Digital Signal Processing Laboratory Report
Experiment 1: Sampling

Rajeswari Mahapatra
15EC10044
Thursday: Group 40

1 Aim

(a) Sampling of a sinusoidal waveform above Nyquist rate

(b) Sampling at below Nyquist rate and observing the effect of aliasing

(c) Observe the spectrum of a square wave and regenerate it by interpolating it by sinc interpolation and plot the mean squared error of the regenerated signal for different values of sampling frequency.

(d) Interpolation or Upsampling

(e) Bandpass Sampling of a signal

2 Part (a) and (b): Sampling of a sinusoidal waveform above and below Nyquist rate

2.1 Theory:

Sampling at Nyquist rate:
It refers to the sampling rate being greater than or equal to twice the bandwidth of a bandlimited function or a bandlimited channel. Suppose the highest frequency component, in hertz, for a given analog signal is f_{max}. According to the Nyquist Theorem, the sampling rate must be at least $2f_{max}$, or twice the highest analog frequency component. The sampling in an analog-to-digital converter is actuated by a pulse generator (clock). If the sampling rate is less than $2f_{max}$, some of the highest frequency components in the analog input signal will not be correctly represented in the digitized output. When such a digital signal is converted back to analog form by a digital-to-analog converter, false frequency components appear that were not in the original analog signal. This undesirable condition is a form of distortion called ***Aliasing***.

Thus, the condition to avoid aliasing is:

$$f_s \geq 2f_{max} \tag{1}$$

where,
f_s=Sampling frequency;
f_{max}=Highest frequency present in a signal, or bandwidth;

2.2 Code Snippets:

Given signal:

$$x(t) = 10cos(2\pi \times 10^3 t) + 6cos(2\pi \times 2 \times 10^3 t) + 2cos(2\pi \times 4 \times 10^3 t); \tag{2}$$

This signal has the highest frequency of 4kHz, and thus has a bandwidth of 8kHz. The following MATLAB script samples the given signal at 12kHz:

```
clear all;
clc;

fs=12e3;
f0=200e3;

t=0:1/f0:59/fs;
x = 10*cos(2*pi*1000*t) + 6*cos(2*pi*2000*t) + 2*cos(2*pi*4000*t);

t0=0:1/fs:59/fs;
x0 = 10*cos(2*pi*1000*t0) + 6*cos(2*pi*2000*t0) + 2*cos(2*pi*4000*t0);
figure(4);
plot(t,x);
hold on;
stem(t0,x0);
grid on;
xlabel('Time');
ylabel('Amplitude');
title('Original signal sampled at F=12kHz');
```

The following code snippet does the FFT of the sampled signal, here, it has been shown for N=64, the value of N can be changed for different DFTs:

- N=64

```
n1=64;
t1=0: 1/fs: (n1-1)/fs;
x1 = 10*cos(2*pi*1000*t1) + 6*cos(2*pi*2000*t1) + 2*cos(2*pi*4000*t1);
y1=fft(x1,n1);
y11=fftshift(y1);
```

```
f1=linspace(-pi, pi, n1);



figure(1);
stem(f1, abs(y11));
xlabel('Frequency');
ylabel('DFT');
title('DFT of the sampled signal with N=64');
grid on;
```

- N=128

```
n2=128;
t2=0: 1/fs: (n2-1)/fs;
x2 = 10*cos(2*pi*1000*t2) + 6*cos(2*pi*2000*t2) + 2*cos(2*pi*4000*t2);
y2=fft(x2,n2);
y21=fftshift(y2);
f2=linspace(-pi, pi, n2);

figure(2);
stem(f2, abs(y21));
xlabel('Frequency');
ylabel('DFT');
title('DFT of the sampled signal with N=128');
grid on
```

- N=256

```
n3=256;
t3=0: 1/fs: (n3-1)/fs;
x3 = 10*cos(2*pi*1000*t3) + 6*cos(2*pi*2000*t3) + 2*cos(2*pi*4000*t3);
y3=fft(x3,n3);
y31=fftshift(y3);
f3=linspace(-pi, pi, n3);

figure(3);
stem(f3, abs(y31));
xlabel('Frequency');
ylabel('DFT');
title('DFT of the sampled signal with N=256');
grid on;
```

Part (b) asks for sampling at three different frequencies below the Nyquist rate, i.e at 8kHz, 5kHz and 4kHz. This can be done by changing the value of fs in the above code where the value of fs was previously 12kHz.

2.3 Figures and Plots:

2.3.1 Sampling frequency=12kHz

For, sampling frequency f_s=12kHz, the plots below depict the sampled wave, and the magnitude spectrum of the DFTs of the sampled wave at given different values of N.

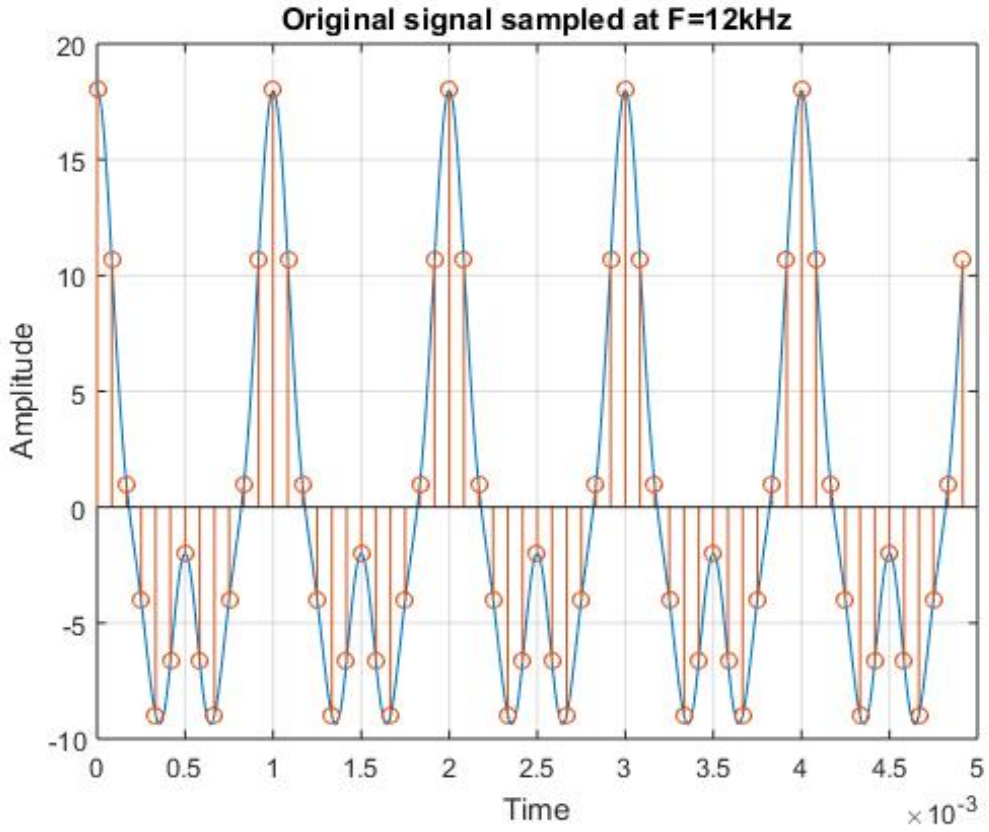

Figure 1: The original signal being sampled at 12kHZ

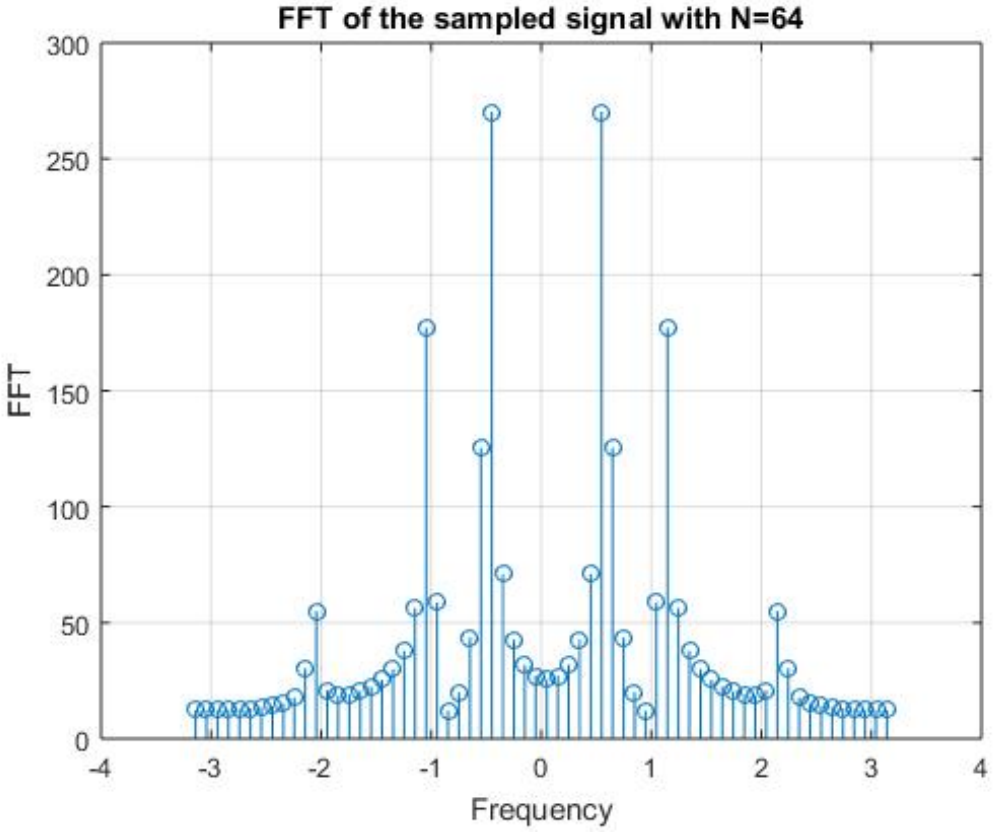

Figure 2: DFT of the sampled signal with N=64

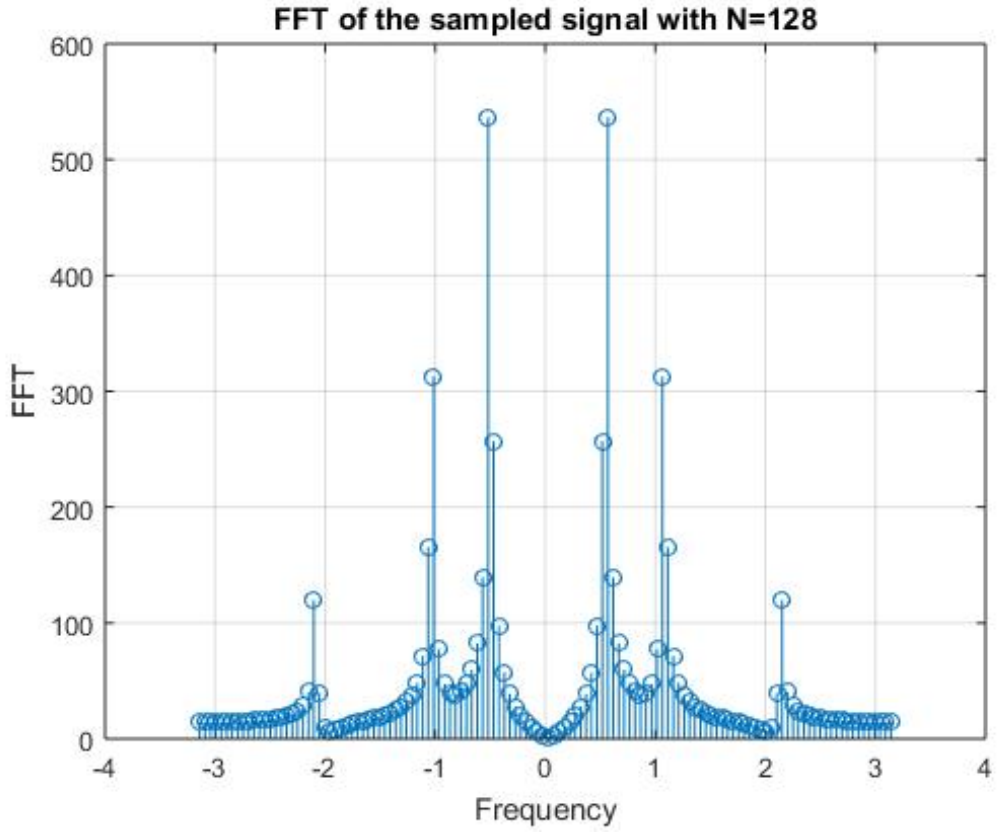

Figure 3: DFT of the sampled signal with N=128

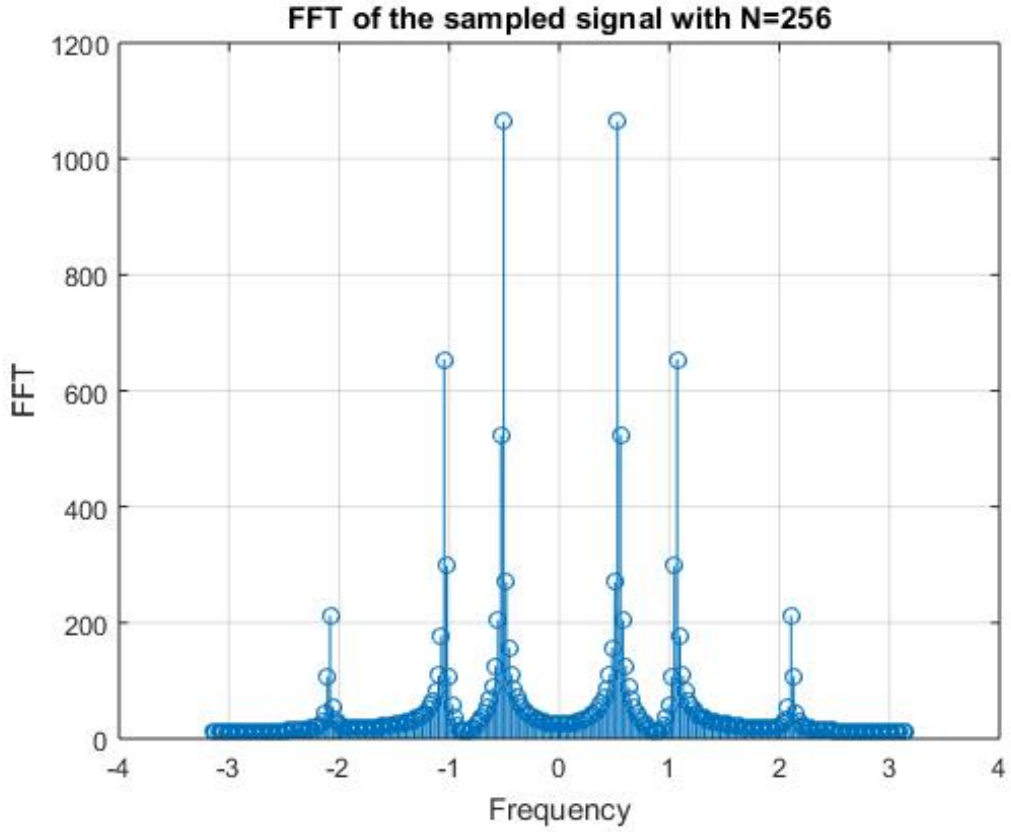

Figure 4: DFT of the sampled signal with N=256

2.3.2 Sampling Frequency=5kHz

For, sampling frequency f_s=5kHz, the plots below depict the sampled wave, and the magnitude spectrum of the DFTs of the sampled wave at given different values of N.

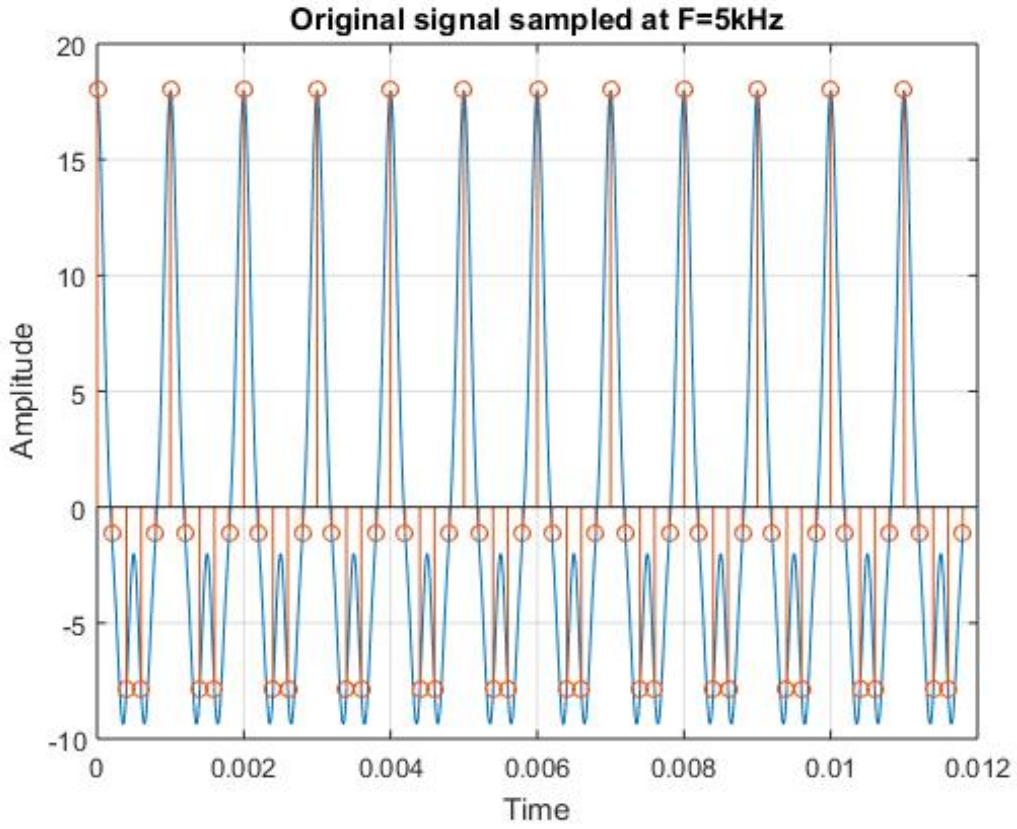

Figure 5: The original signal being sampled at 5kHZ

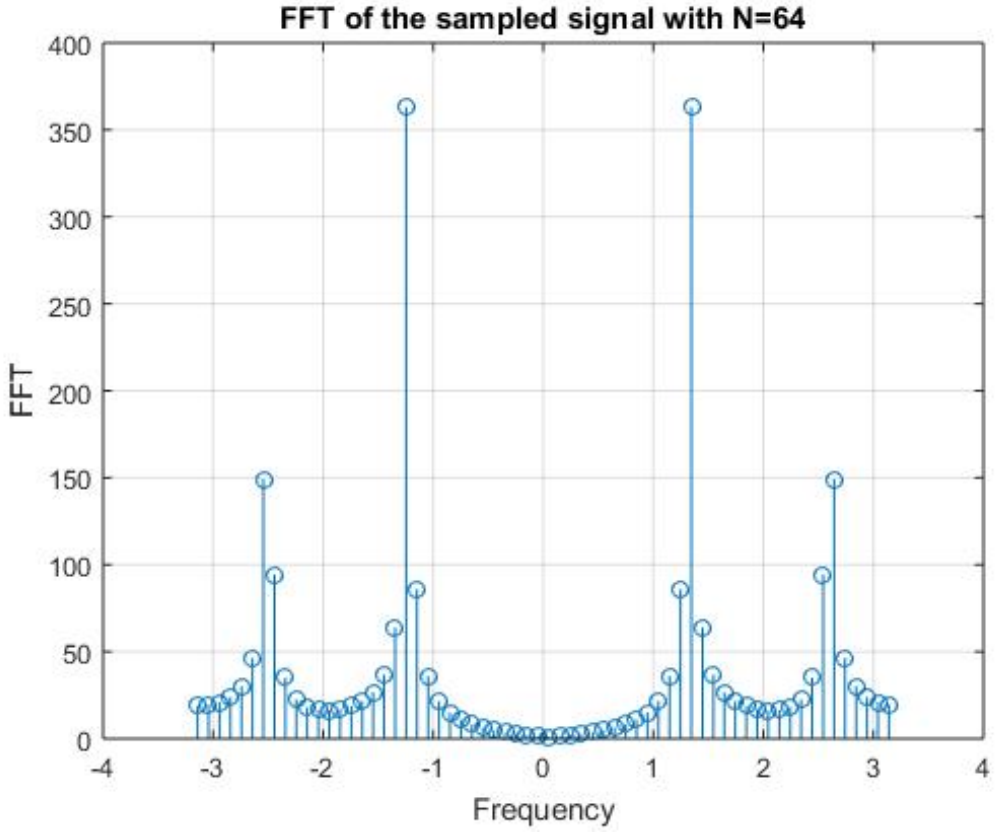

Figure 6: DFT of the sampled signal with N=64

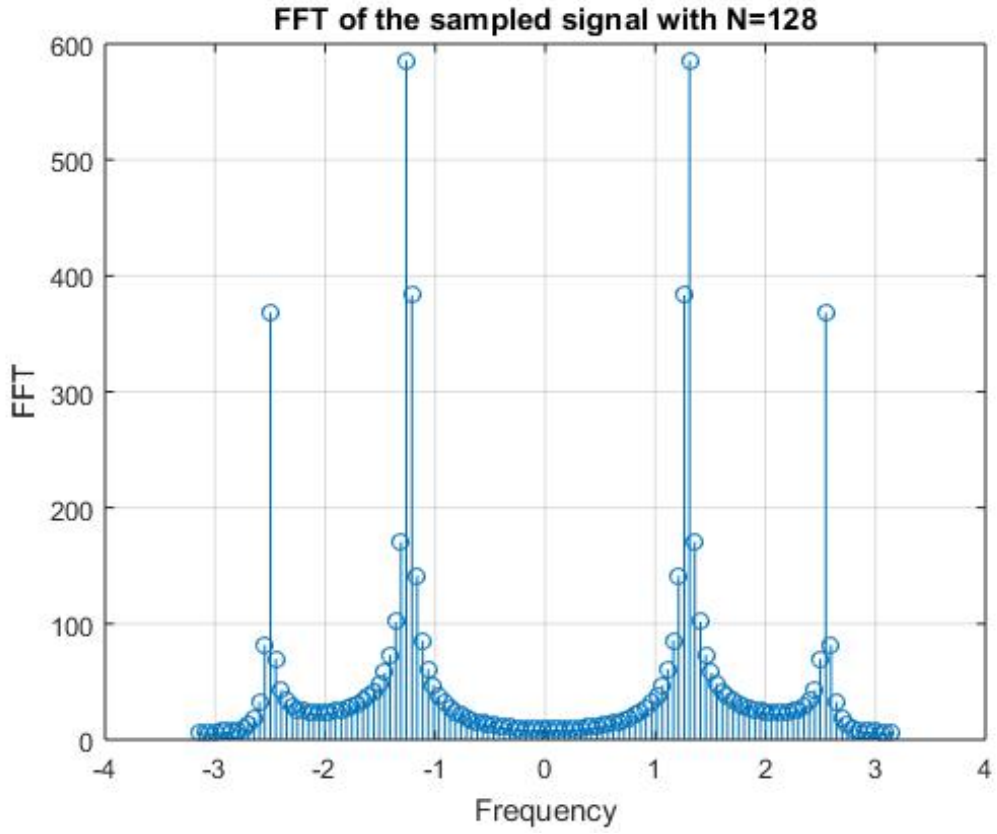

Figure 7: DFT of the sampled signal with N=128

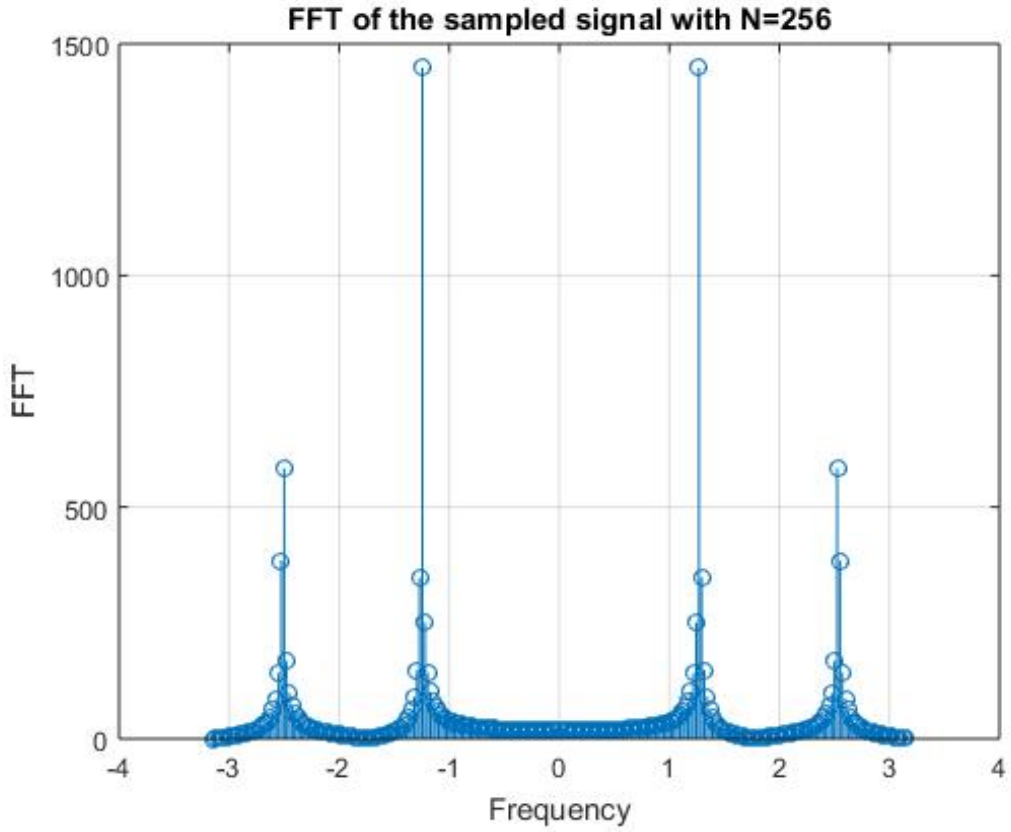

Figure 8: DFT of the sampled signal with N=256

2.3.3 Sampling Frequency=4kHz

For, sampling frequency f_s=4kHz, the plots below depict the sampled wave, and the magnitude spectrum of the DFTs of the sampled wave at given different values of N.

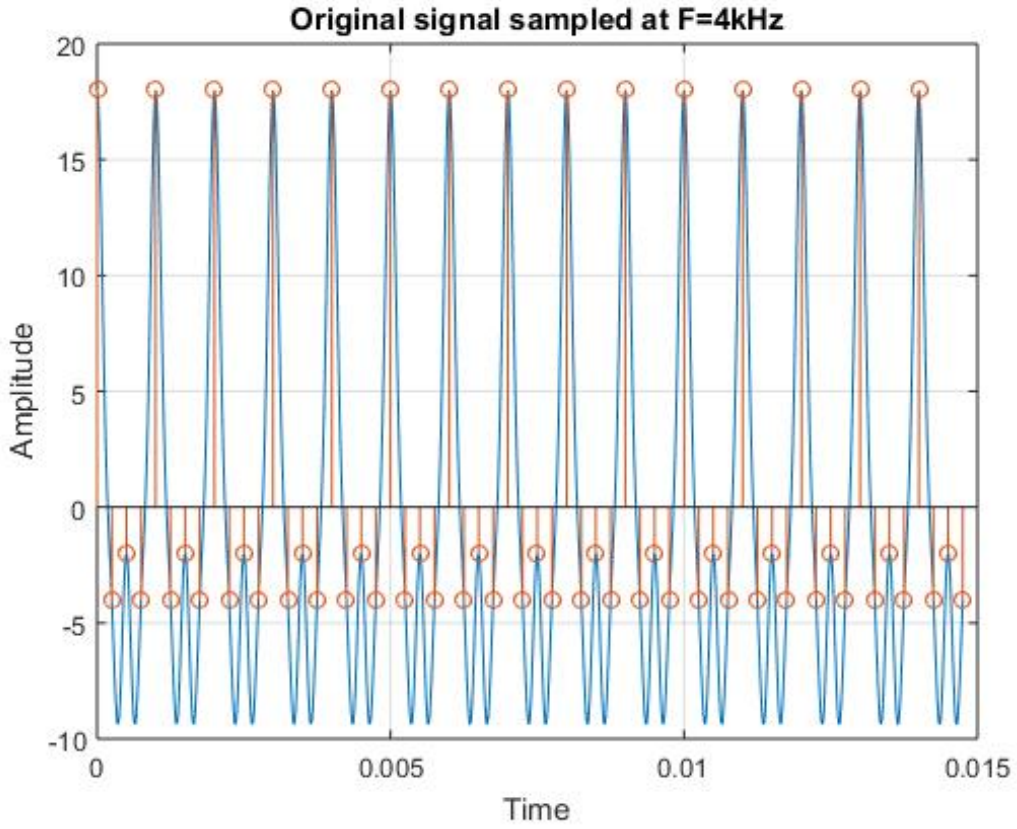

Figure 9: The original signal being sampled at 4kHZ

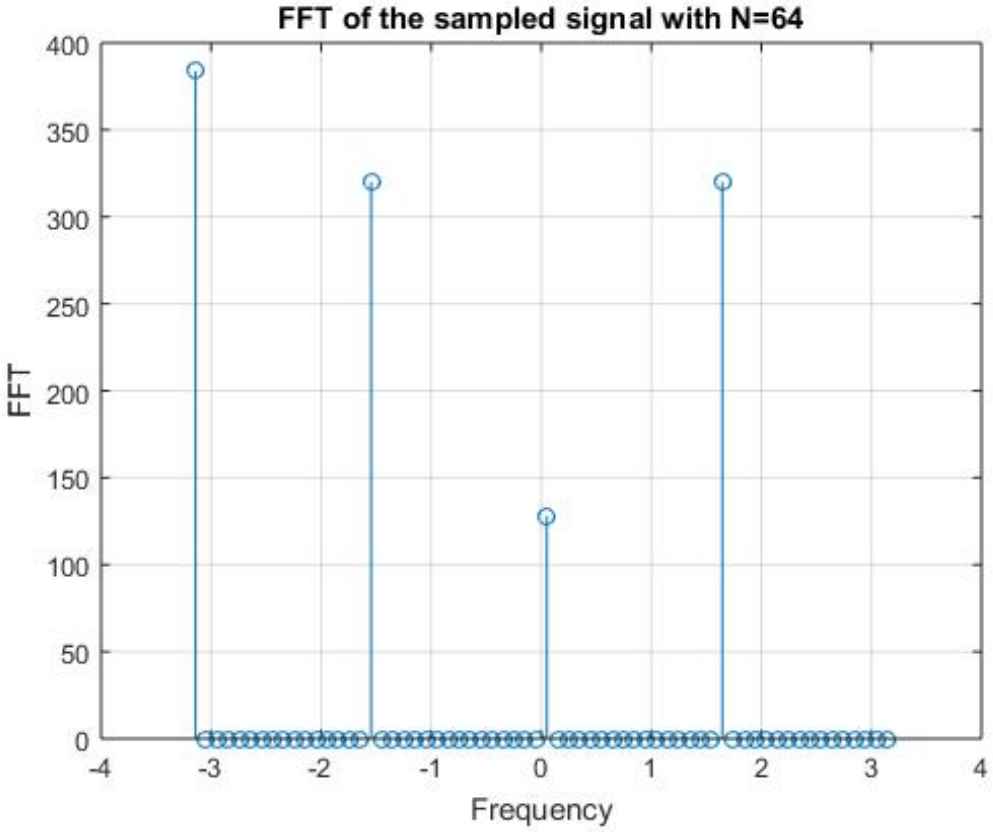

Figure 10: DFT of the sampled signal with N=64

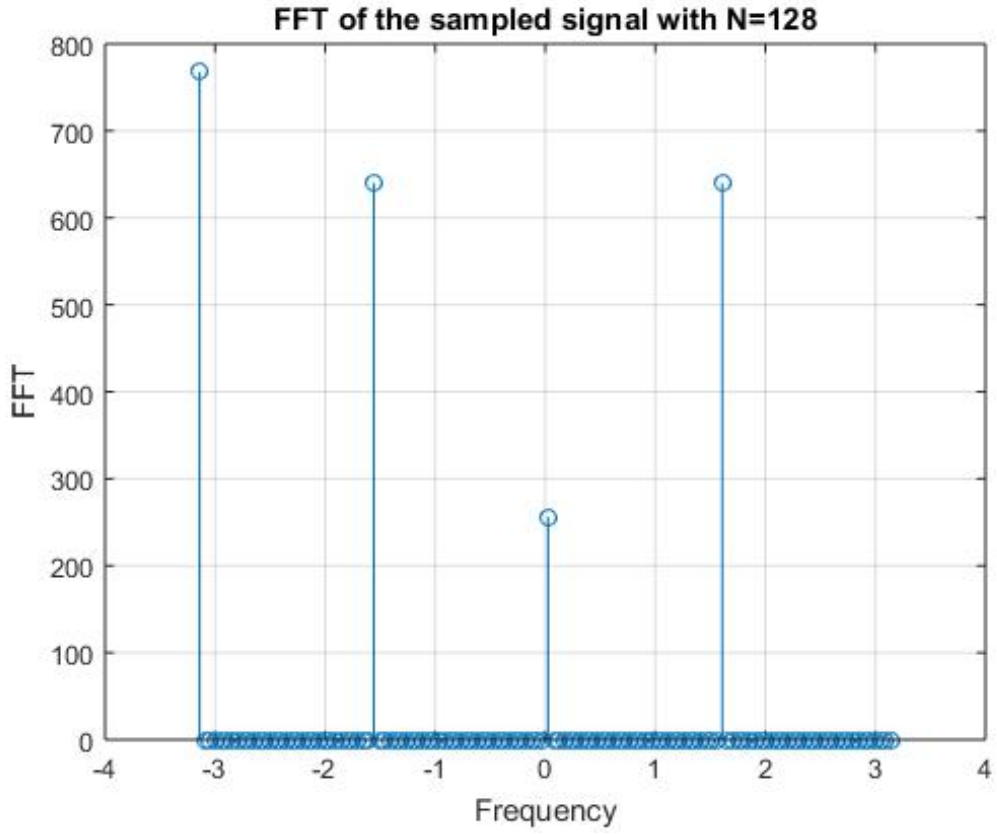

Figure 11: DFT of the sampled signal with N=128

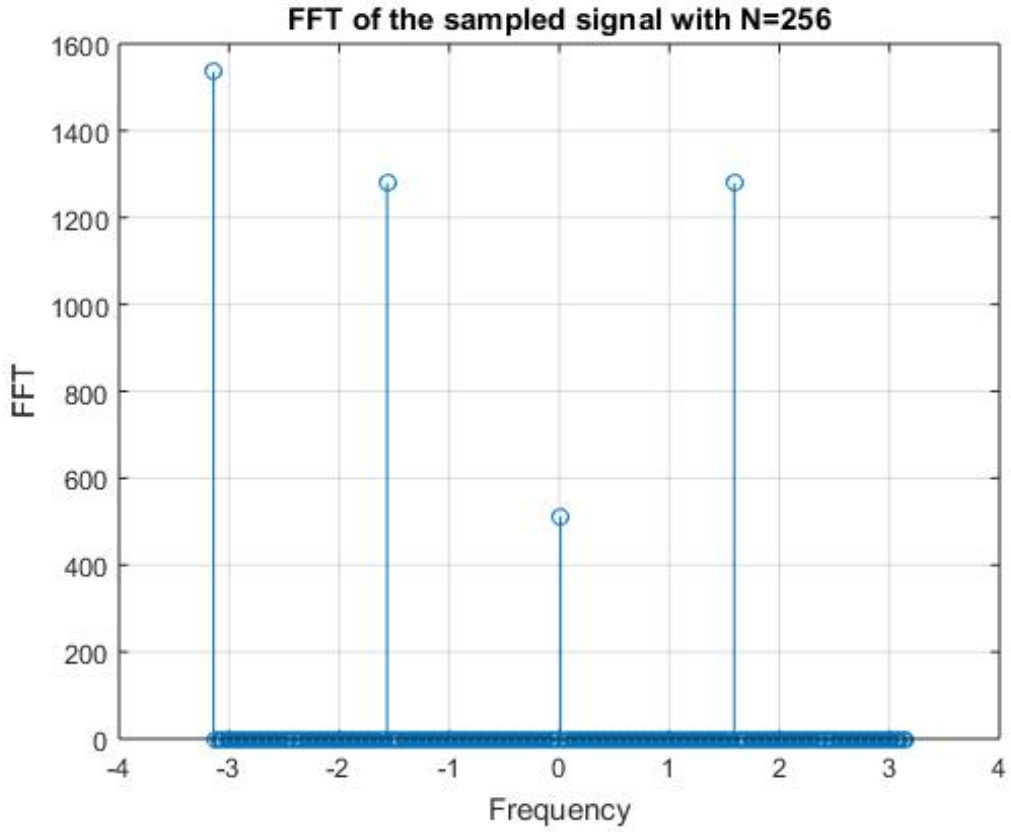

Figure 12: DFT of the sampled signal with N=256

2.3.4 Sampling Frequency=8kHz

For, sampling frequency f_s=8kHz, the plots below depict the sampled wave, and the magnitude spectrum of the DFTs of the sampled wave at given different values of N.

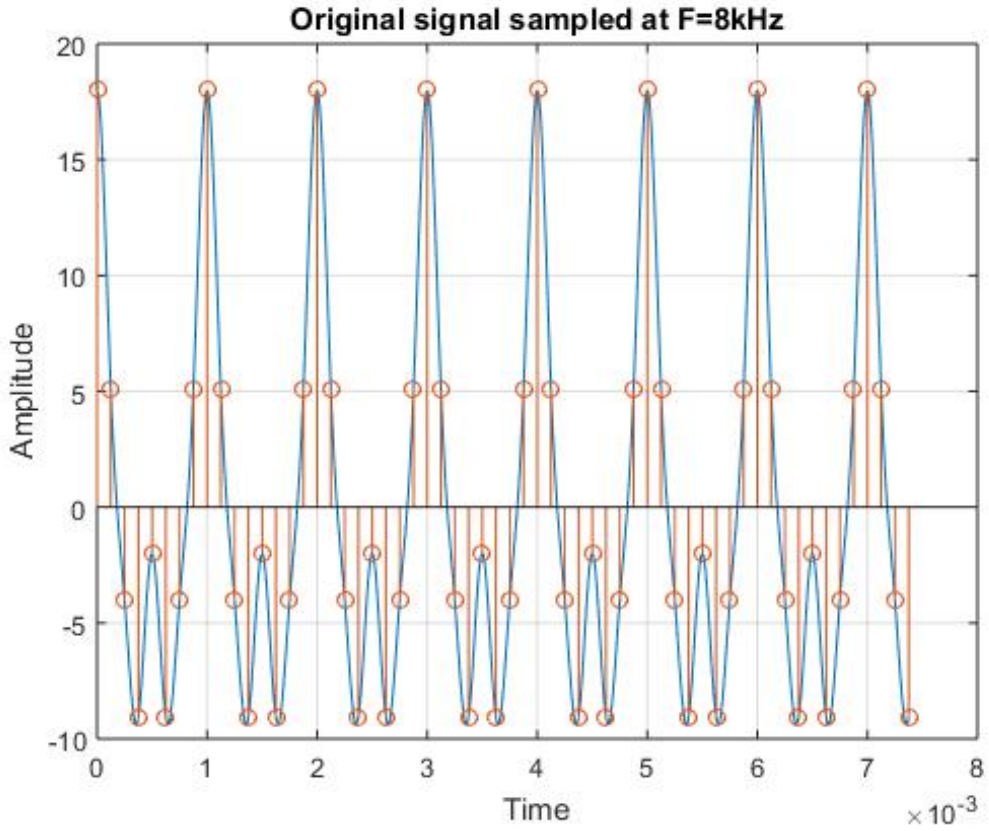

Figure 13: The original signal being sampled at 8kHZ

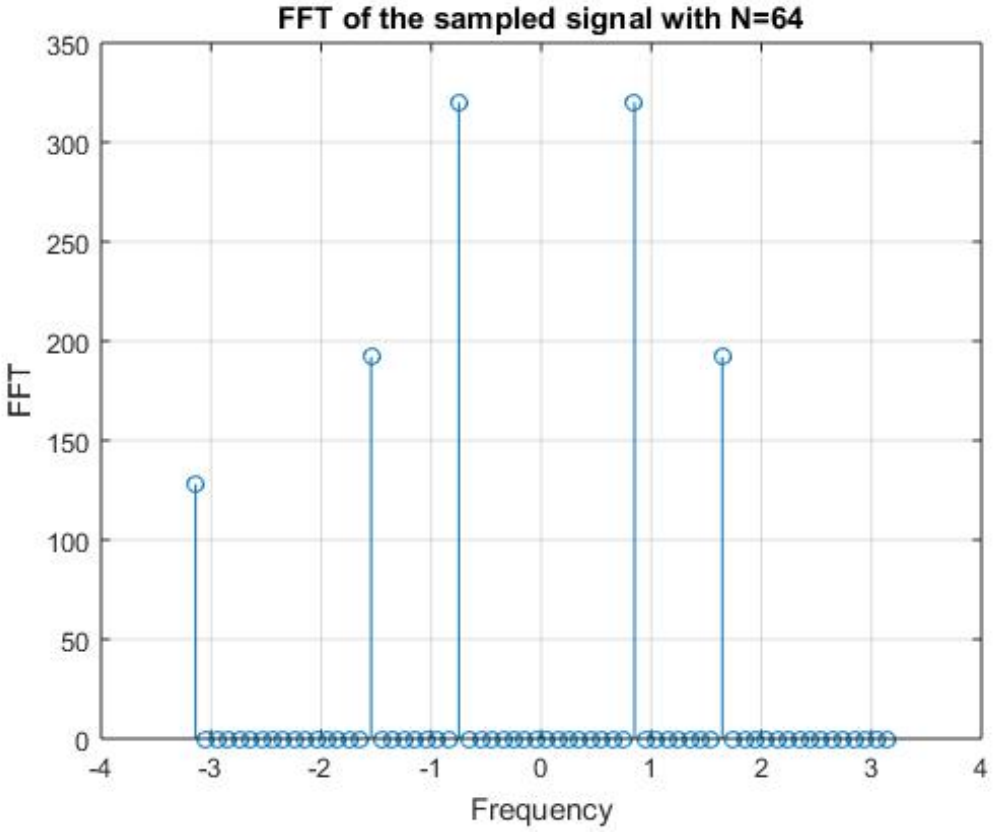

Figure 14: DFT of the sampled signal with N=64

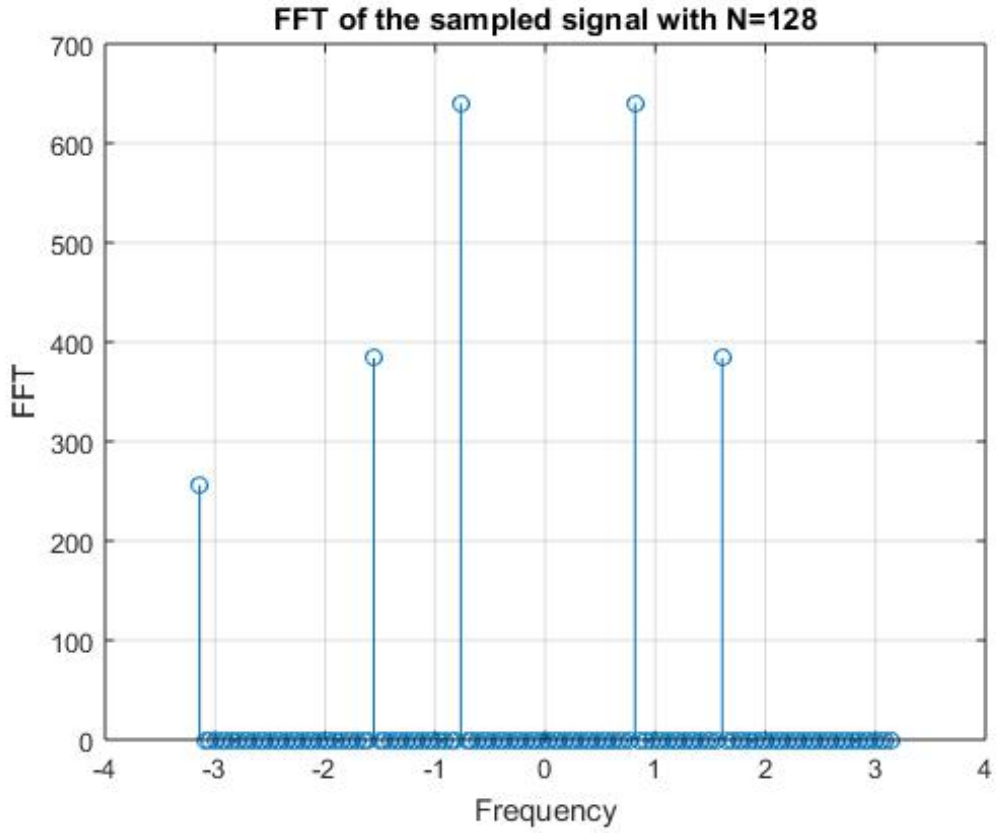

Figure 15: DFT of the sampled signal with N=128

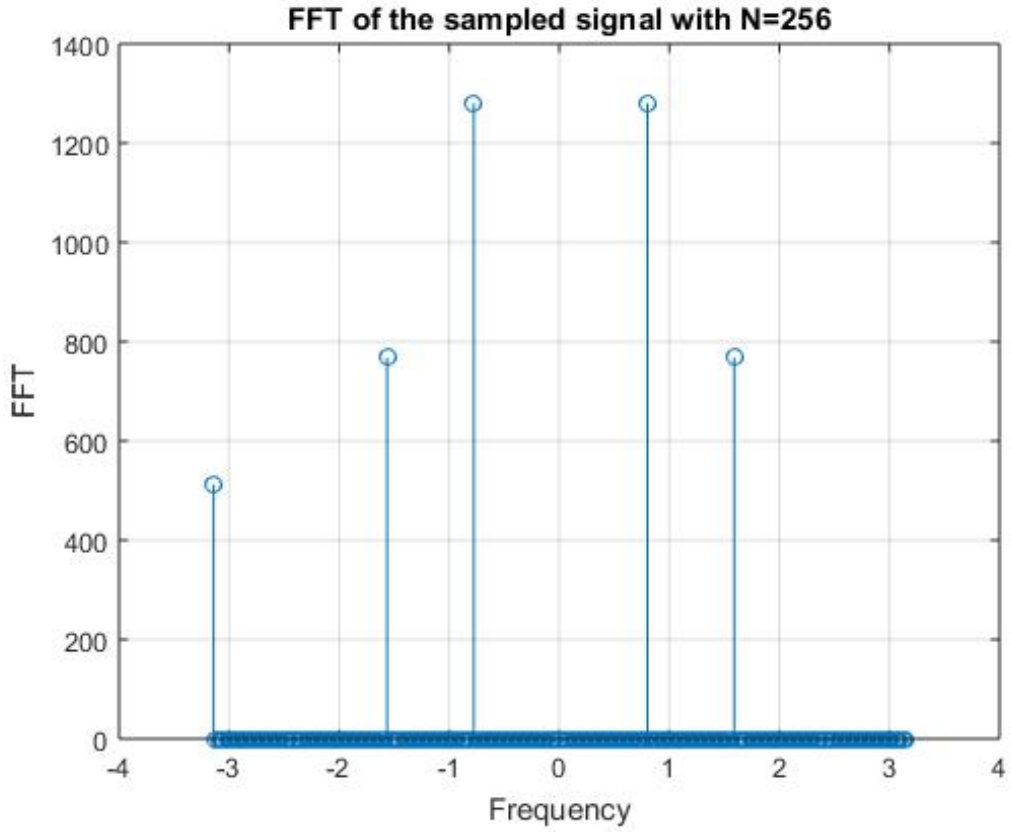

Figure 16: DFT of the sampled signal with N=256

3 Part(c): Observe the spectrum of a square wave and regenerate it by interpolating it by sinc interpolation and plot the mean squared error of the regenerated signal for different values of sampling frequency.

3.1 Theory:

Spectrum of a square wave:
The ideal square wave contains only components of odd-integer harmonic frequencies, and it has an infinite bandwidth.In this experiment, we obtained the DFT of the square wave signal, and reconstructed it by sinc interpolation. Later, the mean squared error Vs. sampling frequency was plotted for a range of 0 to 500kHz.

3.2 Code Snippets:

Here, we take a square wave of frequency 1 kHz, sampled at 20Khz, which is quite above the Nyquist rate. Then, we reconstruct the the given signal from it's DFT, using sinc interpolation. At last, we plot the mean squared error of the reconstructed signals sampled over a frequency range of 0 to 500 kHz frequency.

```
F = 1e3;
N = 2048; % Sampling Frequency

Fs = 20e3;

t_N = linspace(0,(N-1)/Fs, N);
x_N = square(t_N*2*pi*F);

FFT = fft(x_N, N);%FFT of sampled signal (0,2pi)
FFT_shift = fftshift(FFT);
f_t = linspace(-1*pi, pi, N);
figure;
stem(f_t, abs(FFT_shift));
str=sprintf('FFT with Fs = %d kHz', Fs/1000);
title(str);
xlabel('Frequecy') % x-axis label
ylabel('FFT') % y-axis label


t = linspace(0,5/F,5000);
x_t = square(t*2*pi*F);
figure;
subplot(2,1,1);
plot(t,x_t);
str=sprintf('Orignal signal with Fs = %d kHz', Fs/1000);
title(str);
xlabel('t') % x-axis label
ylabel('x(t)') % y-axis label


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

for i = 1:length(t_N);
    regen = regen + x_N(i)*sinc(t*Fs - i);
end

subplot(2,1,2);
plot(t,regen);
str=sprintf('Reconstructed signal with Fs = %d kHz', Fs/1000);
```

```
title(str);
xlabel('t') % x-axis label
ylabel('x(t)') % y-axis label

N = 2048;
fq = [1e3:1e3:0.5e6];
err = zeros(1,length(fq));

for Fs = fq

t_N = linspace(0,(N-1)/Fs, N);
x_N = square(t_N*2*pi*F);

FFT = fft(x_N, N);
FFT_shift = fftshift(FFT);

t = linspace(0,1/F,1000);
x_t = square(t*2*pi*F);

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

for i = 1:length(t_N);
    regen = regen + x_N(i)*sinc(t*Fs - i);
end

D = abs(x_t-regen).^2;
err(Fs/1000) = sum(D(:))/numel(x_t);

end

figure;
plot(fq,err);
title('Variation of Mean Squared Error with Fs');
xlabel('Fs') % x-axis label
ylabel('Mean squared Error')
```

3.3 Figures and Plots:

3.3.1 The square wave and the reconstructed signal(Fs=20kHz):

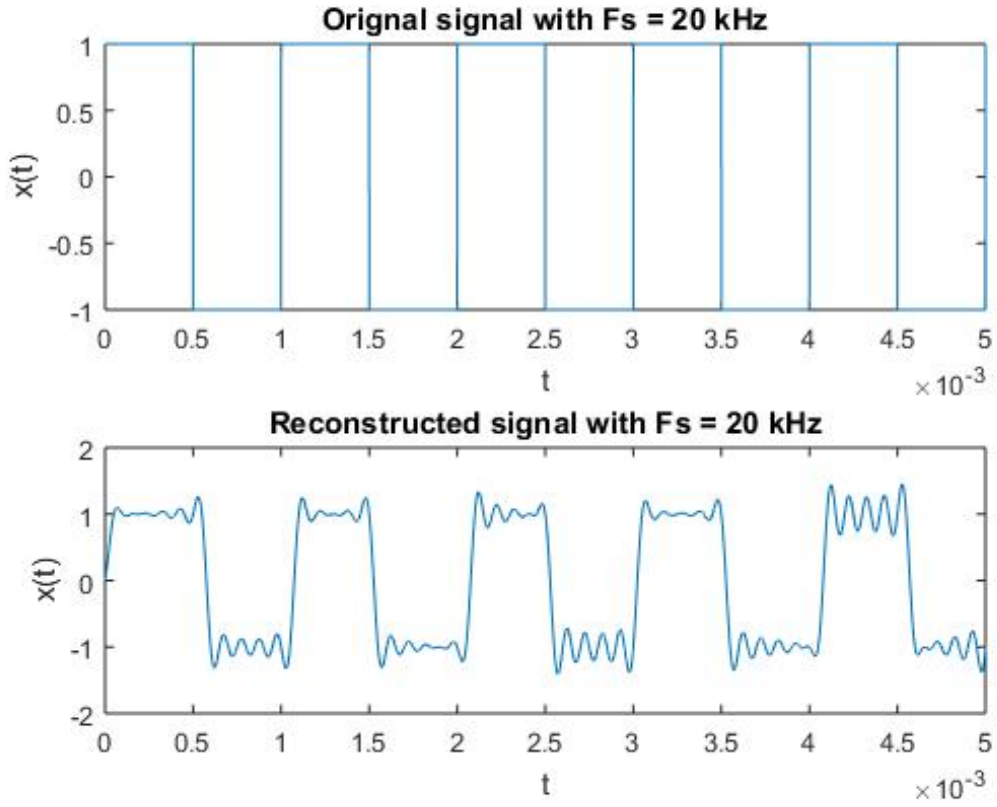

Figure 17: Square wave and its reconstructed version

3.3.2 Spectrum of the square wave:

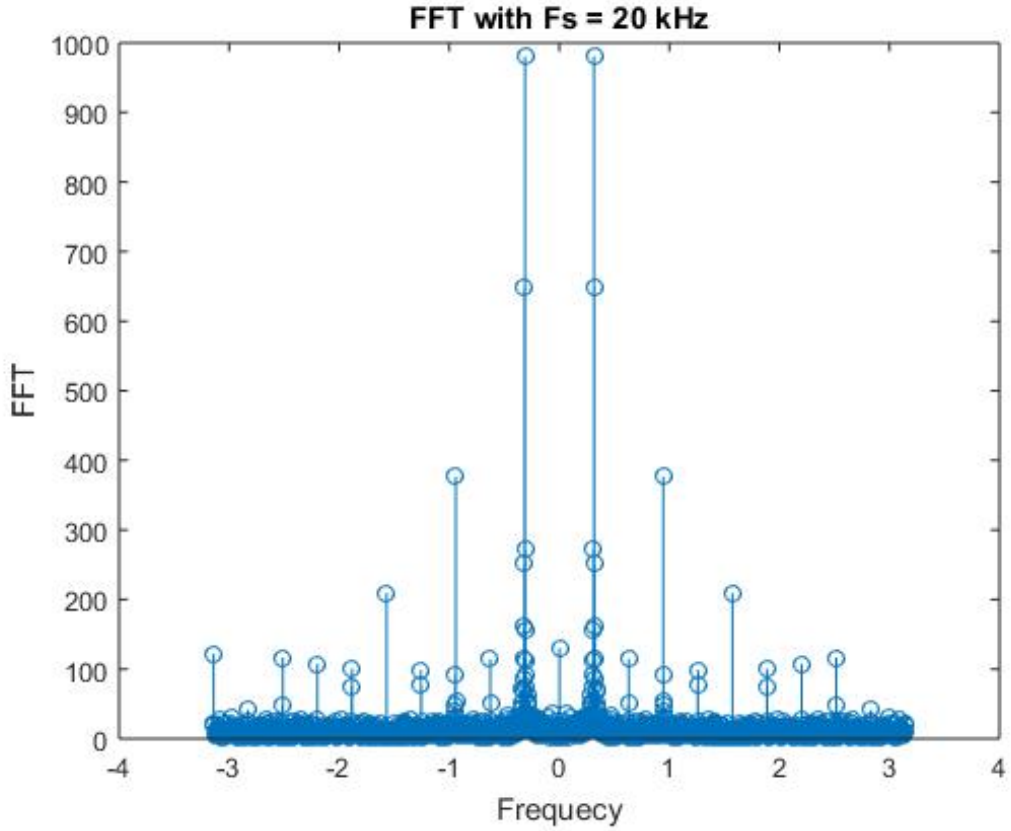

Figure 18: DFT of the square wave

3.3.3 Mean square error of the reconstructed wave:

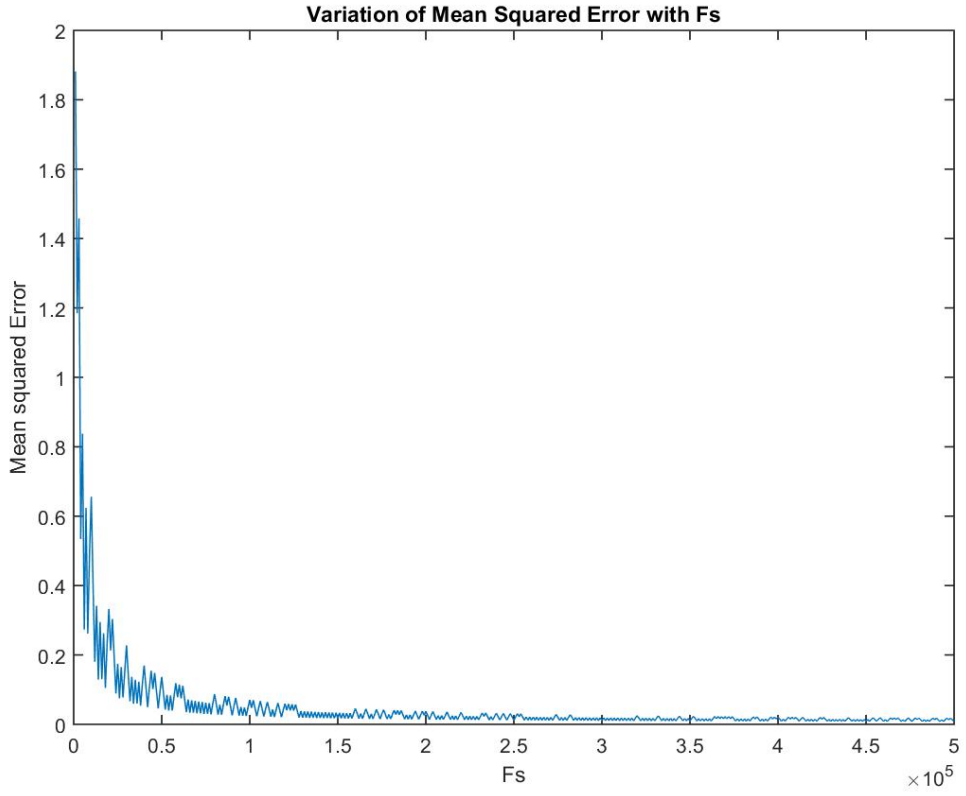

Figure 19: MSE VS Frequency

4 Part (d): Interpolation or Upsampling

4.1 Theory:

"Upsampling" is the process of inserting zero-valued samples between original samples to increase the sampling rate. This is called "zero-stuffing". Upsampling adds to the original signal undesired spectral images which are centered on multiples of the original sampling rate.

"Interpolation", is the process of upsampling followed by filtering. The filtering removes the undesired spectral images.

Thus upsampling consists of two operations:

- If the upsampling factor is L, initially, L-1 zeroes are inserted between samples in the time domain.This effectively scales time axis by a factor of L, and the frequency axis in the spectrum by a factor of 1/L.
- Filter the resulting upsampled sequence, in order to retrieve the original signal, proper choice of the filter leads to interpolation of the non-zero samples of the given signal, as it can remove all replicas of the signal except the baseband of the signal.

$$x_u[nL] = x[n] \tag{3}$$

and zero elsewhere.

$$X_u(j\Omega) = X(j\Omega L) \tag{4}$$

$$X_u(e^{j\omega}) = X(e^{j\omega L}) \tag{5}$$

Thus, the frequency spectrum has L copies of the baseband in the frequency range of (0,2π).

In this, part of the experiment, we apply zero order hold to both the originally sampled signal, and the upsampled signal. Zero order hold operates simply by maintaining the impulse value across the sample period. We did this because, zero order hold in frequency domain is multiplication with the sinc function, which is not a great filter but it somehow approximates the samples across the sample period.

4.2 Code Snippets:

```
clear all;
clc;

fs1=12e3;
fs2=24e3;
f=6e3;
nc=10;
f0=400e3;

t0=0:1/f0:nc/f;
x0=2*cos(2*pi*f*t0)+cos(pi*f*t0);

t=0:1/fs1:nc/f;
x=2*cos(2*pi*f*t)+cos(pi*f*t);

figure(1);

plot(t0,x0);
hold on;
stem(t,x);
title('Original signal');

figure(2);
nfft=1024;
z=fftshift(fft(x,nfft));
fval=fs1*(-nfft/2 : nfft/2-1)/nfft;
%subplot(3,2,3);
plot(fval, abs(z));
```

```
title('FFT of the sampled signal');

%% SAMPLE AND HOLD
figure(3);
%subplot(3,2,5);
stairs(t,x);
title('Sampled and held');

%% upsampling
figure(4);
z1=upsample(x,2);
t2=0:1/fs2:(length(z1)-1)/fs2;
%subplot(3,2,2);
plot(t0,x0);
hold on;
stem(t2,z1);
title('upsampled signal');

%% FFT OF THE UPSAMPLED SIGNAL

nfft=1024;
z3=fftshift(fft(z1,nfft));
fval=fs1*(-nfft/2 : nfft/2-1)/nfft;
%subplot(3,2,4);
figure(5);
plot(fval, abs(z3));
title('FFT of the upsampled signal');

%% SAMPLE AND HOLD
figure(6);
%subplot(3,2,6);
stairs(t2,z1);
title('Sampled and held');

%% LOW PASS FILTER


wn=(2/fs2)*f;
b=fir1(20,wn, 'low', kaiser(21,3));
%fvtool(b,1,'fs',fs2);

figure(7);
c=filter(b,1,z1);
subplot(2,1,1);
plot(c);
title('Signal reconstructed from the upsampled signal');

%% sampled at 24 khz
subplot(2,1,2);
t4=0:1/fs1:nc/f;
x4=2*cos(2*pi*f*t4)+cos(pi*f*t4);
plot(t4,x4);
title('Original signal sampled at 24kHz');
```

4.3 Figures and Plots:

4.3.1 A sample wave along with its reconstruction after upsampling

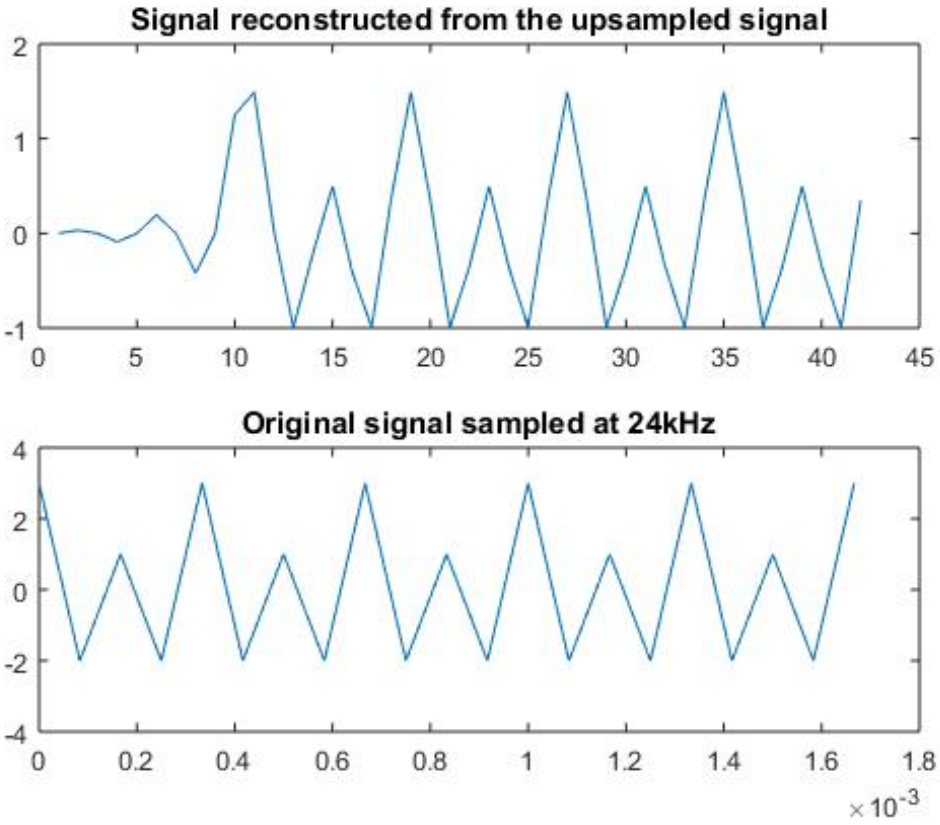

Figure 20: Signal and its reconstruction after upsampling

4.3.2 Signal, zero order hold and frequency spectrum: Before and after upsampling

Sampled waves:

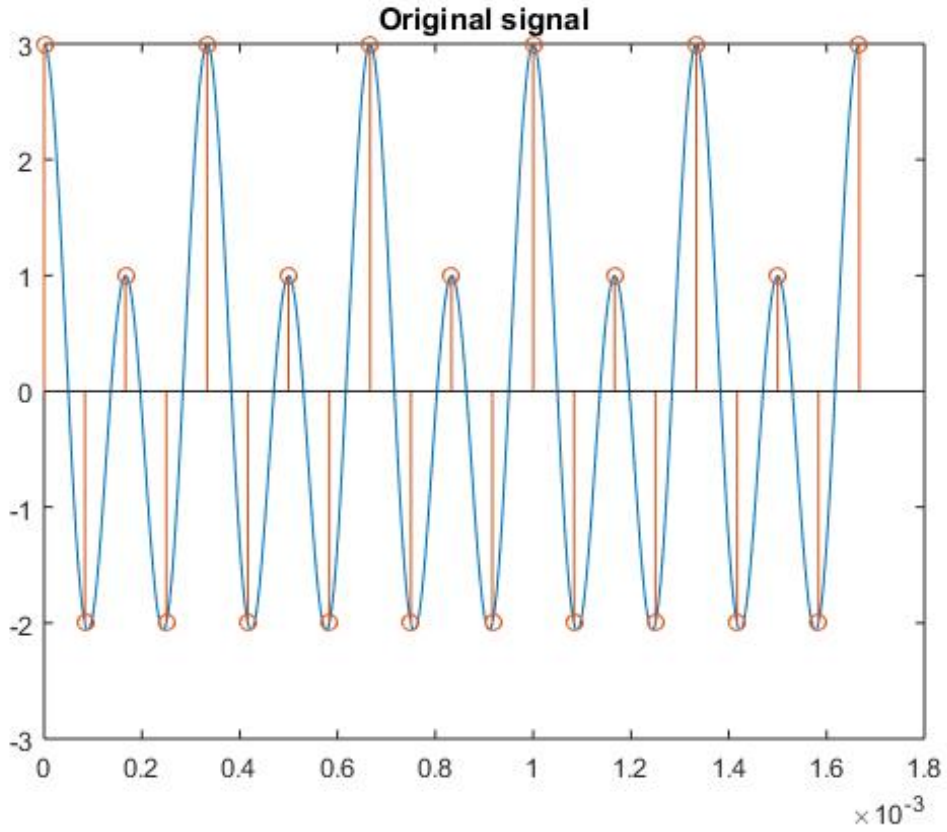

Figure 21: Original signal

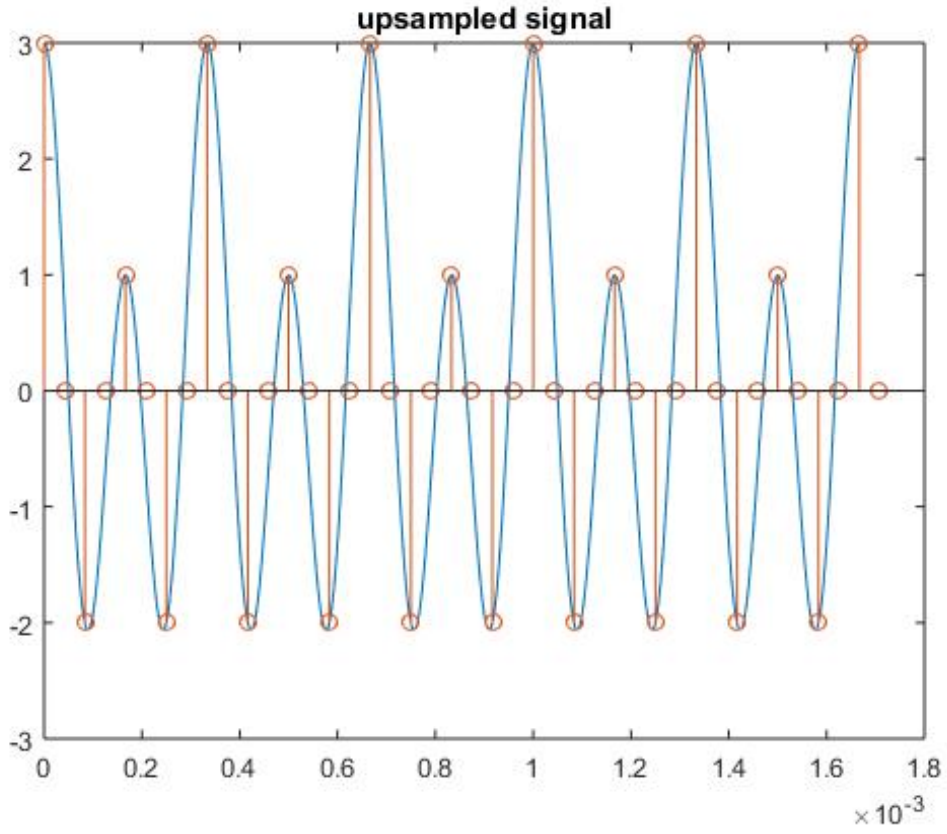

Figure 22: Upsampled signal

When sampled and held:

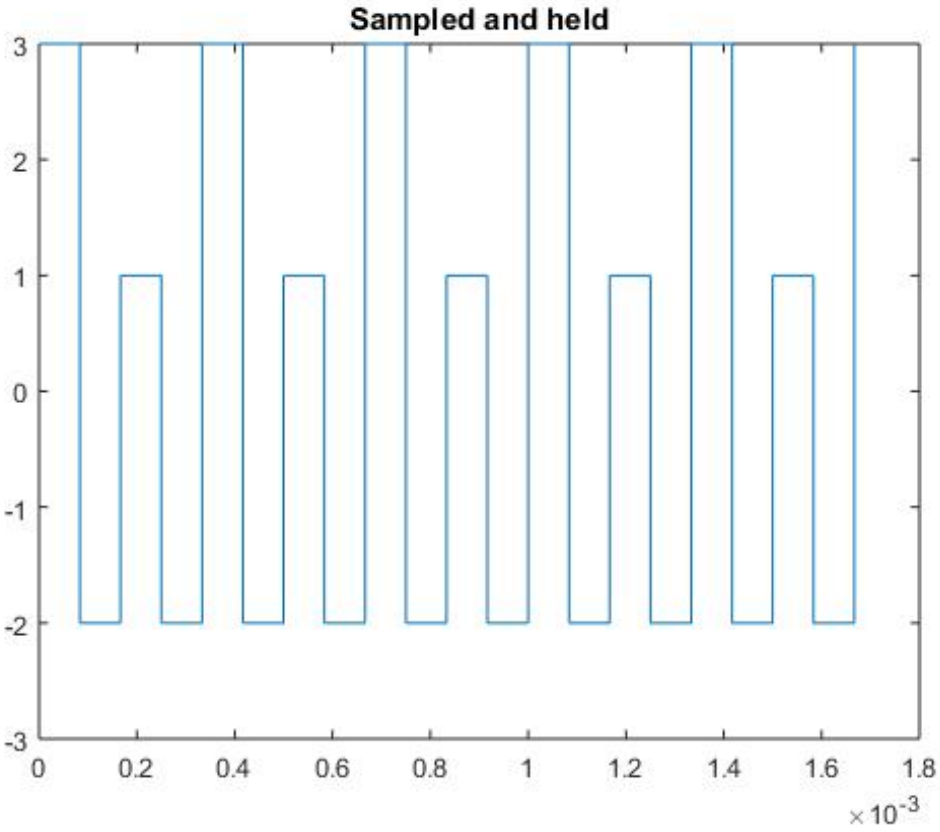

Figure 23: Original signal: Sampled and held

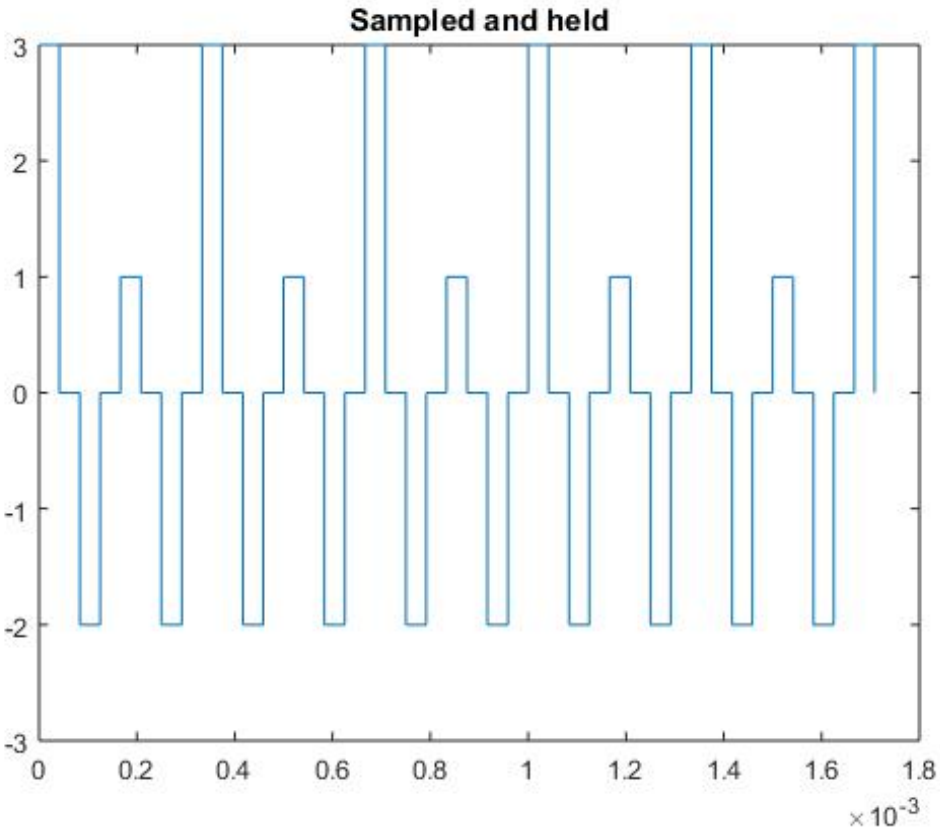

Figure 24: Upsampled signal: Sampled and held

Frequency spectrum:

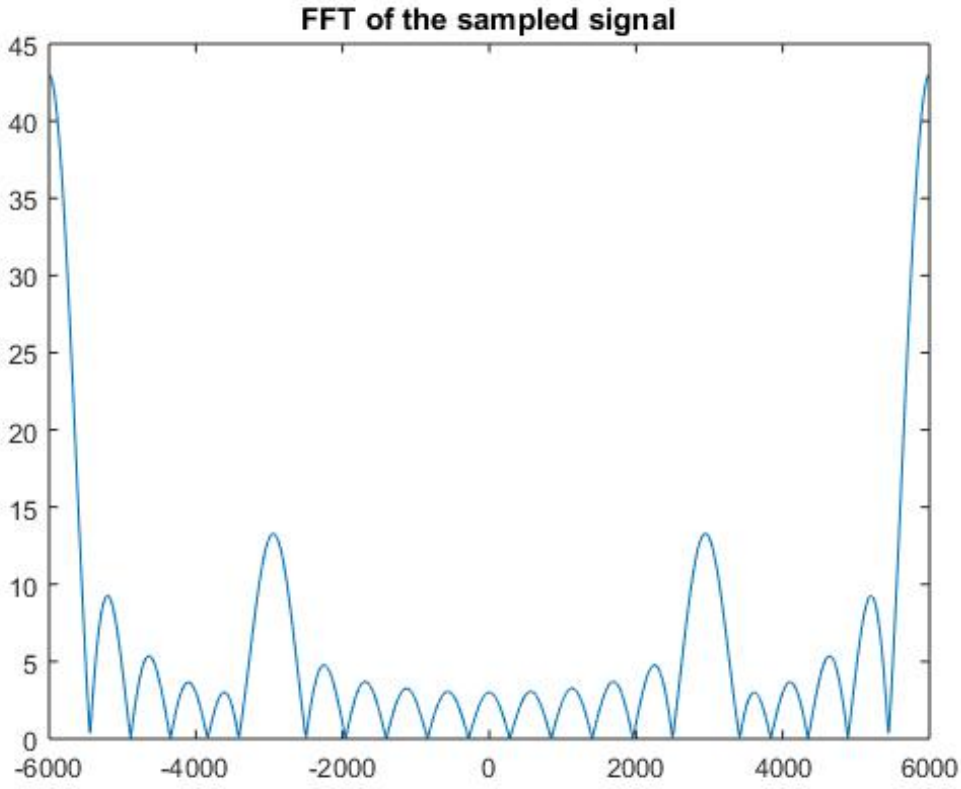

Figure 25: Frequency spectrum of the original signal

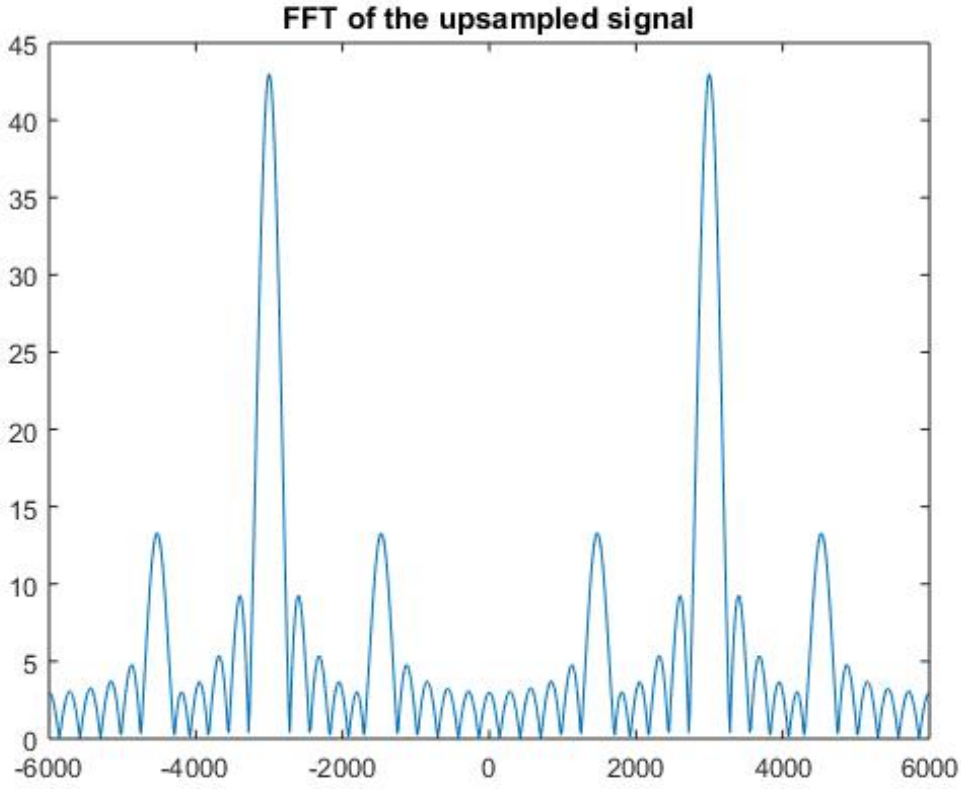

Figure 26: Frequency spectrum of the upsampled signal

5 Part (e): Bandpass Sampling

5.1 Theory:

The essence of bandpass sampling is that, it's the bandwidth of the signal that contains the information and not where the band resides about in the spectrum. Shannon-Nyquist Sampling Theorem is stated for baseband signals where the signal band is assumed to extend from zero hertz and up to its highest frequency (i.e. its bandwidth). The theorem then claims that the samples taken at twice this highest frequency will contain all the information of the baseband signal, so that those samples will represent the continuous signal completely, without a loss of information. Also that a perfect reconstruction of the continuous time signal from its samples is possible.

The bandpass sampling theorem which tells us, as expectedly, that since the information content of the new modulated bandpass signal is the same with the baseband original, the sufficient number of samples (per second) necessary to represent it perfectly is the same and determined by the bandwidth of the signal being sampled and not where specificaly it resides in the spectrum.

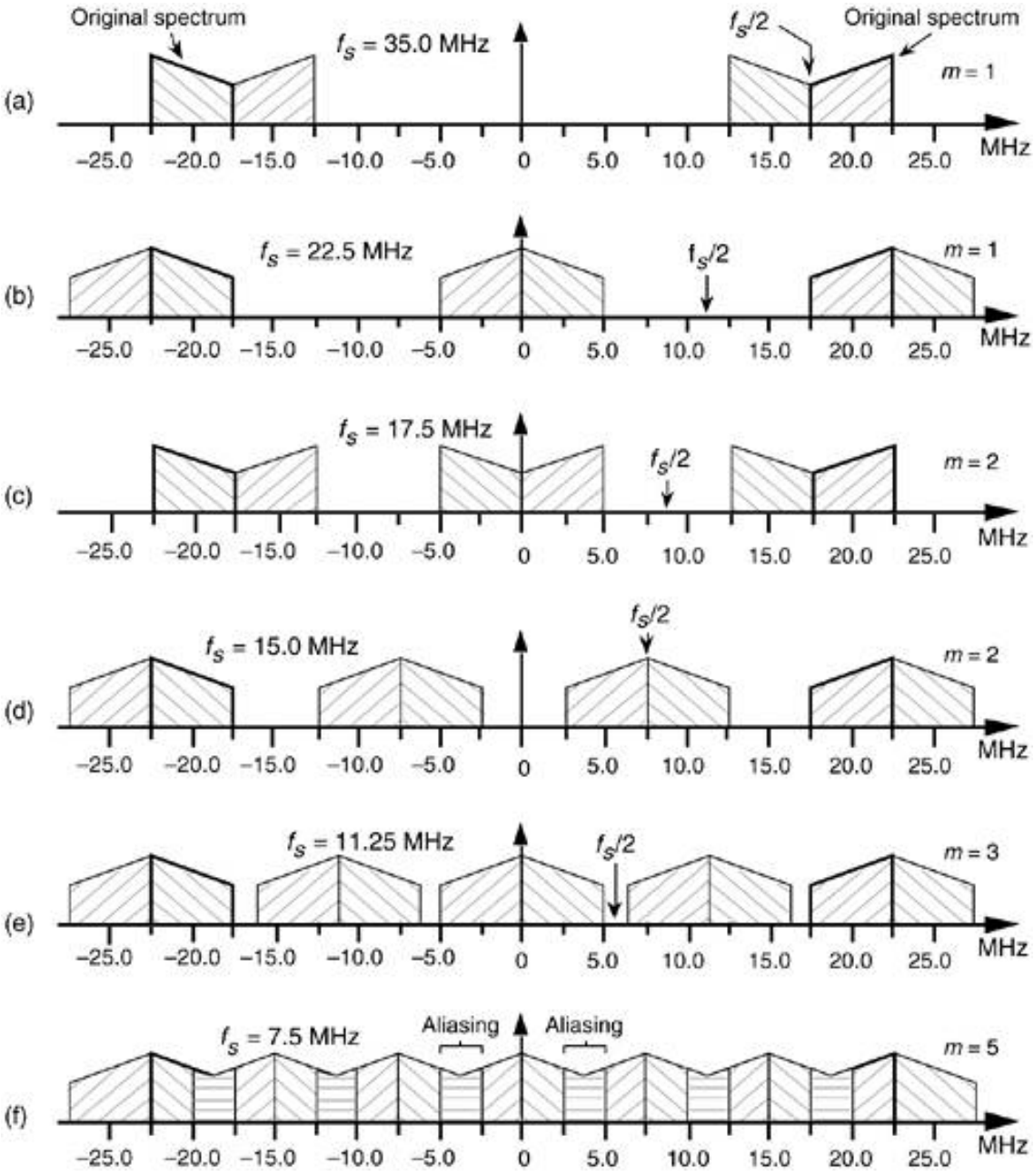

Figure 27: Frequency constraints for bandpass sampling

5.2 Code Snippets:

```
clear all;
clc;
f= 4e3;  %Frequency of sinusoid
fh=3*f;
fl=(2.7)*f;
fm=(fh+fl)/2;
bw=fh-fl;
bwg=(0.1)*bw;
kmax=floor(fh/bw);
fs1=fh/kmax;
fs=(2.8)*fs1;
fsa=30e3; % sampling rate 30 kHz


t=0:1/fs:511*1/fs; %time index
ta=0:1/fsa:511*1/fsa; %time index
L=length(t);
%x=10*cos(2*pi*f*t)+6*cos(2*pi*2*f*t)+2*cos(2*pi*3*f*t);
x=10*cos(2*pi*fl*t)+6*cos(2*pi*fm*t)+2*cos(2*pi*fh*t);
xa=10*cos(2*pi*fl*ta)+6*cos(2*pi*fm*ta)+2*cos(2*pi*fh*ta);


figure(1);
subplot(1,2,1)
plot(ta(1:120),xa(1:120))
title('Actual Signal(fs=30KHz)')
xlabel('Time(s)');
ylabel('Amplitude');
subplot(1,2,2)
plot(t(1:120),x(1:120));
title('Signal reconstructed after bandpass sampling')
xlabel('Time(s)');
ylabel('Amplitude');

%n1 = 64;
X =(fft(x));
Y =fftshift(X);
m = abs(Y);
F=-fs/2:fs/L:fs/2-fs/L;

Xa =(fft(xa));
Ya =fftshift(Xa);
ma = abs(Ya);
Fa=-fsa/2:fsa/L:fsa/2-fsa/L;

figure(2);
subplot(2,1,1);
plot(Fa,2*ma/L)
title('DTFT of the actual signal (fs=30KHz)')
xlabel('Frequency (Hz)');
ylabel('Magnitude of FFT');
subplot(2,1,2);

plot(F,2*m/L)
title('DTFT of the signal obtained after bandpass sampling')
xlabel('Frequency (Hz)');
ylabel('Magnitude of FFT');

%fs=(2.5)*fs1;
%fs=12e3; % sampling rate 12 kHz
```

5.3 Figures and Plots:

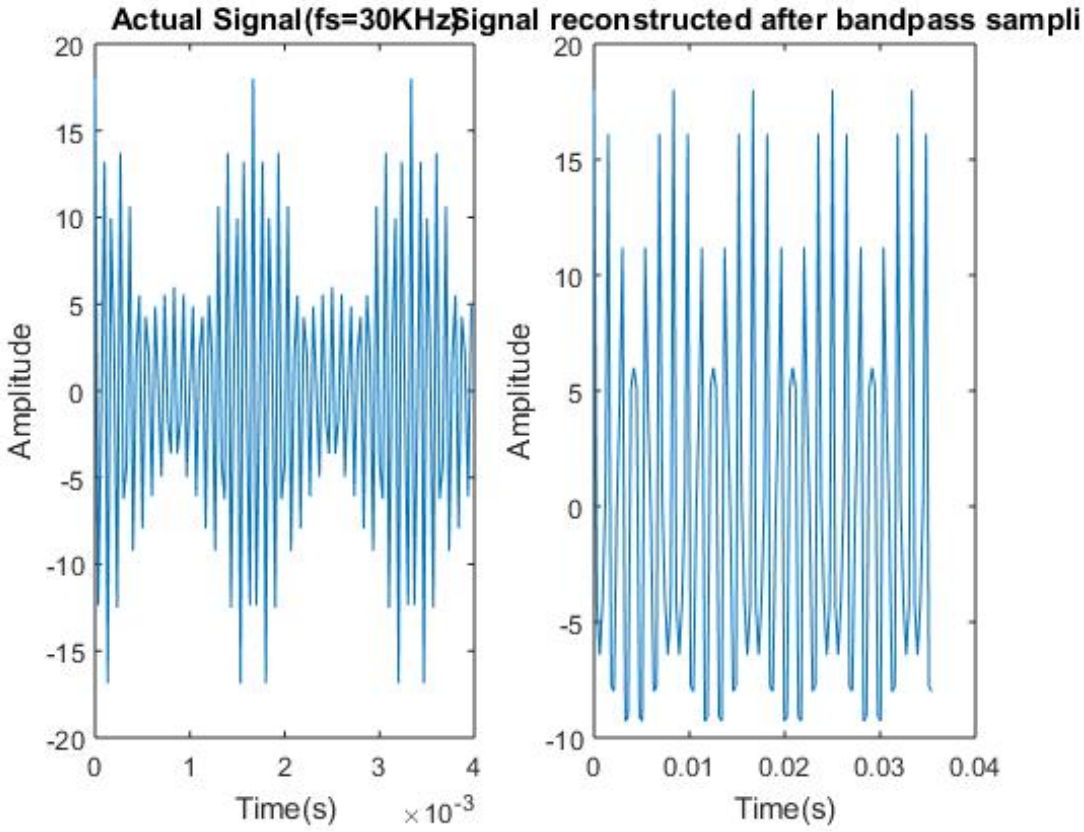

Figure 28: Original signal and reconstructed signal after bandpass sampling

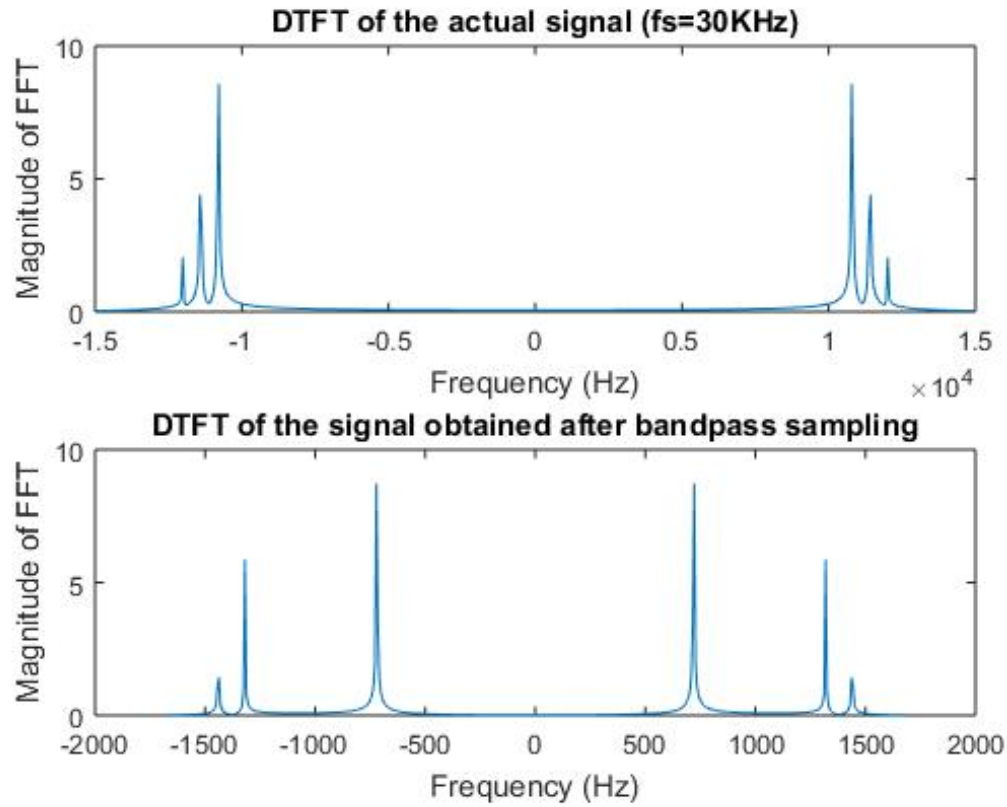

Figure 29: Frequency spectrum of the original signal and reconstructed signal after bandpass sampling

6 Part (f): Interpolation or up-sampling of an audio signal

6.1 Theory:

If an analog signal is sampled at a frequency higher than the Nyquist rate, it is possible to interpolate the intermediate L-1 samples or in other words to obtain the samples at Fs2 = LFs1 frequency. This can be simply done by passing the sampled signals through an ideal low pass filter of cut-off frequency Fmax and sampling it again at a higher rate. In discrete time domain it can be achieved as shown.

6.2 Code Snippets:

```
close all;
clc;
clear;

%% Upsampling of audio signal

N = 512;

[x Fs] = audioread('audio48kHz.wav');
x_2 = upsample(x,2);

f1 = linspace(-1*pi, pi, N);
f2 = linspace(-1*pi, pi, 2*N);

figure;
subplot(1,2,1);
plot(f1, abs(fftshift(fft(x,N))));
title('FFT of orignal/sampled signal');
xlabel('frequency') % x-axis label
ylabel('FFT') % y-axis label

subplot(1,2,2);
plot(f2, abs(fftshift(fft(x_2, 2*N))));
title('FFT of upsampled signal');
xlabel('frequency') % x-axis label
ylabel('FFT') % y-axis label

%% generation of low pass filter

filter = zeros(1, 2*N);
for i = 1 : N/2
    filter(i) = 1;
end
for i = 3*N/2 + 1 : 2 * N
    filter(i) = 1;
end

%% Passing through low pass filter
fft_2 = fft(x_2,2*N).*transpose(filter);

%% Reconstructed signal and comparision with orignal sampled

figure;

s_time = 1: 2 : 2*N;
u_time = 1 : 1 : 2*N;

subplot(2,1,1);
plot(s_time,x(1:N))
```

```
title('Orignal signal sampled at 48 kHz');
xlabel('t') % x-axis label
ylabel('x(t)') % y-axis label

subplot(2,1,2);
plot(u_time,real(ifft(fft_2)), 'r');
title('Reconstructed signal upsampled');
xlabel('t') % x-axis label
ylabel('x(t)') % y-axis label

%% comparision of sounds by listening
sound(x(1:N)) %orignal signal
pause(3) % Pause for some time
sound(real(ifft(fft_2))) % regenerated signal
```

6.3 Figures and Plots:

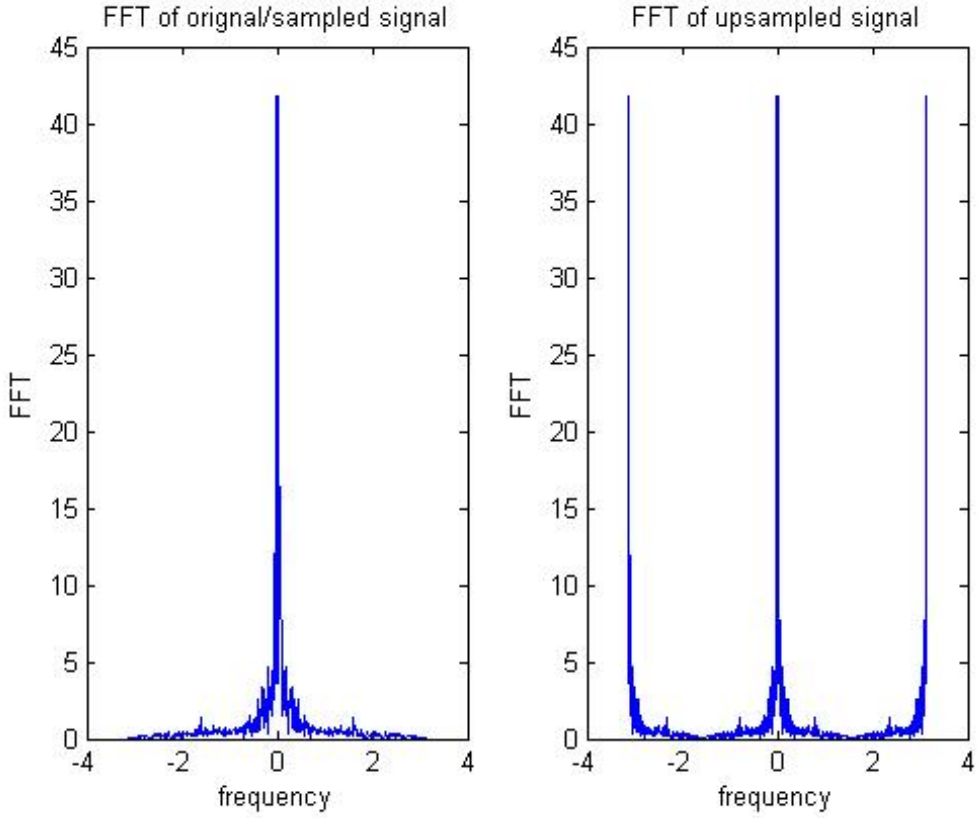

Figure 30: FFT of original and upsampled signals

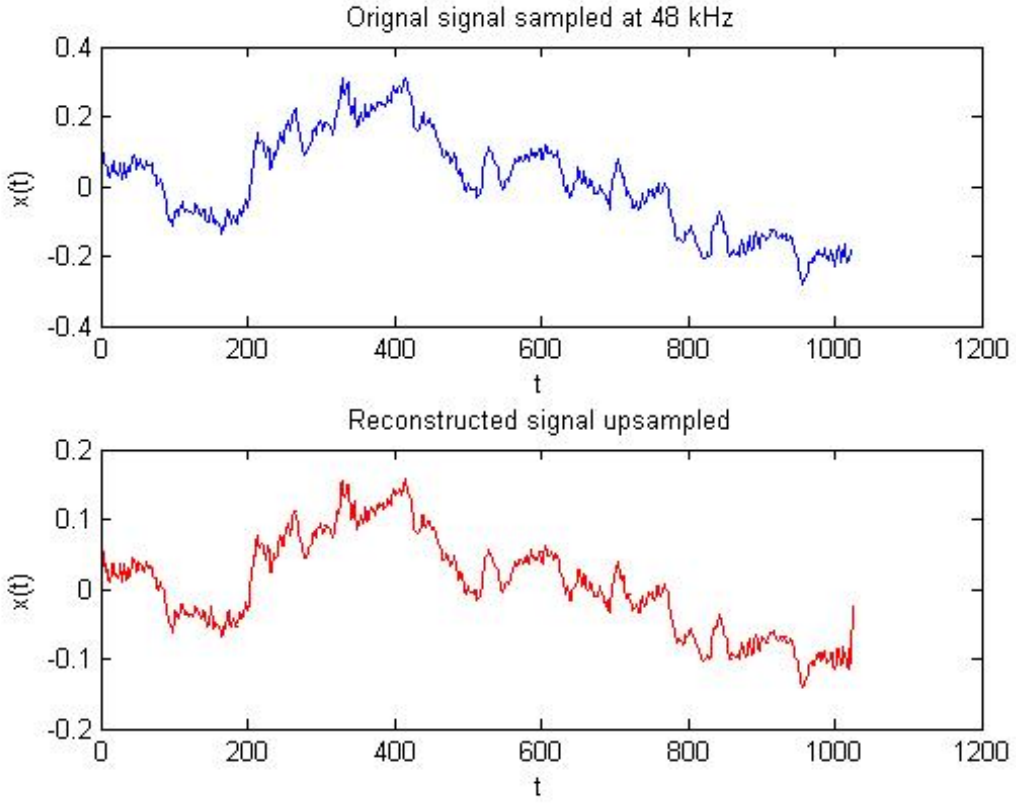

Figure 31: Original signal and reconstructed signal

7 Discussions:

- In the first part of the experiment, aliasing can be observed at frequencies lower than 8kHz, as the bandwidth of the original signal is 4kHz.
- Giving the FFT a close look, overlapping can be observed. This overlapping is the reason we have aliasing in the reconstructed signal. Hence, we must ensure that sampling is done at frequency higher than twice the maximum frequency of the original signal.
- The square wave has infinite bandwidth, so, exact reconstruction is not possible, but, the reconstructed signal quite approaches the original signal as the sampling rate is increased gradually. To reconstruct the signal, the FFT has been interpolated with the sinc function, which can act as a low pass filter.
- As expected, the mean squared error between the original signal and the reconstructed signal decreases with increasing sampling frequency. However, the curve plotted is not smooth, because of the smaller value of N, it could get further smoother on increasing N.
- When upsampled by a factor of L, it leads to a generation of L-1 zeroes. Looking at FFTs of the original signal and the upsampled version, we see, the FFT has been compressed. Thus, a low pass filter with quite lower cut off frequency can be used to reconstruct the original signal. The reconstructed signal is just a scaled and amplified version of the original signal. To, reconstruct back, the signal, the spectrum is multiplied with a low pass filter.
- Interpolation is equivalent to applying a low-pass filter to remove the high frequency replica's created by shrinking in frequency. In time, this removes all of the extra zeros and smooths the signal. To remove the high frequency components, the cut-off frequency of an ideal reconstruction/interpolation filter would ideally be $\Omega c = \pi/M$, where M is the downsampling factor and the gain would be M