PART I

EXPERIMENTS USING MATLAB

DIGITAL SIGNAL PROCESSING (2:0:2)

I LIST OF EXPERIMENTS USING MATLAB / SCILAB / OCTAVE / WAB

Course Outcome:

CO1: Design and Implementation of Signal Processing algorithms for applications in control systems and Signal Processing

1. Verification of sampling theorem.
2. Impulse response of a given system
3. Linear convolution of two given sequences.
4. Circular convolution of two given sequences
5. Solving a given difference equation.
6. Computation of N point DFT of a given sequence and to plot magnitude and phase spectrum.
7. Design and implementation of FIR filter to meet given specifications.
8. Design and implementation of IIR filter to meet given specifications.

II LIST OF EXPERIMENTS USING DSP PROCESSOR

1. Linear convolution of two given sequences.
2. Circular convolution of two given sequences.
3. Computation of N- Point DFT of a given sequence
4. Realization of an FIR filter (any type) to meet given specifications. The input can be a signal from function generator
5. Realization of an IIR filter (any type) to meet given specifications. The input can be a signal from function generator

Textbook:

1. Vinay K Ingle, John G Proakis, Digital Signal Processing using MATLAB, Fourth Edition, Cengage India Private Limited, 2017.

EXPERIMENT No 1

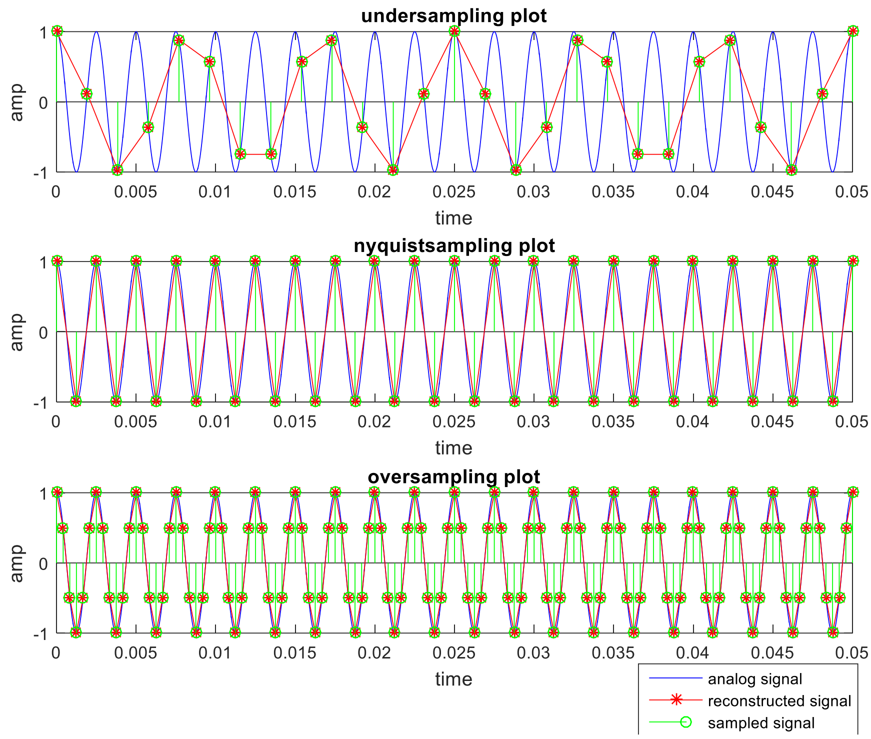
VERIFICATION OF SAMPLING THEOREM.

Aim: To verify sampling theorem for a signal of given frequency.

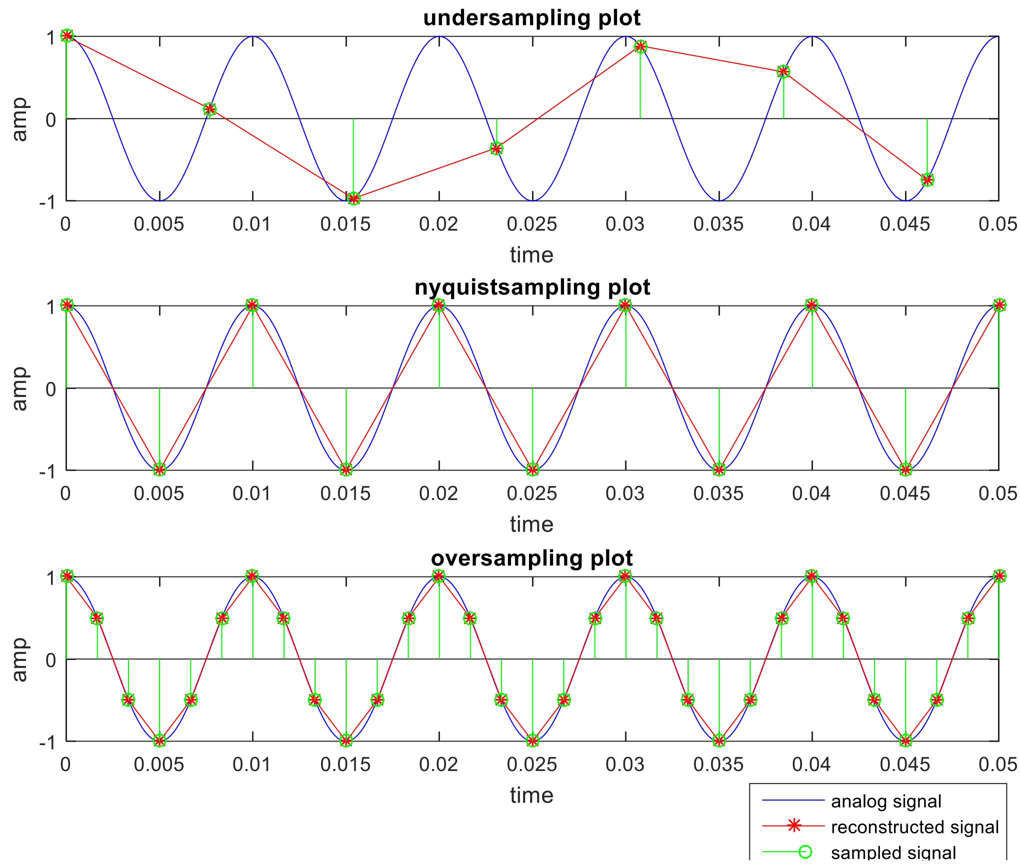
```
%1.a) Sampling theorem:

clc;
close all;
clear all;

%Input signal

f=input('enter the input freq f=');
t=0:0.001:0.1;
x=cos(2*pi*f*t);

%under sampling is fs<2fm

fs=1.5*f;
ts=1/fs;
tn=0:ts:0.1;
x1=cos(2*pi*f*tn);
subplot(3,1,1)
plot(t,x,'g',tn,x1,'r*--');
xlabel('time');
ylabel('amplitude');
title('Under Sampling');

%nyquist sampling is fs=2fm
fs=2*f;
ts=1/fs;
tn=0:ts:0.1;
x1=cos(2*pi*f*tn);
subplot(3,1,2);
plot(t,x,'g',tn,x1,'r*--');
xlabel('time');
ylabel('amplitude');
title('Critical Sampling');

%Over sampling is fs<2fm
fs=6*f;
ts=1/fs;
```

```
tn=0:ts:0.1;
x1=cos(2*pi*f*tn);
subplot(3,1,3);
plot(t,x,'g',tn,x1,'r*--');
xlabel('time');
ylabel('amplitude');
title('Over Sampling');
```

```
Output: enter the frequecny of the analog signal:400
```

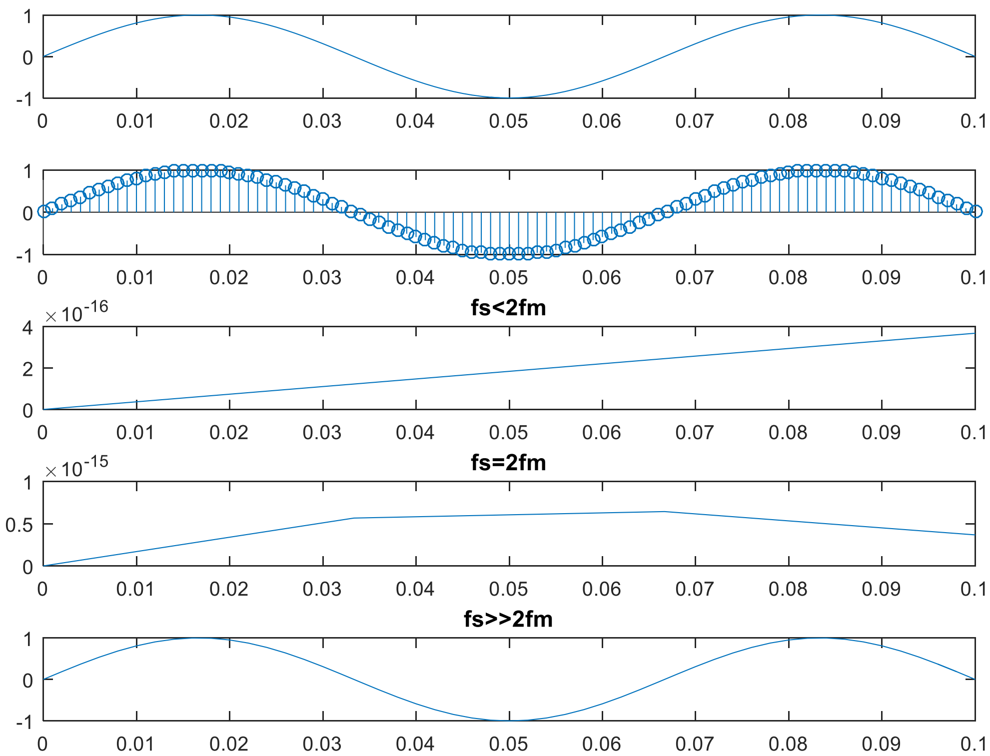



```
Output: enter the frequecny of the analog signal:100
```




```
%1.b) Sampling theorem:

close all;
clear all;
clc;
fm=15;%maximum frequency of the input signal
t=0:0.001:0.1;
msg=sin(2*pi*fm*t);
subplot(5,1,1)
plot(t,msg)
subplot(5,1,2)
stem(t,msg)

%To verify cases of Nyquist criterion
%case1:fs<2fm
fs1=10;
t1=0:1/fs1:0.1;
msg1=sin(2*pi*fm*t1);
subplot(5,1,3)
plot(t1,msg1)
title('fs<2fm')
```

```
%case2:fs=2fm
fs2=30
t2=0:1/fs2:0.1;
msg2=sin(2*pi*fm*t2);
subplot(5,1,4)
plot(t2,msg2)
title('fs=2fm')

%case3:fs>>2fm
fs3=500;
t3=0:1/fs3:0.1;
msg3=sin(2*pi*fm*t3)
subplot(5,1,5)
plot(t3,msg3)
title('fs>>2fm')
```



FOR OBSERVATION--ALIASING

```
% You may use the following code to demonstrate
%aliasing. Stem output coincides with both low and
%high frequency waveforms. If we have only stem o/p.
%we are not sure if the samples are from high or low
% frequency waveforms


hold off
t=0:1/100:1;
s=cos(2*pi*t) ; % 1Hz signal
plot(t,s) ;
hold on
s=cos(10*pi*t) ; % 5 Hz signal
plot(t,s,'r') ;
t=0:1/6:1 ;  % reduced sampling
s=cos(2*pi*t) ;
stem(t,s,'g')


%Following code will demonstrate aliasing and -ve
%frequency effects.  We don't need LabView/xcos.
%We can see the aliasing effect between 9-11 Hz and
%19-21 Hz


t=0:0.1:1;
t1=0:1/200:1;
hold off;
for i=1:0.5:22
  subplot(211);
  s=sin(i*2*pi*t1) ;
  plot(t1,s) ;
  subplot(212);
  c=sin(i*2*pi*t) ;
  stem(t,c) ;
  disp('Analog frequency=%i\n')
  input('Hit enter')
end
```

EXPERIMENT No 2

IMPULSE RESPONE OF A GINEN SYSTEM

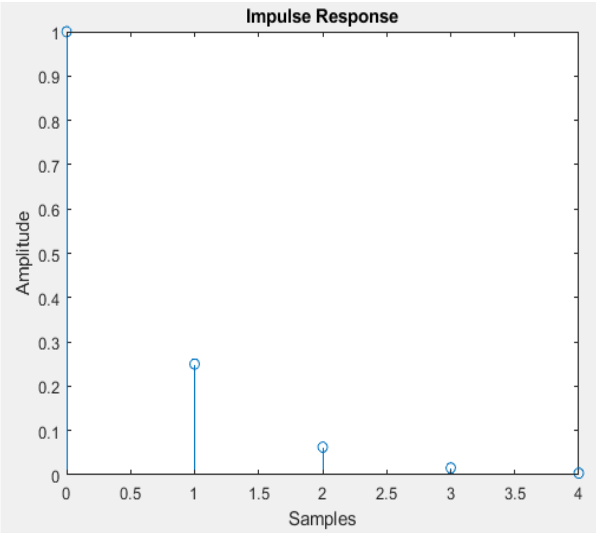
2.a For the difference equation y(n) – (1/4) y(n-1) = x(n) obtain impulse response.

b. y(n) – (3/4) y(n-1) +(1/8) y(n-2) = x(n) obtain impulse response.

```
clc;
clear all;
close all;

y=input('Enter the Numerator coefficients: ');
x=input('Enter the Denominator coefficients: ');
N=input('Enter the required impulse samples: ');
h=impz(y,x,N);
disp('The impulse response of the system is: ');
disp(h);

n=0:length(h)-1;
stem(n,h);
xlabel('Sample');
ylabel('Amplitude');
title('Impulse Response);
```

RESULT

```
Enter the Numerator coefficients: [1]
Enter the Denominator coefficients: [1 -1/4]
Enter the required impulse samples: 5
The impulse response of the system is:
    1.0000
    0.2500
    0.0625
    0.0156
    0.0039
```



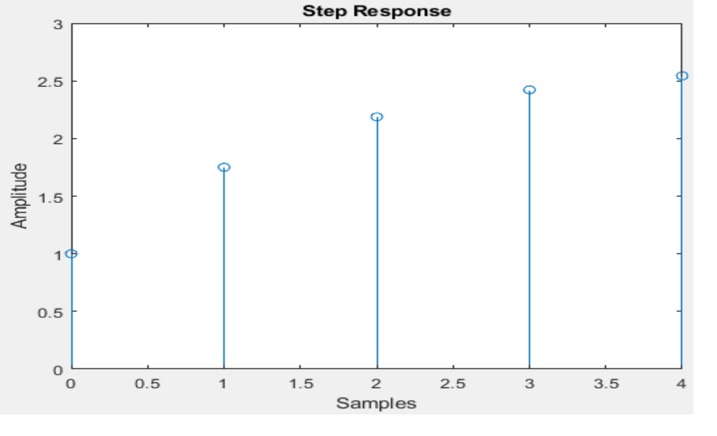
2. b. For the difference equation y(n) – (1/4) y(n-1) = x(n) obtain impulse response and step response.

```
clc;
clear all;
close all;

y=input('Enter the Numerator coefficients: ');
x=input('Enter the Denominator coefficients: ');
N=input('Enter the required impulse samples: ');

% Impulse and Step input signal

%I=[1,zeros(1,N-1)]; % impulse function
I=ones(1,N);% Step function

h=filter(y,x,I);
disp('The second order impulse response is: ');
disp(h);

n=0:length(h)-1;
stem(n,h);
xlabel('Samples');
ylabel('Amplitude');
title('Step Response’);
```

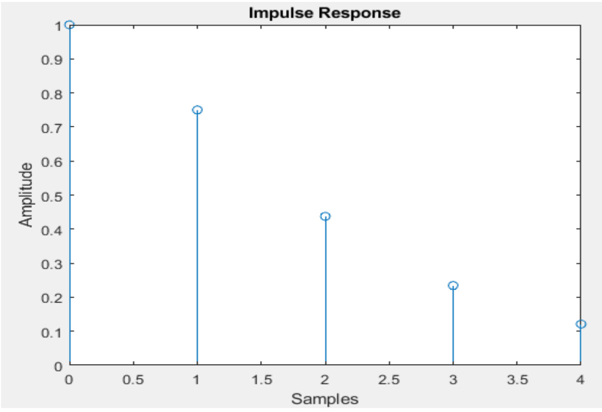
Note: Enable any one of the input signal

RESULT

```
Enter the Numerator coefficients: [1]
Enter the Denominator coefficients: [1 -3/4 1/8]
Enter the required impulse samples: 5
The second order response is:
    1.0000    1.7500    2.1875    2.4219    2.5430
```



```
Enter the Numerator coefficients: [1]
Enter the Denominator coefficients: [1 -3/4 1/8]
Enter the required impulse samples: 5
The second order impulse response is:
    1.0000    0.7500    0.4375    0.2344    0.1211
```



FOR OBSERVATION--OPTIMIZATION

```
%Simple code to demonstrate optimization.  Takes more
%time with for loop.  Avoid for loop in Matlab

clc; clear all;

N=500;
b=1:N;

tic();
c=b.*b;
toc()

tic();
for i=1:N
  a(i) = b(i) * b(i) ;
 end
toc()
```