Sampling of a Sinusoidal Waveform

AIM:

- To sample an Analog signal waveform above its Nyquist sampling rate.
- To obtain DFT of Analog waveform

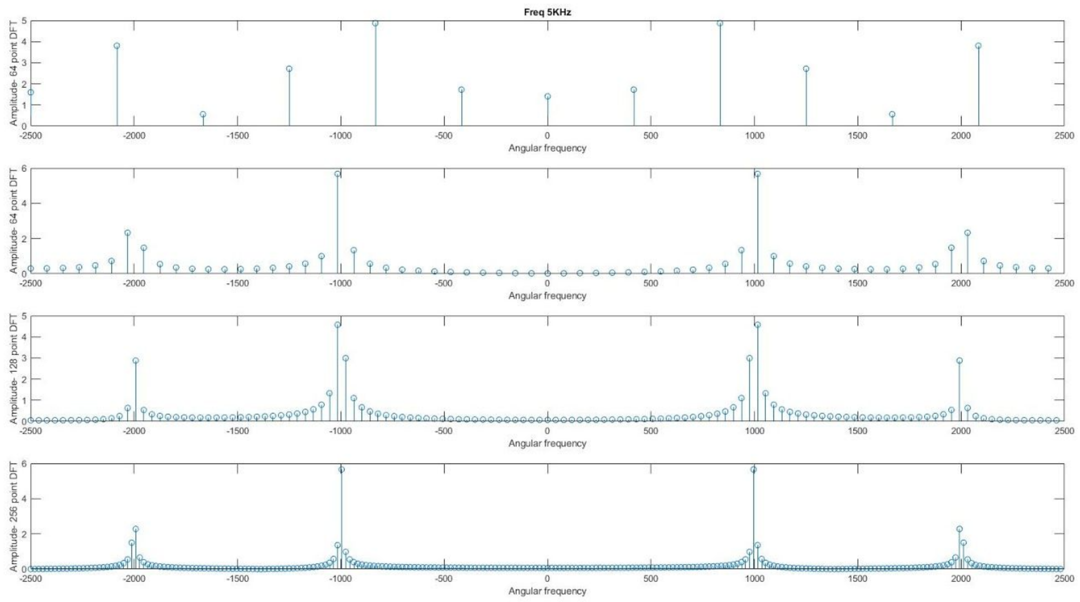
MATLAB FUNCTION USED

fftshift, fft, cos, subplot, stem, abs

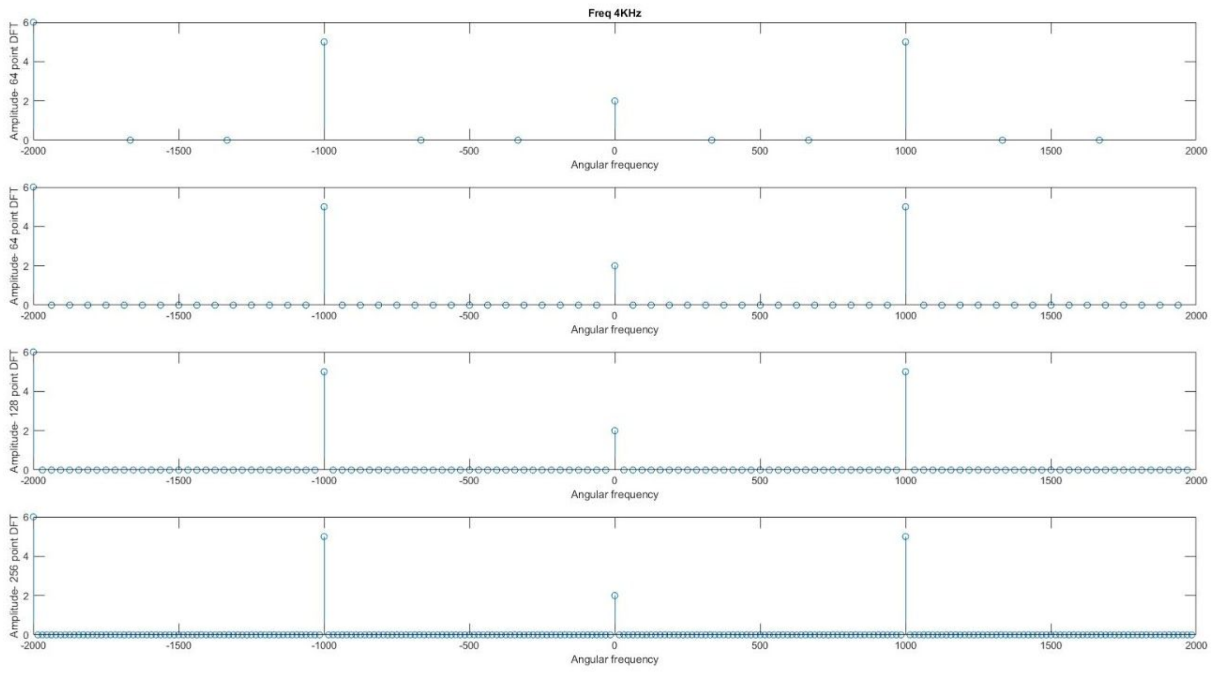
THEORY

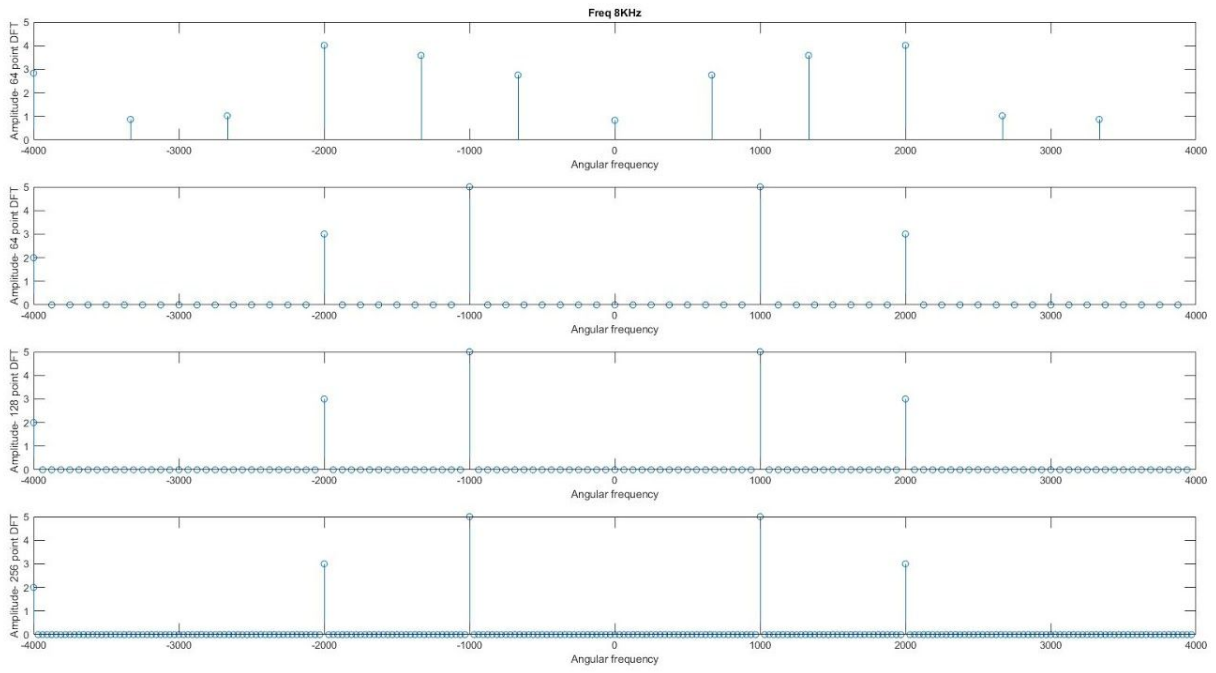
The Nyquist Theorem, also known as the sampling theorem, is a principle that engineers follow in the digitization of analog signals. For analog-to-digital conversion (ADC) to result in a faithful reproduction of the signal, slices, called *samples*, of the analog waveform must be taken frequently. The number of samples per second is called the sampling rate or sampling frequency. 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.

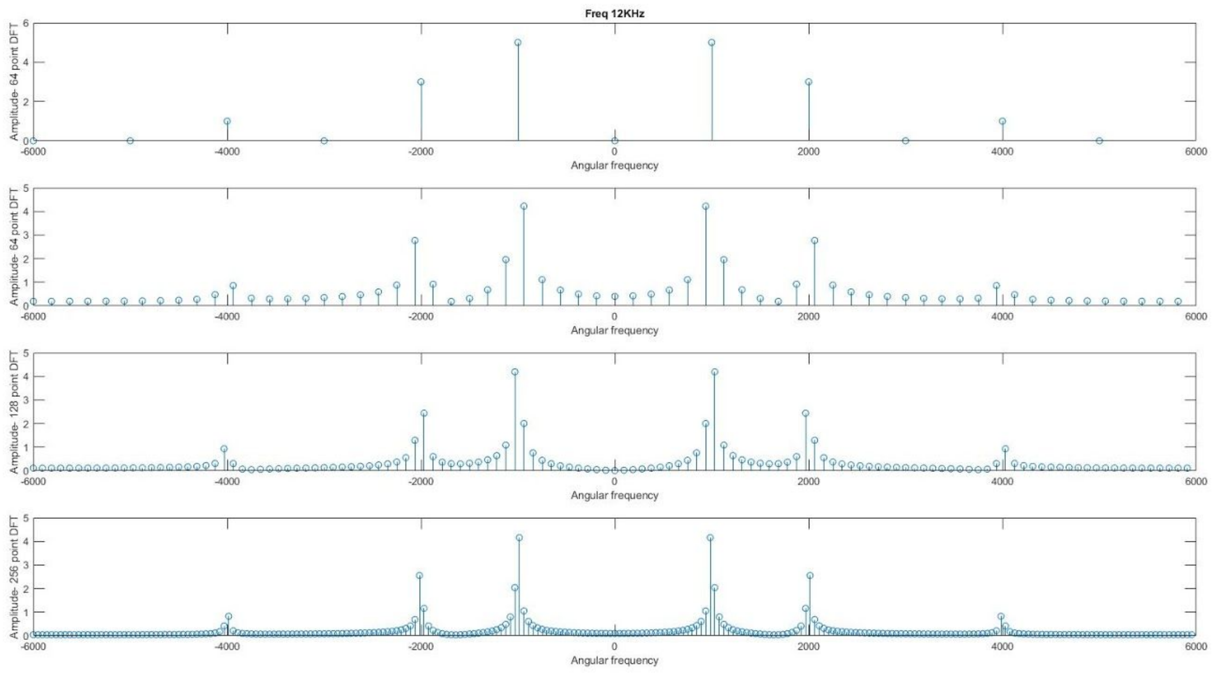
SOURCE CODE AND RESULTS



```
Fs = 5e3;
t = -1:1/Fs:1;

wave = 10*cos(2*pi*1e3*t) + 6*cos(2*pi*2e3*t) +2*cos(2*pi*4e3*t);
figure();

subplot(4,1,1);
%plot(wave)
Y=abs(fftshift(fft(wave',12)))/12;
df = Fs/length(Y);
freqvec = -Fs/2:df:Fs/2-df;
stem(freqvec,abs(fftshift(fft(wave',12)))/12);
xlabel('Angular frequency');
ylabel('Amplitude- 64 point DFT');

title(['Freq ',num2str(Fs/1e3),'KHz']);


subplot(4,1,2);
%plot(wave)
Y=abs(fftshift(fft(wave',64)))/64;
df = Fs/length(Y);
freqvec = -Fs/2:df:Fs/2-df;
stem(freqvec,abs(fftshift(fft(wave',64)))/64);
xlabel('Angular frequency');
ylabel('Amplitude- 64 point DFT');


subplot(4,1,3);
Y=abs(fftshift(fft(wave',128)))/128;
df = Fs/length(Y);
freqvec = -Fs/2:df:Fs/2-df;
stem(freqvec,abs(fftshift(fft(wave',128)))/128);

xlabel('Angular frequency');
ylabel('Amplitude- 128 point DFT');



subplot(4,1,4);

Y=abs(fftshift(fft(wave',256)))/256;
df = Fs/length(Y);
freqvec = -Fs/2:df:Fs/2-df;
stem(freqvec,abs(fftshift(fft(wave',256)))/256);

xlabel('Angular frequency');
ylabel('Amplitude- 256 point DFT');
```


Freq 4KHz
Amplitude- 64 point DFT
Amplitude- 64 point DFT
Amplitude- 128 point DFT
Amplitude- 256 point DFT
Angular frequency


Freq 8KHz
Amplitude- 64 point DFT
Amplitude- 64 point DFT
Amplitude- 128 point DFT
Amplitude- 256 point DFT
Angular frequency



DISCUSSION

In these 2 cases the sampling frequencies are less than 2*(fmax) thus they do not satisfy Nyquist Theorem. In this case we can clearly see the effects of aliasing. In 8khz sampling we see a peak at 0 Hz in the DFT plot. This is because when we are sampling at lower frequencies the higher frequency parts of the signal are not correctly sampled which leads to incorrect frequency components. More accurately this is due to the overlapping of the periodic copies of the baseband signal. The effect was clearly visible for 4 KHz and 5 KHz.

Spectrum of a Square Wave

AIM:

- To observe the Spectrum of a square wave

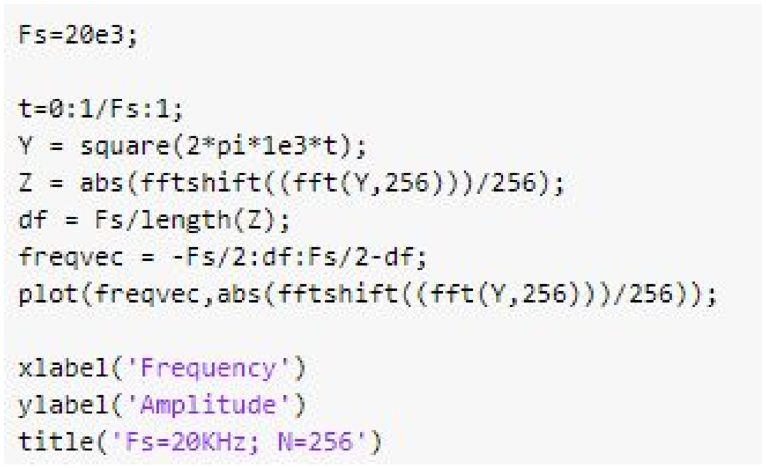
MATLAB FUNCTION USED

fftshift, fft, square, subplot, stem, abs

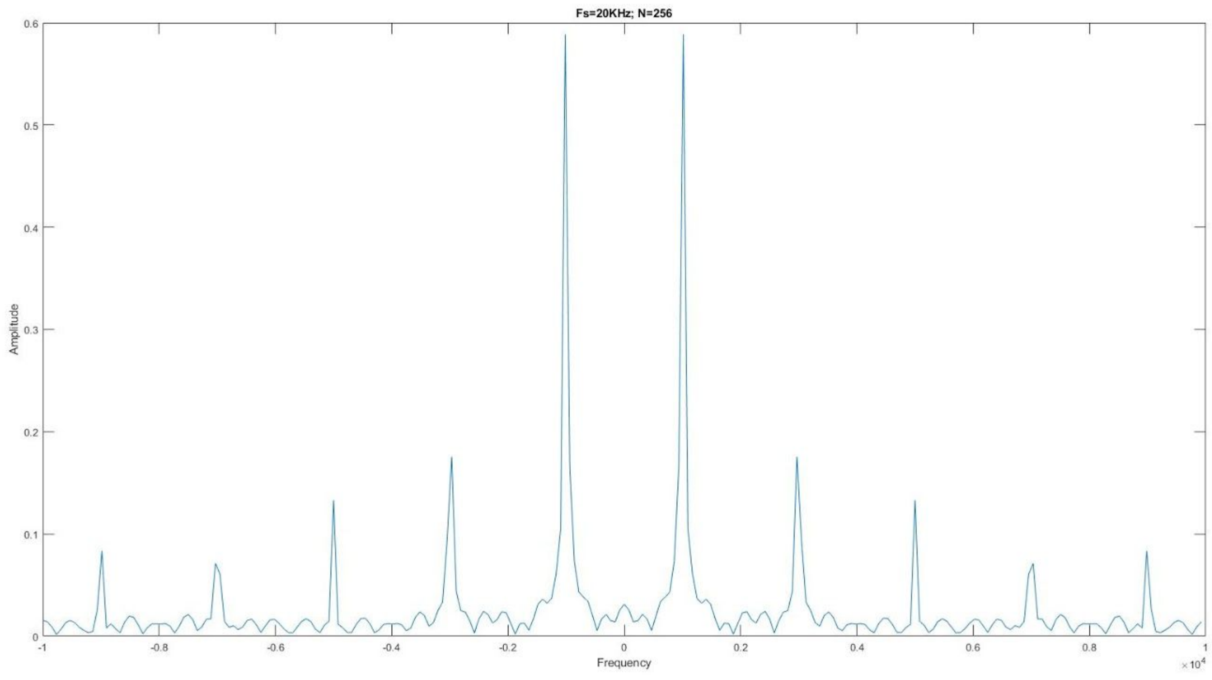
THEORY

A square wave has theoretically has infinite bandwidth. For practical purposes, the spectrum beyond 10th harmonic can be neglected.

SOURCE CODE AND RESULTS



```
Fs=20e3;

t=0:1/Fs:1;
Y = square(2*pi*1e3*t);
Z = abs(fftshift((fft(Y,256)))/256);
df = Fs/length(Z);
freqvec = -Fs/2:df:Fs/2-df;
plot(freqvec,abs(fftshift((fft(Y,256)))/256));

xlabel('Frequency')
ylabel('Amplitude')
title('Fs=20KHz; N=256')
```



DISCUSSION

A square wave has theoretically infinite bandwidth so in this case we are considering uptill the 10th harmonic. This can be done as we can see from the spectrum that the amplitude of the higher harmonics keep on decreasing. Also all the amplitudes of the even harmonics are zero. Only the odd harmonics exists. Their amplitude can be represented as $A_n = 2/(\pi*n)$. We observe that higher frequency components are smaller in magnitude. But this smaller components give the mean square error when we construct the signal from its discrete Fourier transform.

Interpolation or Up sampling

AIM:

- To interpolate an Analog Signal.

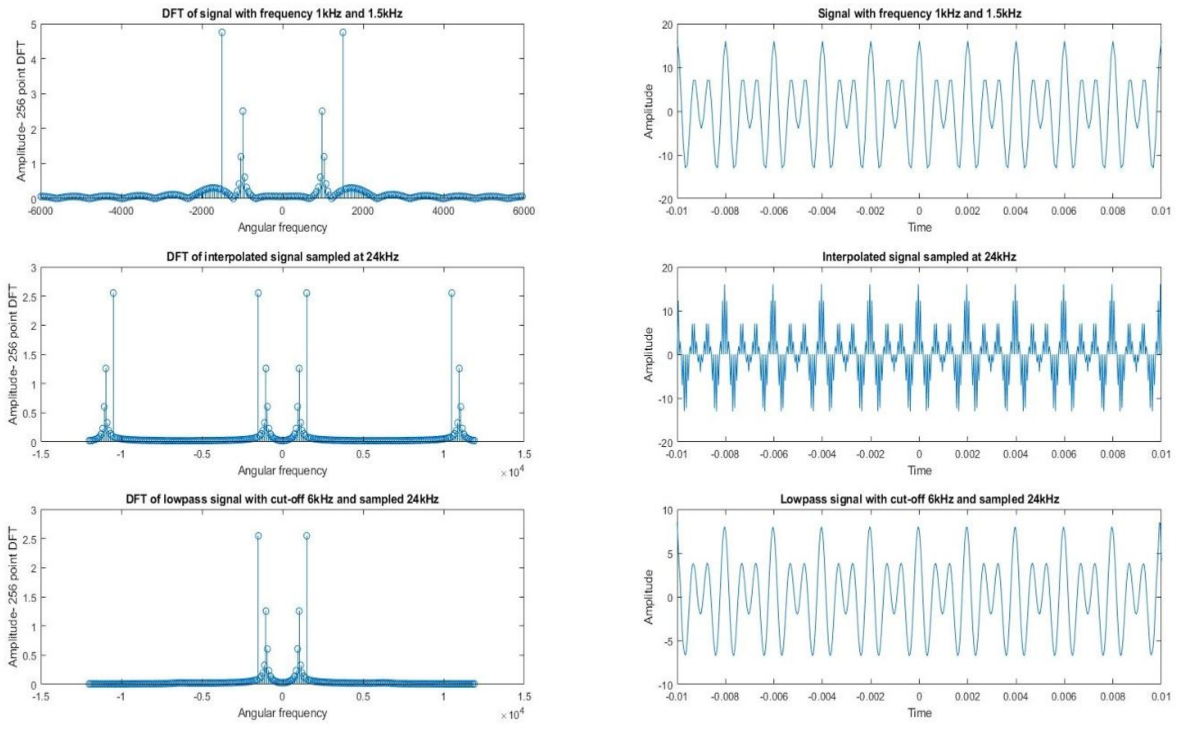
MATLAB FUNCTION USED

fftshift, fft, interpol, subplot, stem, abs, upsample, lowpass

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.

SOURCE CODE AND RESULTS



```matlab
F = 1e3;
wave = [6 10]*cos(2*pi*[F 1.5*F]'.*t);
interpol = upsample(wave, 2);

lowpass_signal = lowpass(interpol, 6e3, 24e3);

%[z,p] = butter(2,0.5,'low');
%[z,p] = cheby2(2,40,0.5,'low');
%output = filter(z,p,interpol);


subplot(321);
Y=abs(fftshift(fft(wave',256)))/256;
df = Fs/length(Y);
freqvec = -Fs/2:df:Fs/2-df;
stem(freqvec,Y);
xlabel('Angular frequency');
ylabel('Amplitude- 256 point DFT');
title('DFT of signal with frequency 1kHz and 1.5kHz');

subplot(322);
plot(t,wave);
xlabel('Time');
ylabel('Amplitude');
title('Signal with frequency 1kHz and 1.5kHz');

Fs = 24e3;
t = -1e-2:1/Fs:1e-2;

subplot(323);
Y=abs(fftshift(fft(interpol',256)))/256;
df = Fs/length(Y);
freqvec = -Fs/2:df:Fs/2-df;
stem(freqvec,Y);
xlabel('Angular frequency');
ylabel('Amplitude- 256 point DFT');
title('DFT of interpolated signal sampled at 24kHz');

subplot(324);
plot(t,interpol(2:end));
xlabel('Time');
ylabel('Amplitude');
title('Interpolated signal sampled at 24kHz');

subplot(325);
Y=abs(fftshift(fft(lowpass_signal',256)))/256;
df = Fs/length(Y);
freqvec = -Fs/2:df:Fs/2-df;
stem(freqvec,Y);
xlabel('Angular frequency');
ylabel('Amplitude- 256 point DFT');
title('DFT of lowpass signal with cut-off 6kHz and sampled 24kHz');

subplot(326);
plot(t,lowpass_signal(2:end));
xlabel('Time');
ylabel('Amplitude');
title('Lowpass signal with cut-off 6kHz and sampled 24kHz');
```

DISCUSSION

During up-sampling, we insert zero in between samples which causes duplication of lower frequencies in higher band. We need filter to filter out higher frequency range to have our original signal upsampled. Here, in the case we have used brick wall filter for digital filtering.

Aadi Swadipto Mondal
17EC10065