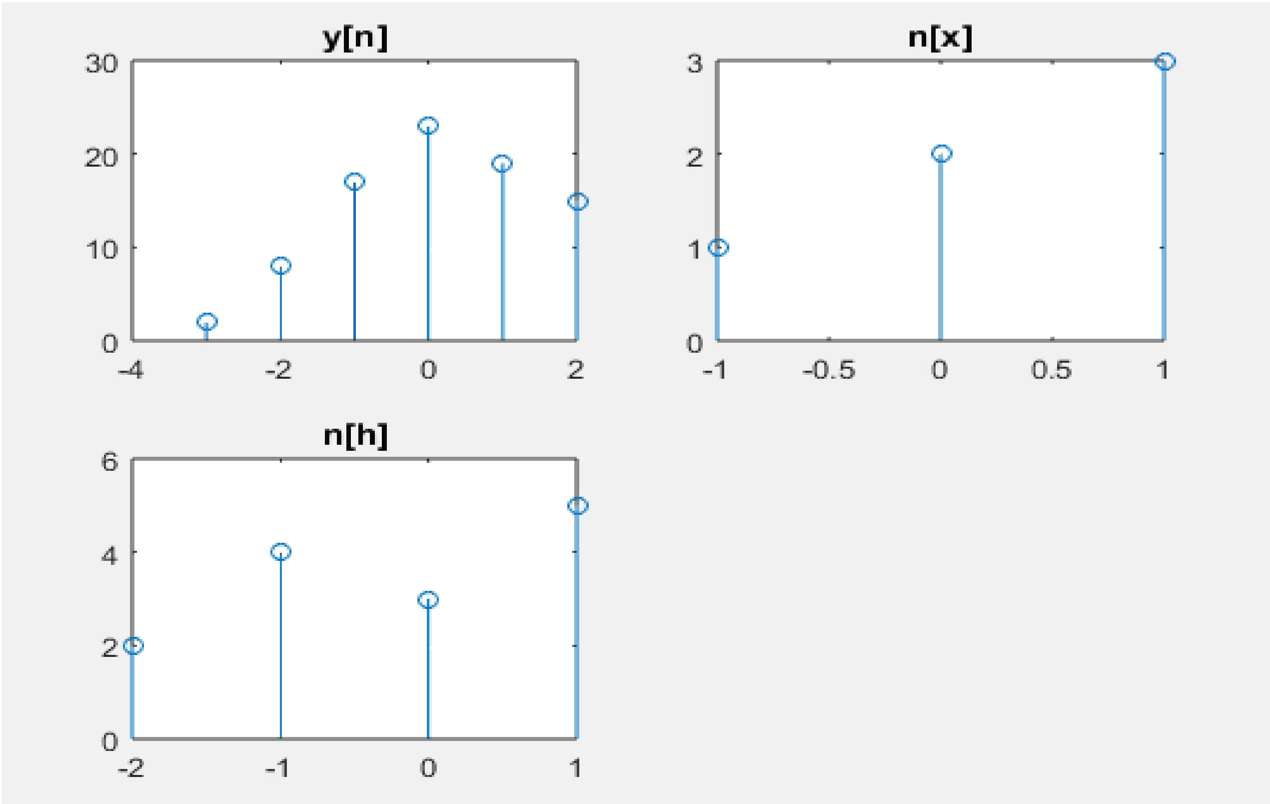
Aim – convolution in z transform.

Code –

```
x1=[2,3,4];
x2=[3,4,5,6];
x3=conv(x1,x2)
conv_mod([1,2,3],[-1:1],[2,4,3,5],[-2:1])
```

Output –

```
x3 =
 Columns 1 through 5

     6    17    34    43    38

  Column 6

    24

nyl =

    -3

ans =

  Columns 1 through 5

     2     8    17    23    19

  Column 6

    15
```



y[n]
30
20
10
0
-4
-2
0
2
n[x]
3
2
1
0
-1
-0.5
0
0.5
1
n[h]
6
4
2
0
-2
-1
0
1

Experiment – 12

Aim – Deconvolution in z transform.

Code –

```
x3=[6,17,34,43,38,24];
x1=[2,3,4];
[x2,r]=deconv(x3,x1)
```

Output –

```
dsp12

x2 =

     3     4     5     6

r =
```

Columns 1 through 5

0 0 0 0 0

Column 6

0

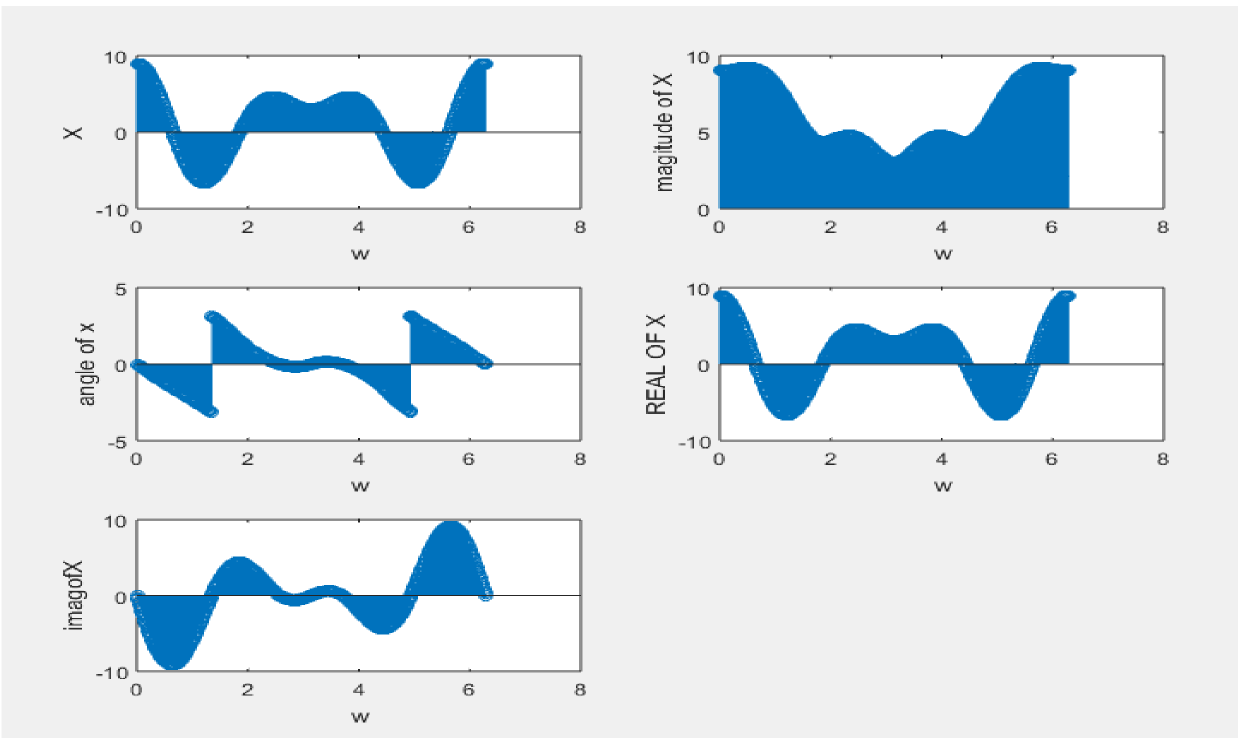
Experiment – 13

Aim – program on DTFT

Code –

```
n=-1:3;
x=[-1,1,0,5,4];
k=0:500;
w=(2*pi*k)/500;
X=x*(exp(-j*2*pi/500)).^(n'*k);
subplot(3,2,1);
stem(w,X);
ylabel('X');
xlabel('w');
magX=abs(X);
subplot(3,2,2);
stem(w,magX);
ylabel('magitude of X')
xlabel('w')
angX=angle(X);
subplot(3,2,3);
stem(w,angX);
ylabel('angle of x');
xlabel('w');
realX=real(X);
subplot(3,2,4);
stem(w,realX);
```

```
ylabel('REAL OF X');
xlabel('w');
imagX=imag(X);
subplot(3,2,5);
stem(w,imagX);
ylabel('imagofX');
xlabel('w');
```

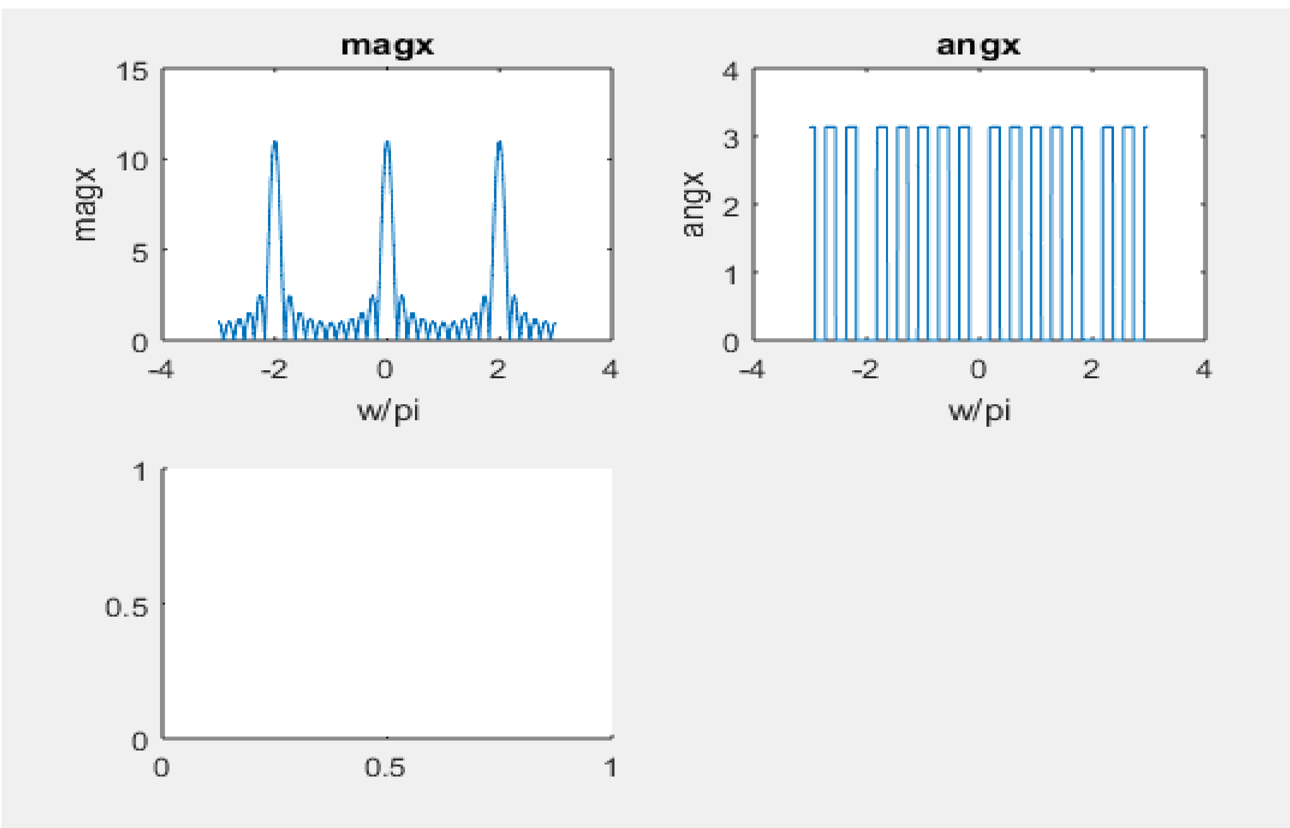


Rectangular pulse –

```
N=5;
n=[0:4];
x=[11111];
w=(-3*pi):0.01:3*pi;
X=(sin(w.*(N+1/2)))./(sin(w./2));
magx=abs(X);
angx=angle(X);
```

```
subplot(2,2,1);
plot((w/pi),magx);
title('magx');
xlabel('w/pi');
ylabel('magx');
subplot(2,2,2);
plot((w/pi),angx);
title('angx');
xlabel('w/pi');
ylabel('angx');
subplot(2,2,3)
stem(n,x)
title('x[n]');
xlabel('n');
ylabel('x');
```

output -



Experiment – 14

Aim – program on DFT.

Code –

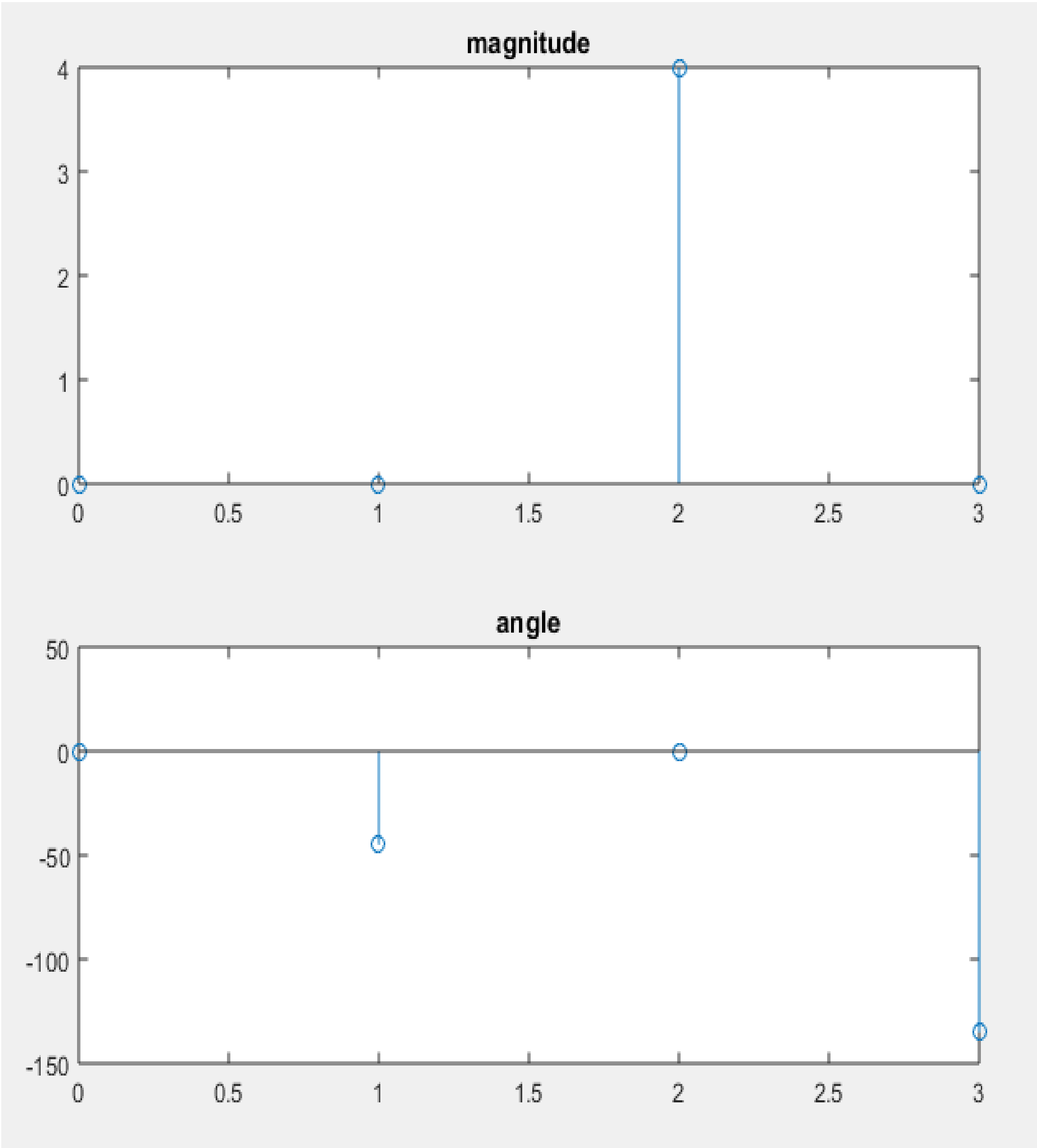
```
N=4;
n= [0:1: N-1];
x= (-1).^n;
X=dft(x,N);
magX=abs(X);
phaX=angle(X)*180/pi;
subplot(2,1,1);
stem(n,magX);
title('magnitude');
subplot(2,1,2);
stem(n,phaX);
title('angle');
```

Output –

```
n =   0      1      2      3

Xk =

 Column 1

  0.0000 + 0.0000i

  Column 2

   0.0000 - 0.0000i

  Column 3

   4.0000 + 0.0000i
```

Column 4

```
-0.0000 - 0.0000i
```


magnitude
4
3
2
1
0
0
0.5
1
1.5
2
2.5
3
angle
50
0
-50
-100
-150
0
0.5
1
1.5
2
2.5
3

Experiment – 15

Aim – program on IDFT.

Code-

```
function [xn]= idft(Xk,N)
%Computes Inverse Discrete Transform
n=[0:1:N-1]; % row vector for n
k=[0:1:N-1]; % row vector for k
WN=exp(-j*2*pi/N); %WN
nk=n'*k; % create a N by MATRIX of nk values
WNnk=WN.^(-nk); % idft  matrix
xn=(Xk*WNnk)/N; % row vector for idft values
end
```

output –

```
>>   idft([1,2,3,4,5,6],6)

ans =

   3.5000 + 0.0000i  -0.5000 - 0.8660i  -0.5000 - 0.2887i  -0.5000 + 0.0000i  -0.5000 + 0.2887i  -0.5000 + 0.8660i
```

Experiment – 16

Aim – Program on circular convolution.

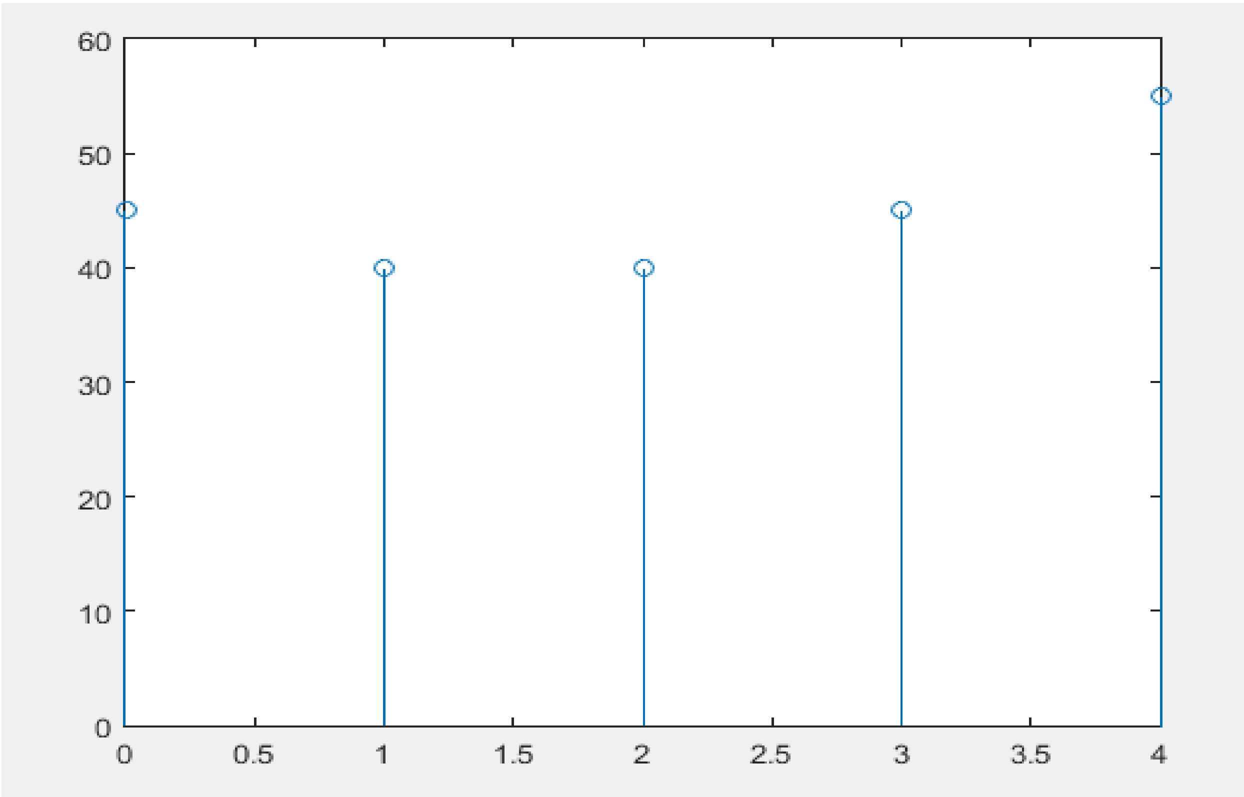
Code -

```
% Circular convolution
a=[1,2,3,4,5]; %x[N]
n=[0:1:4];  % number of samples
b=[5,4,3,2,1]; % y[n]
c=cconv(a,b,5) % circular conv
stem(n,c) % plotting the graph
```

output -

```
>> circularconv

c =

    45    40    40    45    55
```

Graph-



Experiment – 17

Aim – Program on FFT

Code -

```
x=[1,2,3,4,5,6,7,8]; % sample
n=[0:1:7] % number of samples
y=fft(x,8) % output
mag=abs(y); %magnitude
subplot(1,2,1);
stem(mag)
phase=angle(y) % phase
subplot(1,2,2);
stem(phase)
```

output –