**MEEM 5700 Dynamic Measurements and Signal Analysis**

**10/03/2013**

**Windows and the FFT**

**Assignment 03**

**Arjun Darbha**

Instructor: Dr. Jason Blough

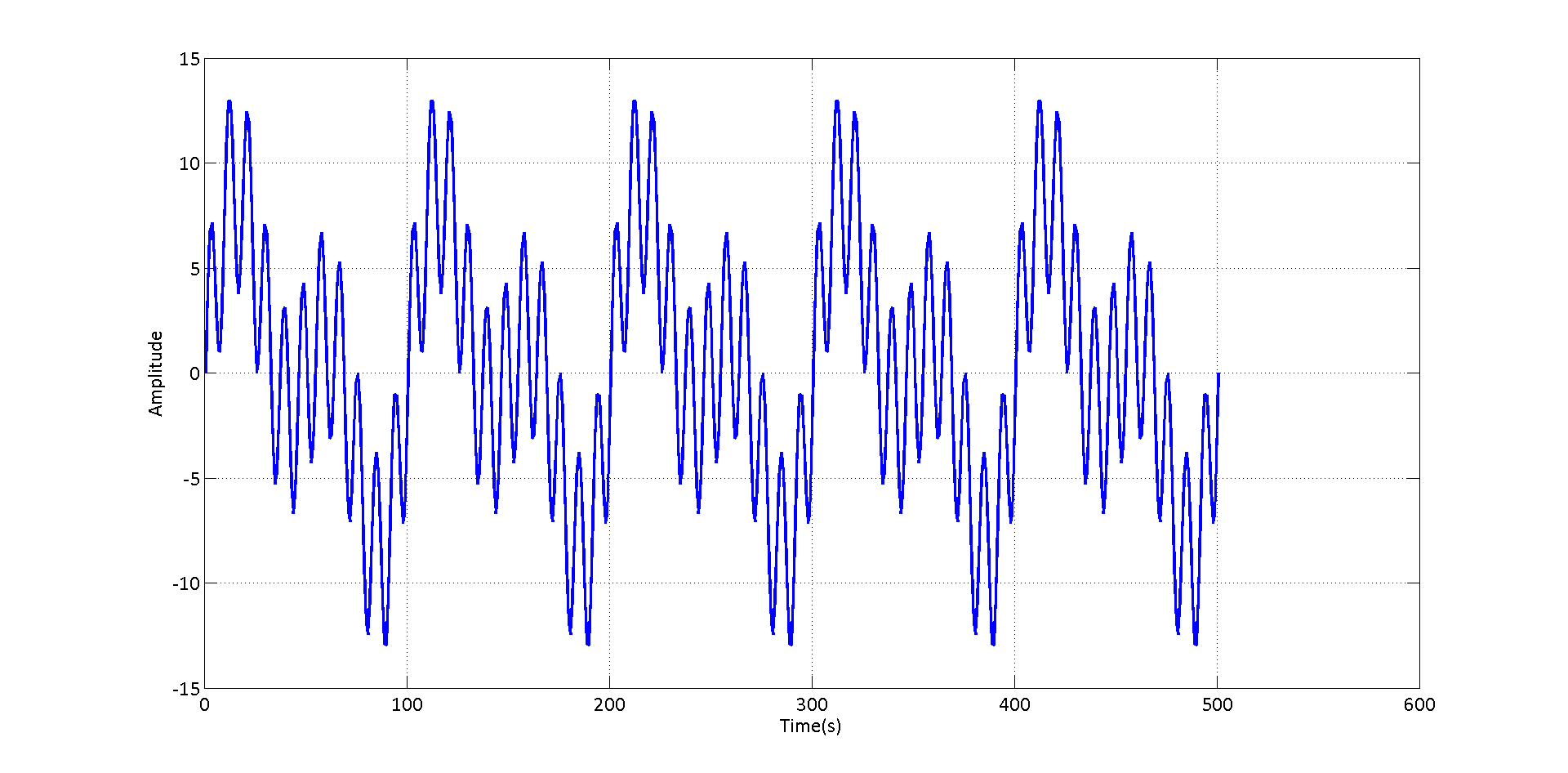
TA: Craig Reynolds

Abstract

*The purpose of this experiment is to perform a spectral analysis on the data and see the effects of leakage. Data is collected through a microphone which captures a known signal through a speaker. A discrete Fourier transform technique called a Fast Fourier Transform (FFT) is used to obtain the spectrum. It has been observed that it is possible to discern the exact frequency from the spectrum only for particular sets of data. These sets of data vary in the number of observations which is equivalent to time for which the measurement was recorded. To overcome the effects of leakage, various windows have been utilized in order to establish the effectiveness of each of these windows over the data. These effects have been examined on the data collected from laboratory and also on the data generated in MATLAB. Furthermore, it can be concluded that, rectangular window provides maximum frequency resolution whereas; flattop window provides maximum amplitude resolution in the frequency spectrum.*

BACKGROUND AND OBJECTIVES

Signals captured in time domain do not reveal all the information a signal possesses. It is required to transform this signal into frequency domain to analyze the signal comprehensively. This is achieved by performing a Fourier transform on the data. According to Fourier, a French mathematician, every periodic signal can be represented by a series of sine and cosine waves. Consider a periodic signal, with different components as shown by figure 1.



**Figure 1 Periodic signal with different components**

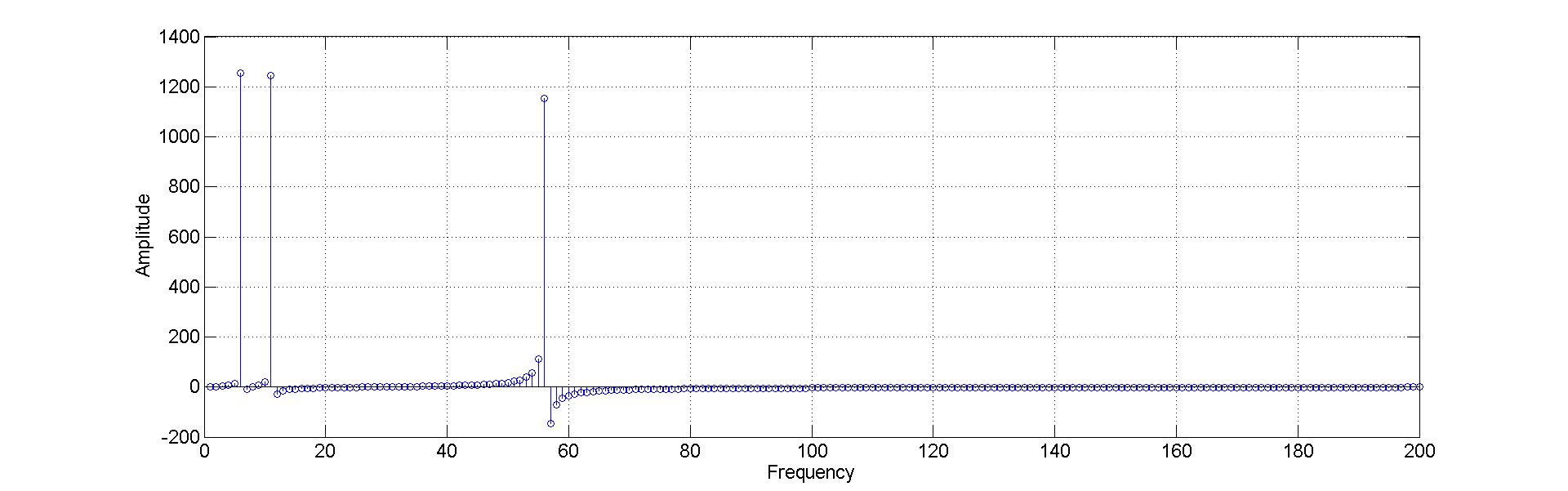
According to Fourier, any such signal can be represented by a series of sine and cosine waves as equation 1 describes

|  |  |  |
| --- | --- | --- |
|  |  |  |

where,

and T is the time period of a wave-form

Performing a Fourier transform on the signal above reveals that it is obtained from a series of three sine-waves of 5Hz, 10Hz and 50Hz as shown in figure 2.



**Figure 2 Imaginary part of the spectrum obtained by performing an FFT on the signal represented by figure 1**

The result of an FFT can be both real and imaginary. All the sine components are represented by imaginary parts of the spectrum and the cosine components are represented by real parts of the spectrum. This can be deduced from the kernel transform of a fourier transform represented by equation 2

|  |  |  |
| --- | --- | --- |
|  |  |  |

In order to evaluate a Fourier transform computationally, exponential form of equation 2, denoted by equation 3, is used. This equation is obtained using Euler’s formulas

|  |  |  |
| --- | --- | --- |
|  |  |  |

As T tend to ∞ deltaT tends to zero, hence equation 3 is transformed into

|  |  |  |
| --- | --- | --- |
|  |  |  |

This integral can be evaluated to obtain a fourier integral transform pair

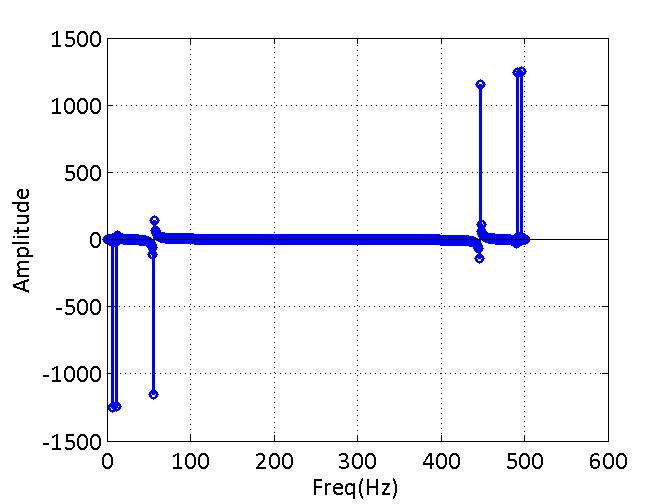
|  |  |  |
| --- | --- | --- |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  |  |  |

It is evident that equation 5 is a time to frequency transformation whereas, equation 6 is a frequency to time transformation. Equation 6 is also known as inverse fourier transform. It is not possible to evaluate integrals on a computer. Hence, a computer-solvable form of equation 6 need to be obtained in order to perform a transform. Also equation 5 suggests that the data be taken continuously which is not practical. Hence equation 5 is evaluated for a single time period T and the using a finite number of points-N using equation 7. N is also called the block-size.

|  |  |  |
| --- | --- | --- |
|  |  |  |

Computational evaluation of equation 7 results in a spectrum that is evaluated for N number of points and a duplicate frequency spike is obtained at about N/2, the amplitudes being halved. This situation is represented in figure 3. FFT is performed on the signal denoted by figure 1 using equation 7 to obtain figure 3.



**Figure 3 FFT results obtained without scaling**

Figure 3 is an appropriate mathematical representation of the signal. However, energy of each of the frequency is collectively represented by two frequencies which are 180 degrees apart. Also, it is known that when a signal is sampled, all the frequencies above Nyquist frequency are clipped off in order to prevent improper sampling effects. Hence equation 8, in which data is scaled using number of points and doubled in amplitude, is used as the correct representation of the FFT result

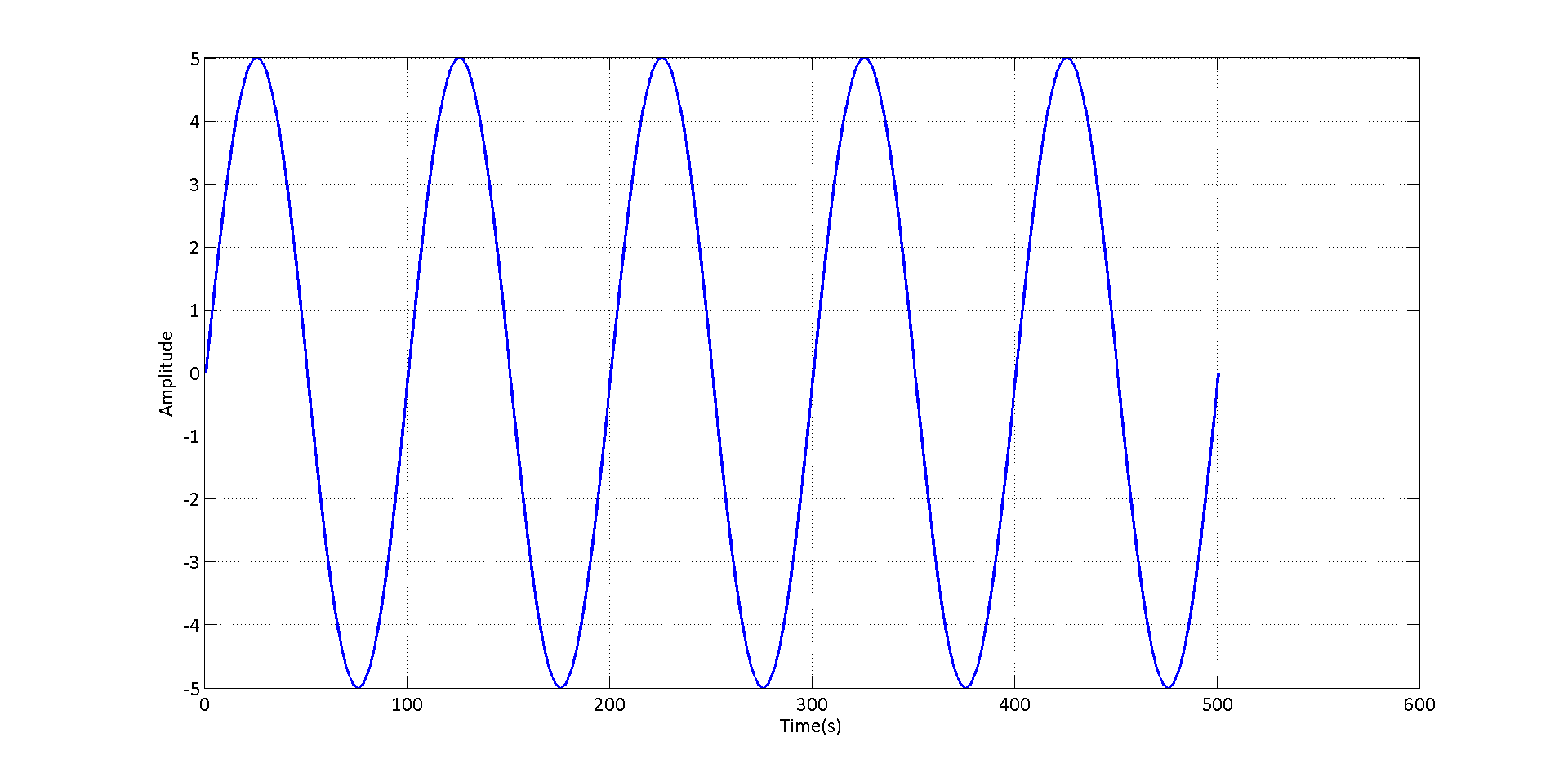
|  |  |  |
| --- | --- | --- |
|  |  |  |

For Fourier transform to be applied on data the following conditions are to be met, otherwise, the results obtained by an FFT will not be appropriate

* The data has to be periodic. , T being the period
* Discontinuities may occur in the data, but the data has to be periodic
* Function should possess a finite number of maxima and minima in a time period T
* Function should be absolutely integrable over any period T

|  |  |  |
| --- | --- | --- |
|  |  |  |

When a measurement is made, it is not possible to meet these conditions on the data acquired as in most cases when the measurement is being made, the signal frequency is not known. This is explained by figure 3



Moreover, the signal can also be transient in nature in which case the signal is never periodic iany windo

Window B

Window A

**Figure 4 Figure representing periodicity and non-periodicity**

In figure 4, window A and window B represent the start and end point of when a measurement was made. In case of window A, it is evident that 1 period of the data is captured. Hence, the signal is periodic and an FFT would result in reasonable frequencies. However, in case of window B, data is captured only for 0.75 period. This means that the reconstructed signal will not be a representation of the original wave. An FFT on this piece of data would suffer from misrepresentation of actual frequency because of non-periodicity in the data. In most measurements, this case arises.

There also a possibility when the data is a transient (time-varying) signal. Periodicity assumption breaks down even in this case, irrespective of the window size as there is no constant time period T to start with.

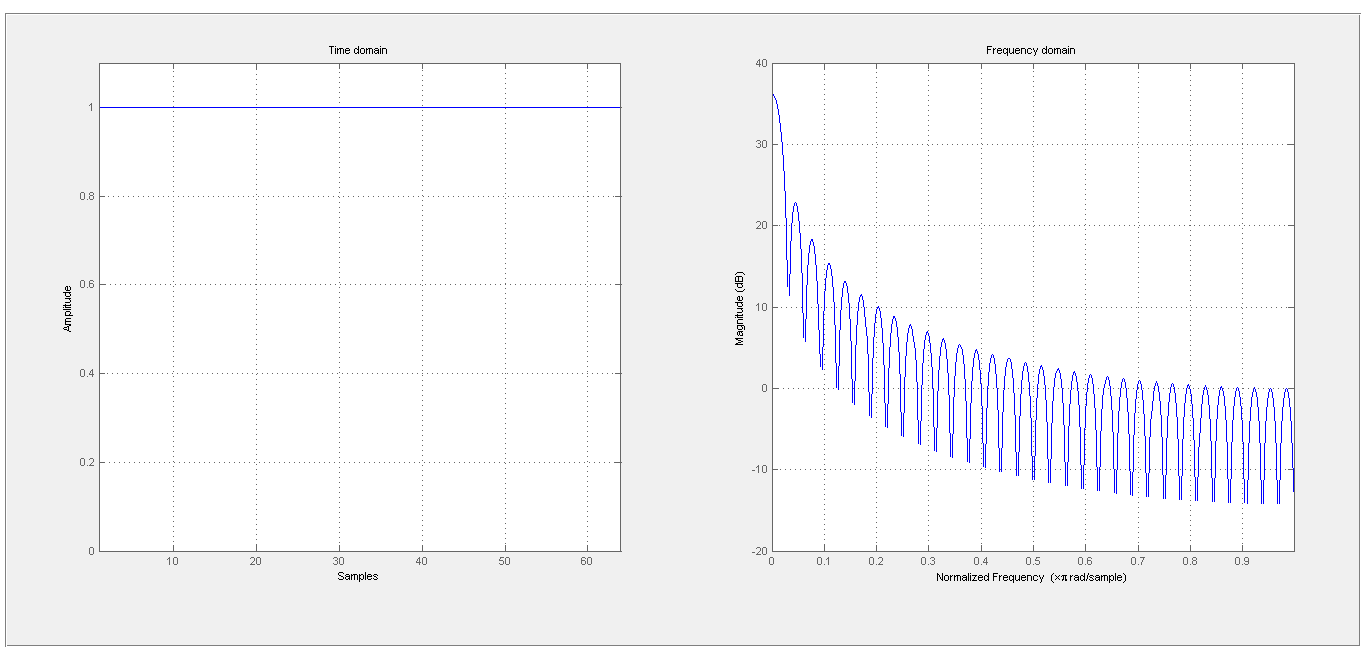
This leads to a phenomenon called leakage (see Fig.11). Leakage is the most common and serious error and cannot be eliminated. There are several ways to reduce leakage

* Averaging techniques
* increased frequency resolution
* use of periodic/special excitation
* use of window functions

In this experiment, window functions have been used to reduce leakage. There are a variety of windows that can be used to reduce leakage the most common ones being,

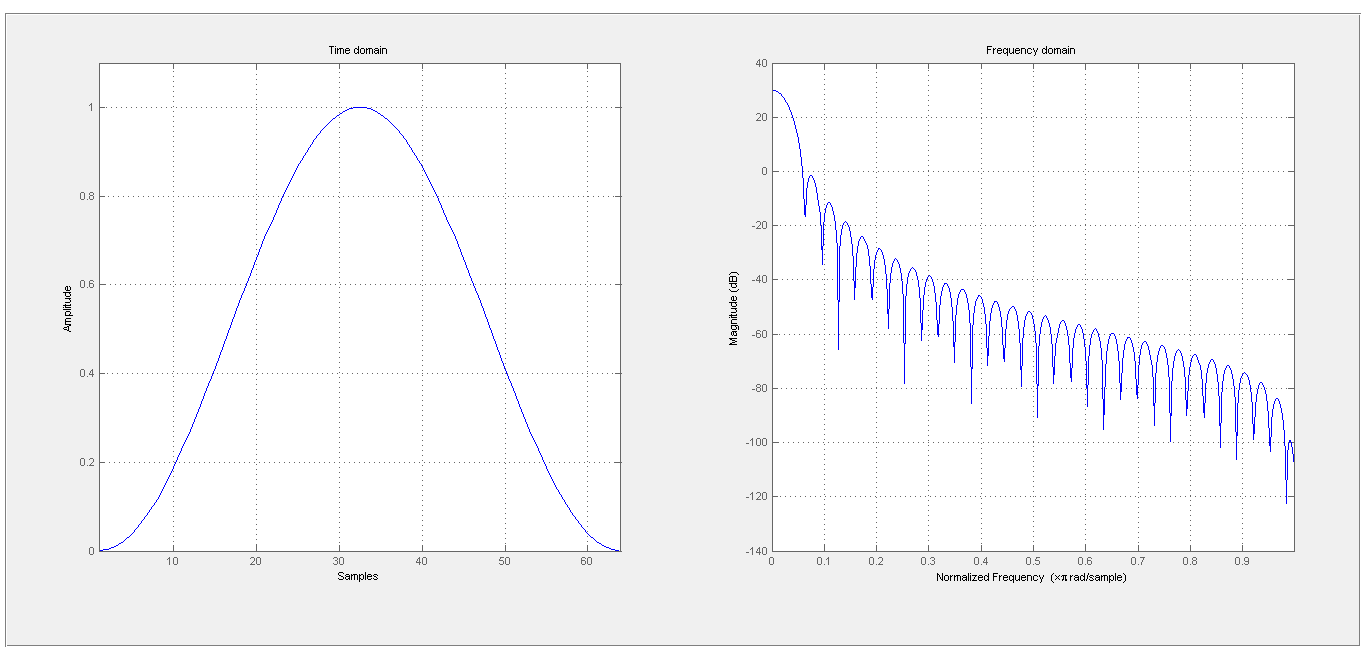
* Rectangular
* Hanning
* Flattop

Figure 5 describes time-domain and frequency domain of a typical rectangular window. When a rectangular window is applied, the data is multiplied by a signal of unit amplitude in time-domain. However, referring to the frequency domain plot, it can be noticed that center lobe loses 36% of energy in . This window is used for better frequency resolution.

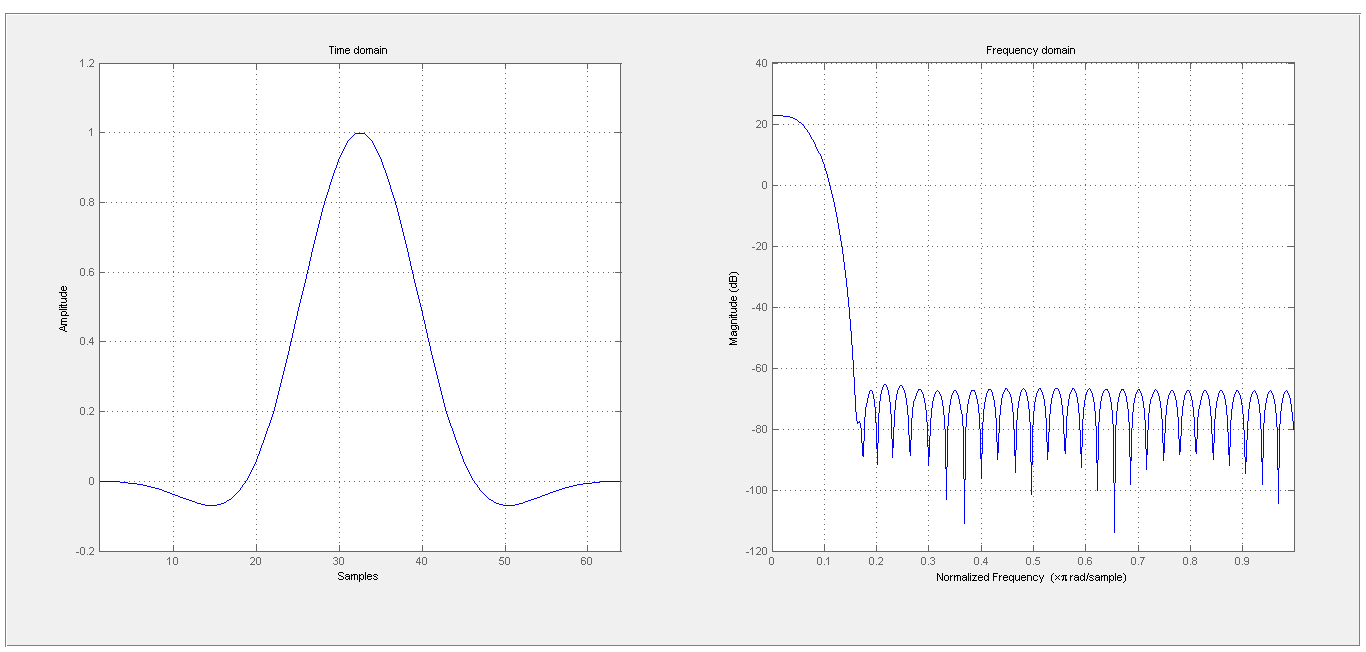
****

**Figure 5 Time domain and frequency domain representation of rectangular window**

Figure 6 describes a hanning window in time and frequency domains. When this window is used, signal is multiplied in time-domain by a signal equivalent to a signal in time domain in figure 6. Referring to the scaled frequency domain plot, it is evident that centre lobe loses 16% of energy in amplitude. This window gives more weightage to the centre of the window when compared to rectangular window but lesser weightage compared to a flattop window. This window is used to when get moderate frequency and amplitude resolution.



**Figure 6 Time domain and frequency domain representation of Hanning window**



**Figure 7 Time domain and frequency domain representation of flattop window**

Figure 7 describes a flat-top window in time and frequency domains. When this window is used, signal is multiplied in time-domain by a signal equivalent to a signal in time domain in figure 7. Referring to the scaled frequency domain plot, it is evident that centre lobe loses only 0.1% of energy in amplitude. This window gives most weightage to the center of the window when compared to other windows. This window is used to get good amplitude resolution at the cost of frequency resolution. It is not advisable to use this window to obtain frequencies that are closely spaced.

|  |  |  |
| --- | --- | --- |
| Window | % of energy lost in | Advantages |
| Rectangular | 36% | Good frequency resolution but worst amplitude resolution |
| Hanning | 16% |  |
| Flattop | 0.1% |  |

When windows are used to process the data, it is apparent from the time domain of each of the windows as shown in figures 6 and 7 that the amplitude of the signal at the center of the domain is given maximum weightage. This makes the actual signal to lose some energy or amplitude which makes it imperative to use a correction factor when windows are used. There are two factors that can be used to process the signal – amplitude correction factor (ACF) and energy correction factor (ECF) given by equations 10 and 11

|  |  |  |
| --- | --- | --- |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  |  |  |

Amplitude correction factors are used for narrow band data whereas energy correction factors are used in the case of broadband data.

The objective of this experiment is to

* Generate a signal of a given frequency using the apparatus, discussed in the next section with a sampling rate allowed by the NI-cDAQ system
* Evaluate using sampling rate – Fs and block size N and to be able generate signal with maximum leakage and acquire data for this signal
* Perform an FFT on this signal and in order to detect leakage
* Use different windows and effectively be able to estimate actual amplitudes on the signal after the windows and correction factors are applied
* Simulate the same signal using MATLABs virtual ADC and compare the two signals
* Calculate sampling parameters to obtain a 179Hz, 1000mV(rms) signal, without leakage
* Implement block averaging on a 50Hz square wave

APPARATUS

The following apparatus has been used for the experiment

* Signal generator – Sony Tektronix AFG310
* Data Acquisition System – National Instruments – NI-cDAQ-9172
* Microphone-PCB Piezotronics ICP SN 23957
* Amplifier – RCA SA155 Integrated stereo
* Speaker – Optimus XTS-40

The schematic representation of apparatus is presented in figure 8

Signal Generator

Amplifier

Speaker

NI-cDAQ

Microphone

**Figure 8 Schematic representation of the apparatus**

Signal generator is used to obtain a signal of particular frequency and amplitude. The output of a signal generator is connected to the input of an amplifier. Amplifier knob is used to adjust the intensity, and hence amplitude, of the signal. Signal generated with a desired frequency and amplitude is captured through the microphone and is analyzed using NI-cDAQ. NI-cDAQ has a 24-bit ADC with adjustable sample rate according to equation 10

|  |  |  |
| --- | --- | --- |
|  |  |  |

Where n is an integer with range 1-31 and fM is the internal master time-base with fM=13.1072MHz.

Although FFTs can be performed within the Sound and Vibration assistant application, data acquired though NI-cDAQ is imported into MATLAB for comparison with simulated values.

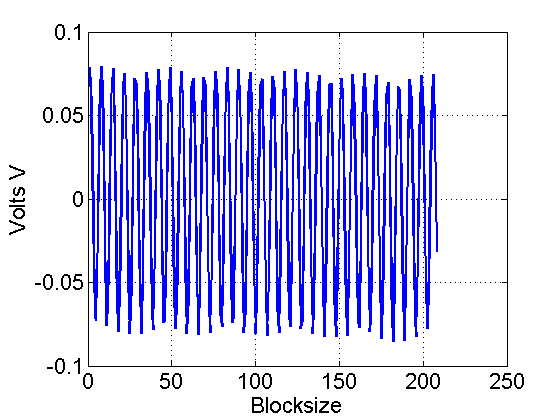
EXPERIMENTAL PROCEDURE

As already mentioned, equation 10 is initially used to calculate the frequency at which the whole experiment is conducted. For obtaining sampling frequency, M-number is utilized and later rounded off to closest frequency. The details of this calculation are presented in Appendix A. The sampling frequency is evaluated as **10240Hz.** Also, at this sampling rate Nyquist frequency (Fn) is **5120Hz.** This means that only frequencies up to this value appear in the spectrum and all the values beyond Fn are clipped off. Also, it has been suggested to choose a frequency closer to the center of the spectrum. With these suggestions in mind, a frequency of 1500Hz is chosen for generating the signal. Amplitude is not chosen at this point. At this point, a signal is generated in the signal generator which is connected to the amplifier which is in turn connected to the stereo. The knob of the amplifier is adjusted in such a way that signal generated is above noise floor but well beneath maximum range of micro-phone which is about 1V. The signal captured by microphone is transferred to NI-cDAQ data acquisition system. Sampling rate is set to appropriate value and the sensitivity is adjusted to 1000mV/Pa so that the amplitudes can be directly read from the data-view screen of the acquisition system without using a conversion factor. The data collected is then analyzed using power spectrum option of Sound and Vibration Assistant just to check if frequency is closer to that of actual signal. After the check is made, data is exported to MATLAB for further analysis.

For further analyses, another square wave at 50**Hz** is generated with the same sampling frequency, for implementing block-averaging techniques. Block-size desired is 1024 with a deltat of 1/10240s and a block average is computed for 50 blocks. Hence the signal has to be sample for at least . This data is later exported to MATLAB to implement block averaging.

MEASUREMENT AND DATA SUMMARY

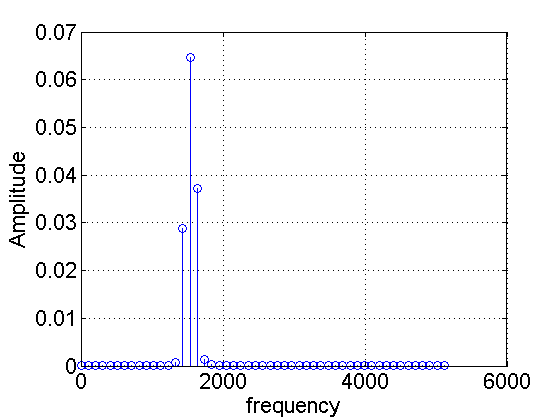
Sinusoidal signal generated at 1500Hz and recorded amplitude of 0.08V is presented in figure 9. In figure 10, spectrum of the same data is presented as stem plot.



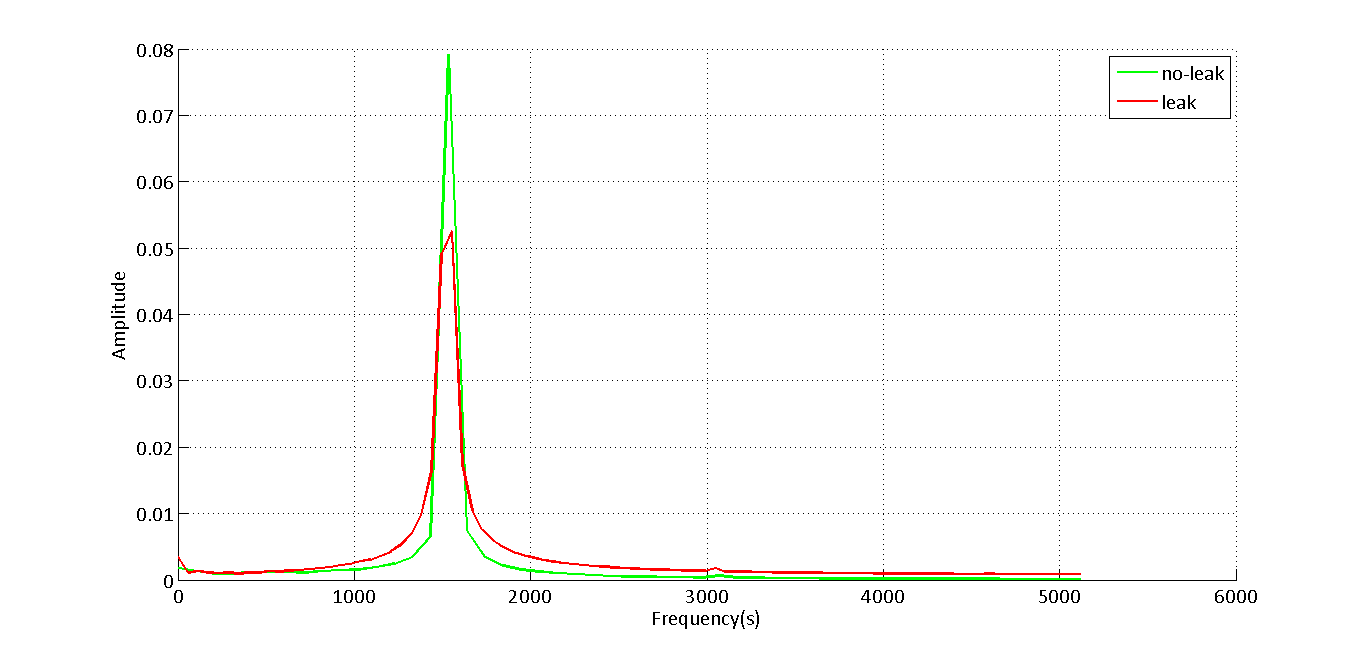
**Figure 9 Window of raw data on which spectral analysis is performed**

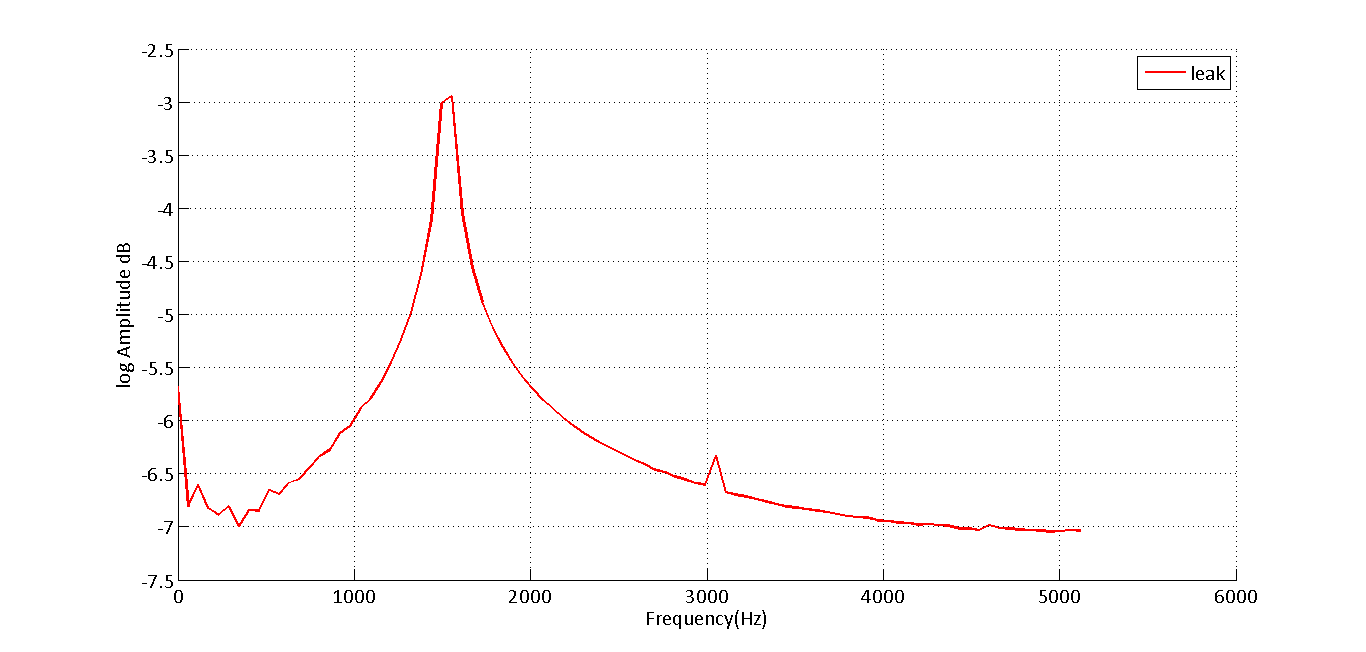
It is evident that amplitude of signal is about 0.08V. However, on performing a spectral analysis on the data, the frequency is clearly close to 1500Hz but this frequency lost its energy to neighboring frequencies. This is due to the leakage which is explained in the background section.

With the same sampling parameters and knowing the fact that maximum leakage occurs at , the block-size of the signal is adjusted to generate a signal with maximum amplitude. Calculations for obtaining block-size for max leakage are presented in Appendix B. This spectrum overlaid on the original spectrum is presented in figure 11.

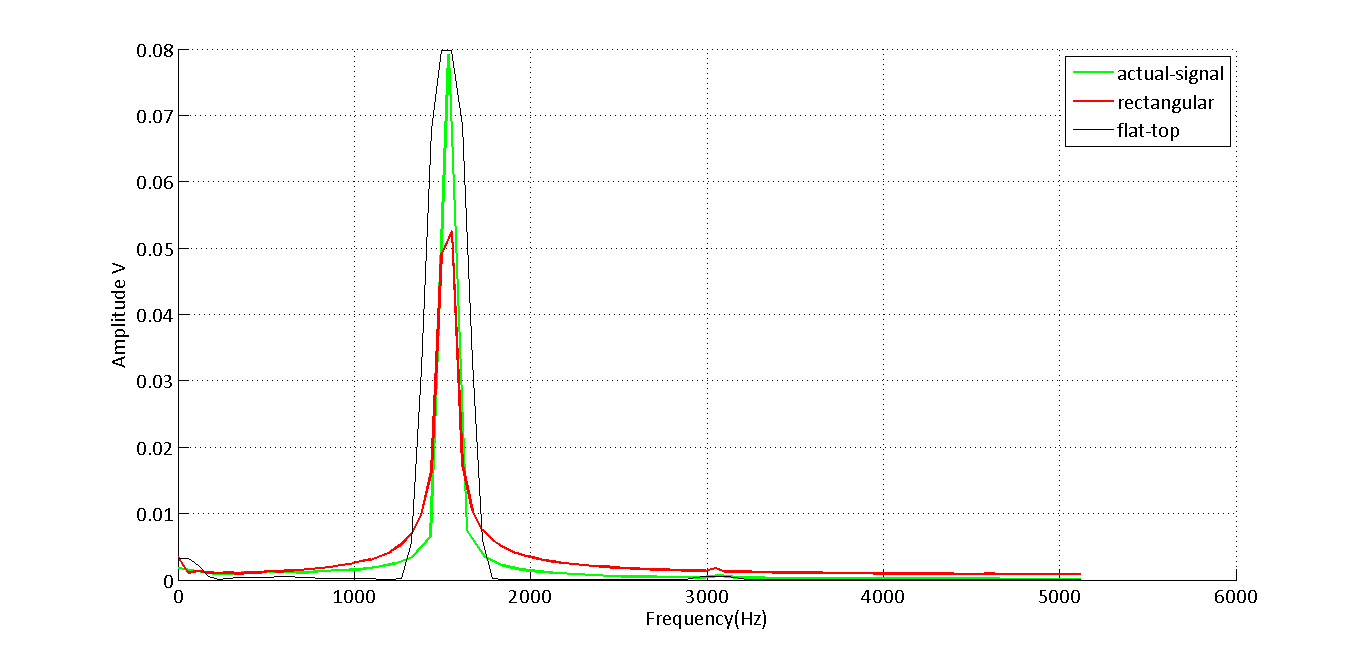


**Figure 10 Spectral analysis on actual data**



**Figure 11 Spectral analysis of the signal with maximum leakage (red). Log amplitude of the same signal reveals the extent of leakage in the signal**

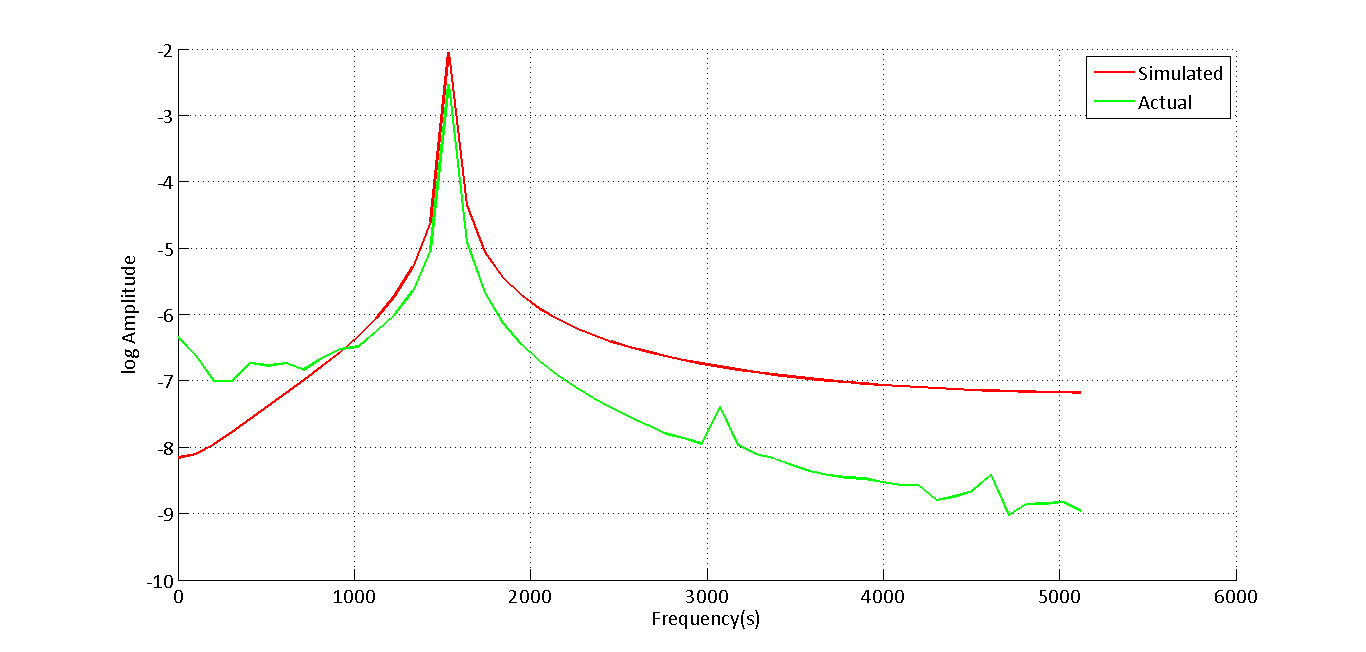
Rectangular window or no-window provides maximum frequency resolution whereas a flat-top window generates a signal with maximum amplitude resolution. The signal with maximum leakage is processed with these windows to get the spectra presented by figure 11



**Figure 11 Amplitude and frequency variation when different windows are used**

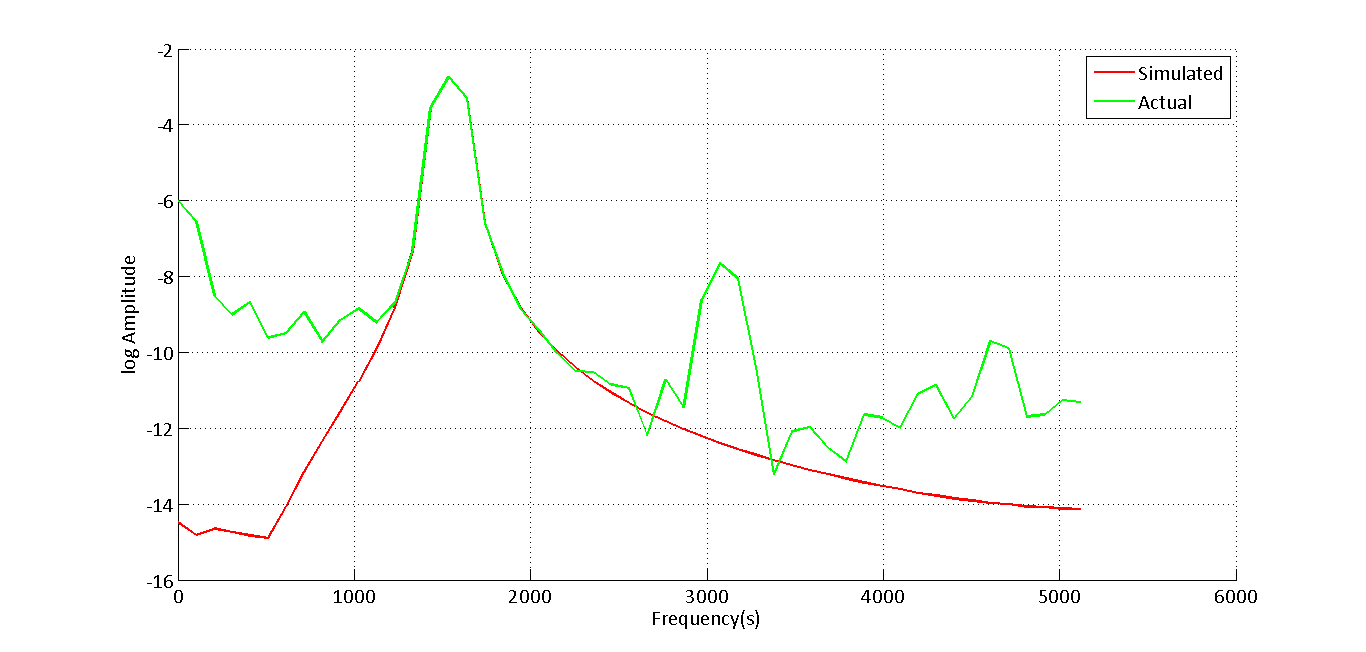
Flat top window clearly denotes a peak closer to the actual amplitude. However, because of flatness of the peak, if two frequencies are adjacent to each other, within the range of the peak, they get merged, hence the frequency resolution is not obtained using a flat-top window. However, in the case a rectangular window, a single peak stands out, at one frequency but it does not denote the correct amplitude of this frequency. Hence rectangular window cannot be used for obtaining amplitude from a signal.

MATLAB is used to generate an equivalent signal with amplitude of 0.08V and frequency of 1500Hz. The signal is then quantized using a 24-bit virtual ADC. An FFT is performed on this signal to check accuracy of the simulated signal with the actual signal. The overlaid result is presented in figure 12.

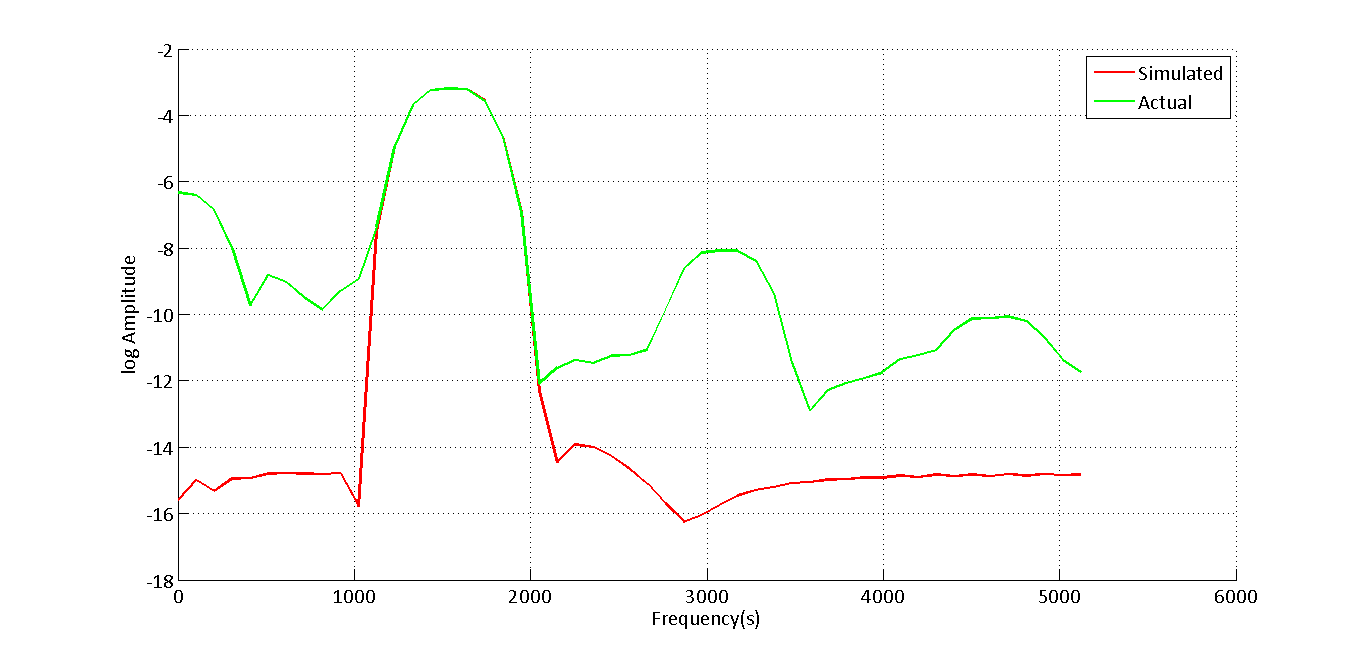


**Figure 12 Matlab Spectrum overlaid on the actual spectrum (with leakage)**

From figure 12 it can be concluded that the spectrum generated in MATLAB by simulating the signal, is close to the actual signal and the logic used to generate this spectrum can be used to simulate actual conditions. It can also be noticed that the actual signal has local peaks at 3000 Hz and 4500 Hz. These peaks are due to the harmonic distortion and are generated because of the inefficiency of the system to generate a perfect sine-wave. Also, it can be noticed that these peaks are not generated in the simulated spectrum. In order to compare how the spectrum behaves when different windows are used on the simulated data and to compare how this is different from actual spectrum, hanning and flat-top windows are also implemented and are presented in figures 13 and 14. The peaks can be used to distinguish between the windows used on the data. It is also evident from these figures that hanning window which is a trade-off between amplitude correction and frequency correction has both flat and pointy characteristics. MATLAB code used for comparing actual data with virtual ADC is in Appendix D

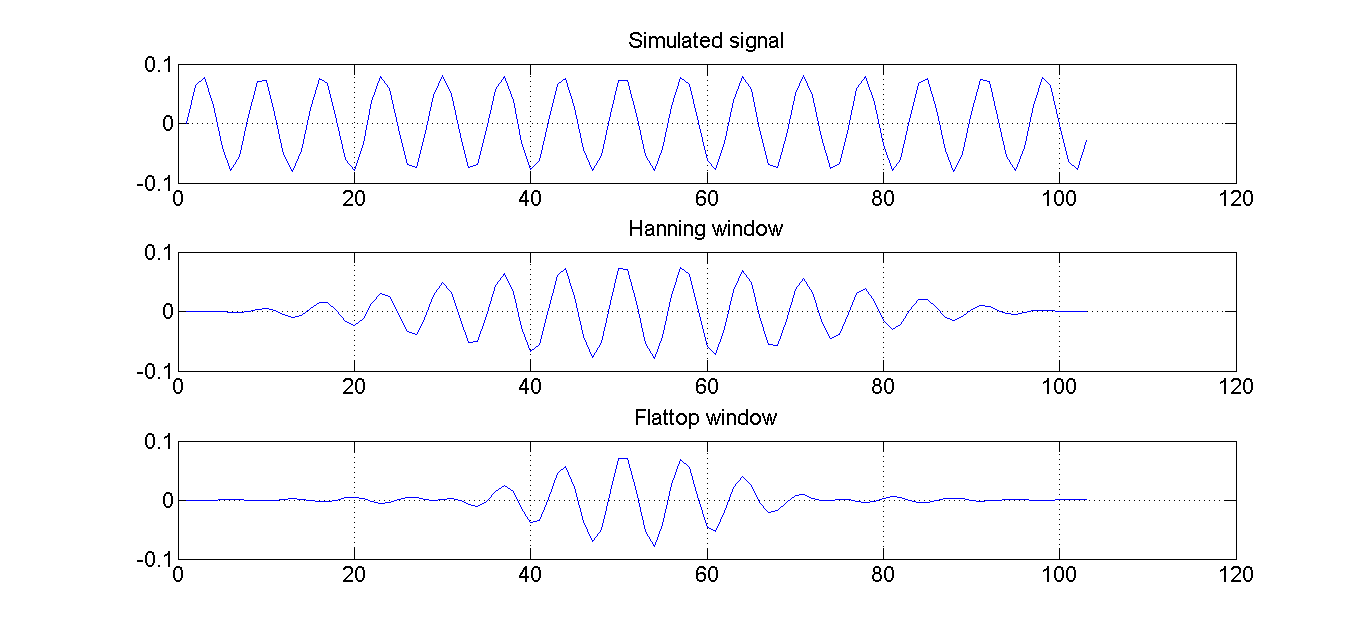


**Figure 13 Hanning window applied on simulated and actual data**



**Figure 14 Flat-top window applied on simulated and actual data**

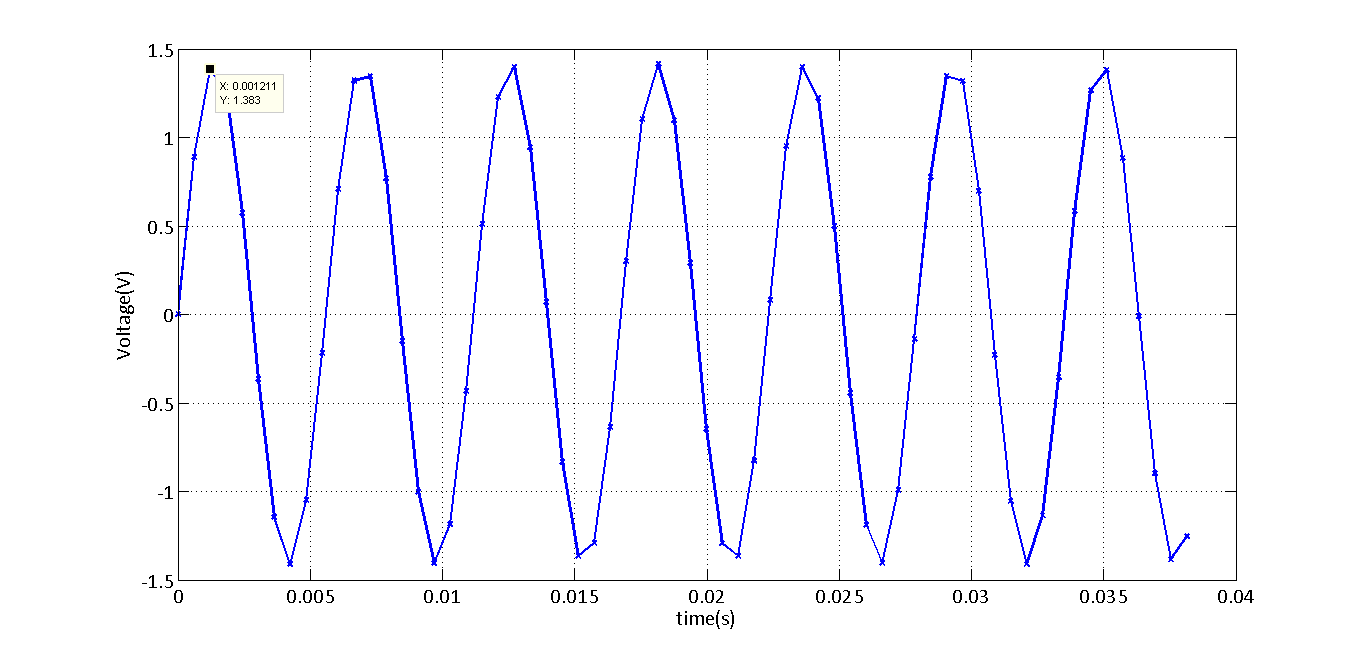
Window is applied to a signal in time-domain before an FFT is applied over the resultant data. In order to visualize the effects of windows on the time-domain data set of plots have been presented in figure 15 which denote the time domain transformation of windows on actual data



**Figure 15 Transformation of time-signal when a particular window is applied to the data**

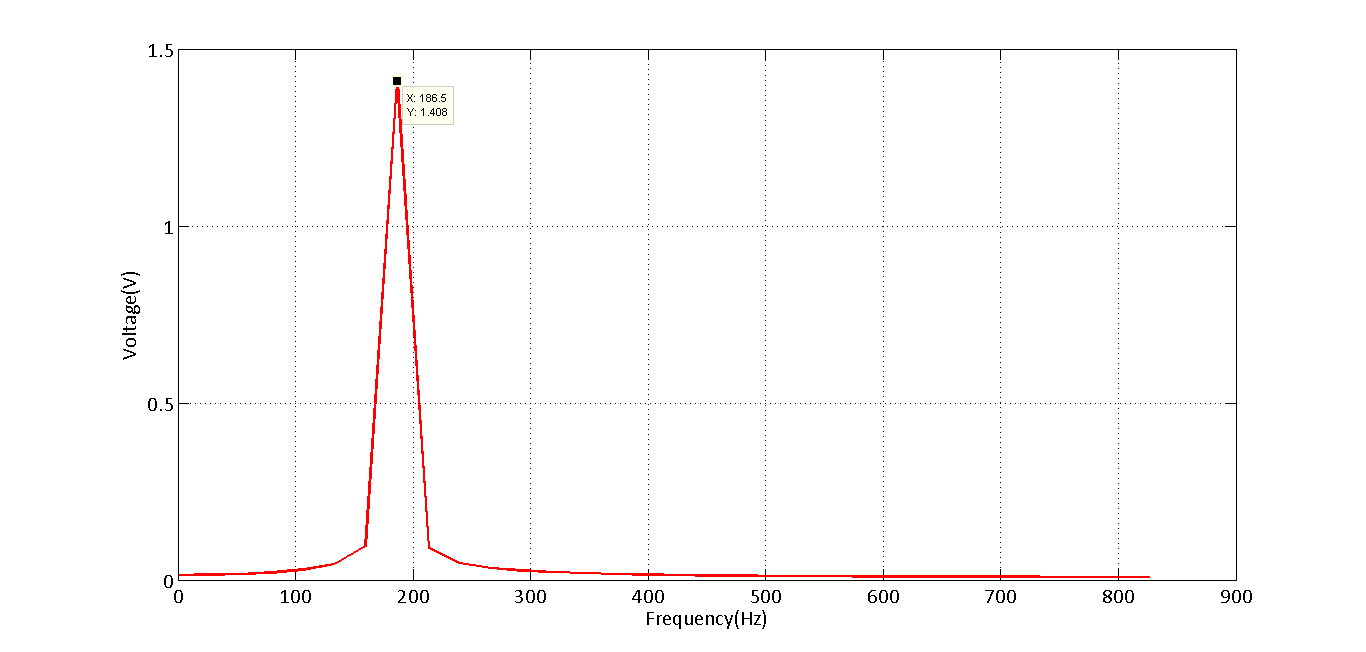
**Determination of sampling parameters for processing a 179Hz signal**

As it has been concluded in the previous section that MATLAB simulation is an accurate way of simulating real-time conditions, a signal of frequency 179Hz is to be processed virtually in MATLAB. The amplitude of the signal is 1000mV rms. This means that the peak of the signal occurs at 1414mV or 1.414V.Further, according to Shannon’s sampling rules the sampling frequency must be greater than twice that of the signal frequency. Hence sampling frequency must be greater that 358Hz. However, the frequency of the ADC is calculated from equation 10. The minimum possible frequency obtained at n=31 for the ADC is 1652Hz. As this value is well above the 358Hz, this can be used as a starting point to evaluate the signal. The other parameter to choose is the block-size N. The block sizes which ADC allows are of the form 2n. Hence, the signal is evaluated with N=64, Fs=1652Hz on a 24-bit virtual ADC with a range of +/-5V. The time-history of the signal for one block is presented in figure 16



**Figure 16 179Hz signal processed at Fs=179Hz for N=64**

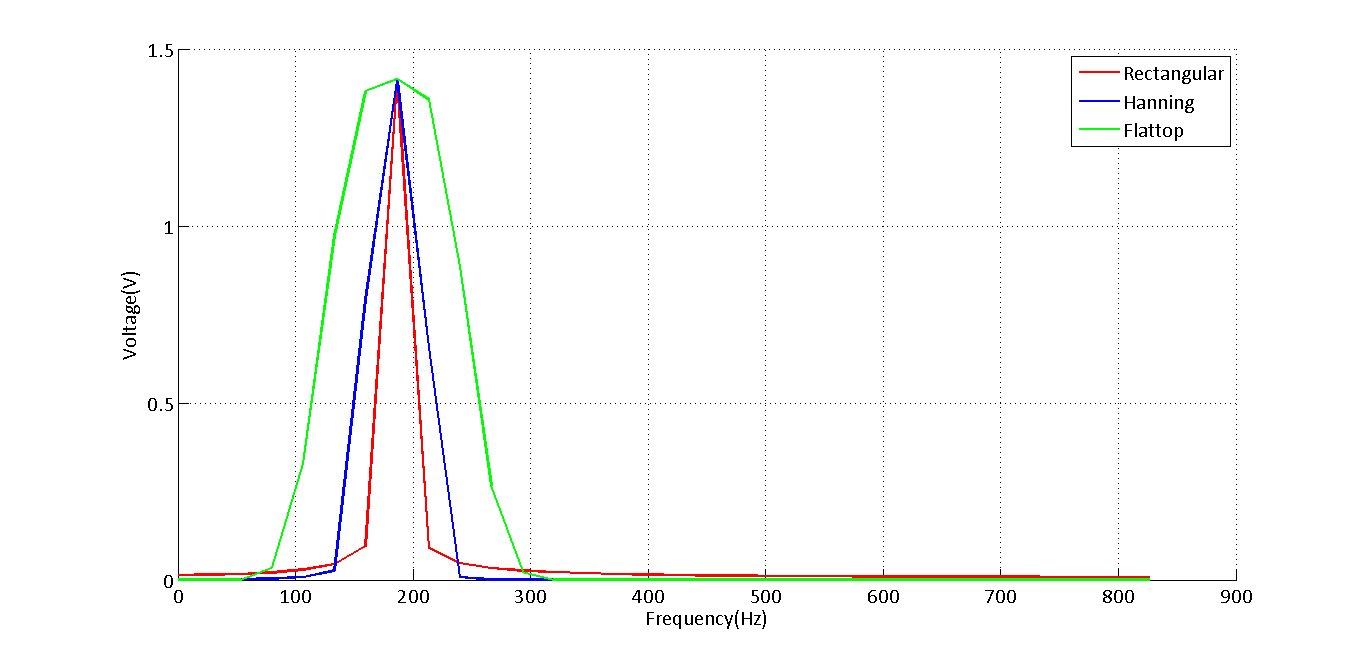
From figure 16, it is also evident that amplitudes are in the range of 1.4mV. Hence this is a correct representation of the signal in time-domain. On performing FFT on this signal figure 17 is obtained.



**Figure 17 Frequency domain of the signal processed at 179Hz**

It can be noticed in the signal that the frequency domain is processed until 826Hz. which is the half of the sampling frequency or the Nyquist frequency. Peak of this data occurs at 186Hz and 1.408V. As the measurement of interest in this case is amplitude, it can be concluded that the sampling parameters used to process the signal are appropriate with an error of 0.57%.

It is known that the original signal is implemented at 1.4V and the sampled signal is processed at 1.4V. This means that there is no leakage in the data. Also, if any of the windows are implemented on actual data, it would not make a difference in the amplitude. This is evident from figure 18 which denotes an implementation of windows on the virtually sampled data.



**Figure 18 Implementation of different windows on sampled data**

Hence it can be concluded that, implementation of windows is not necessary for data that does not undergo leakage. MATLAB code attached in Appendix D.

**Block Averaging**

Consider a block size of length N and let T be the total length of the time history. Let this time history contain M number of blocks. If spectral analysis is performed on the data, it results in N number of points at which the frequency is calculated. This means that on performing FFT on a block of size N, we generate a vector of N – frequencies. As the time history T has M number of blocks of the same size, it can be concluded that there would be M spectral lines for each N.

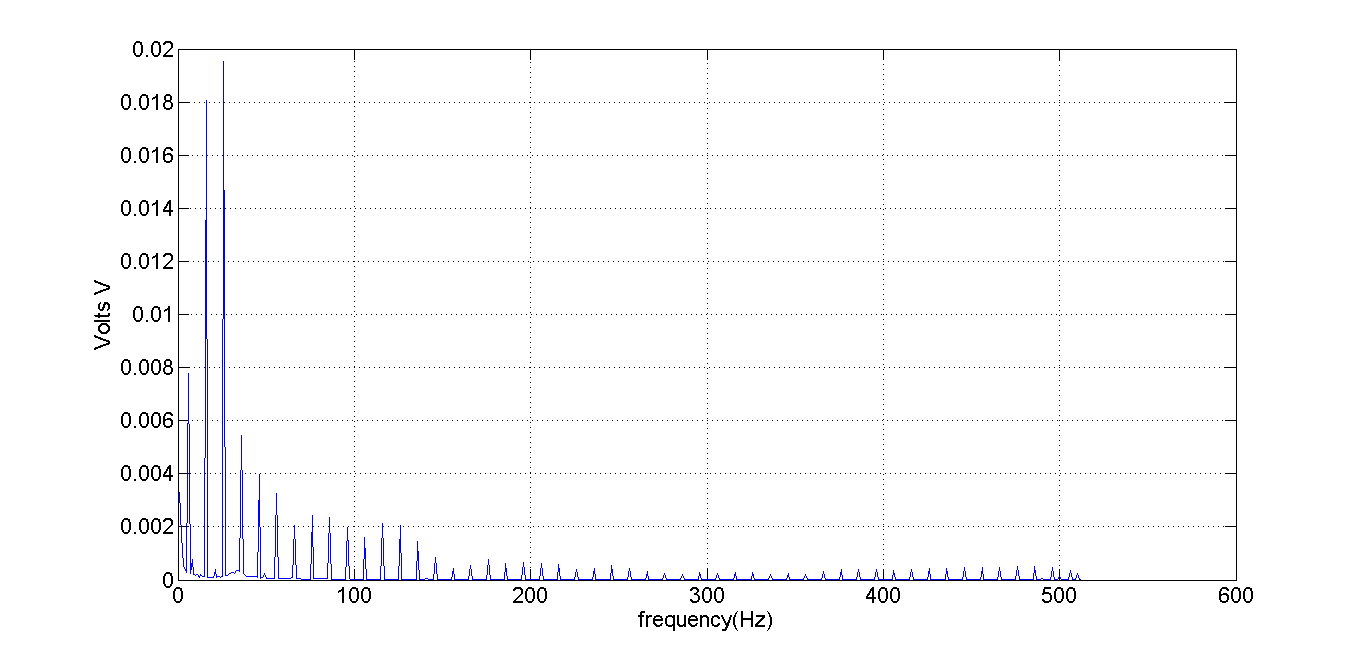
Let the frequencies of each block denoted by Ni={ni1, ni2, ni3 ….nin} where i = 1 to M

Then the block average can be calculated by using the following equation

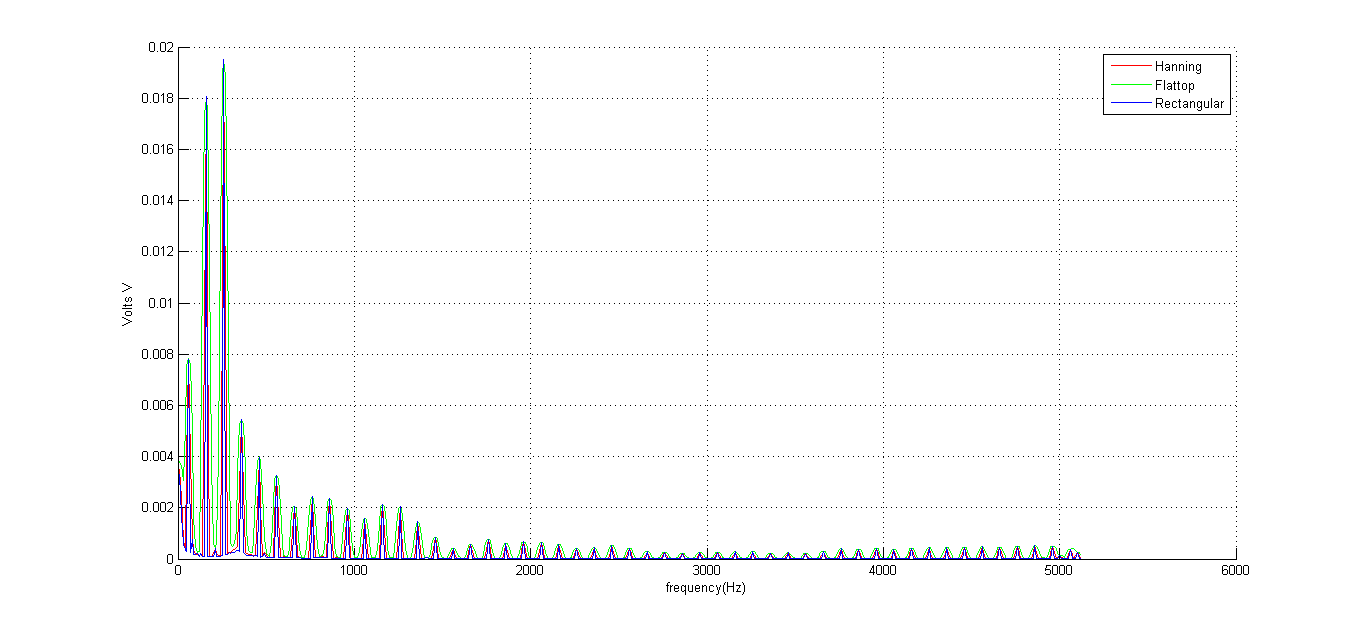
|  |  |  |
| --- | --- | --- |
|  |  |  |

The MATLAB code implementing the above equation is presented in Appendix C

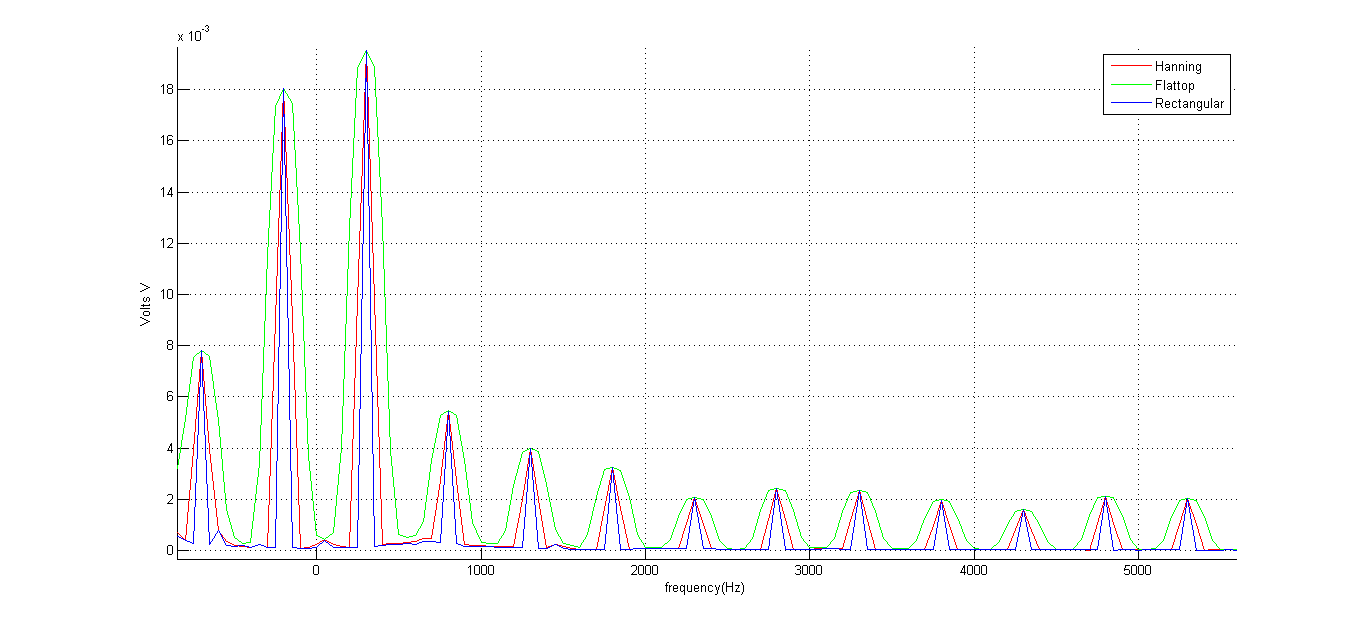
In order to test the implementation of the code above, a square of 50Hz is generated using the signal generator and the data is imported to MATLAB. Block-size N is adjusted to 1024 and an arbitrary time-history is captured. The sampling rate of the ADC is set to 10240Hz for this analysis. Hence in order to obtain 50 blocks, the data has to be sampled for atleast 50\*1024\*(1/10240) = 5s. Implementation of block –averaging on the data collected results in the following spectrum



**Figure 19 Implementation of block averaging on a square wave**

Figures 20 and 21 present the results of applying windows on a block-averaged spectrum. It is evident that the resulting spectra of each of the windows are closer to the actual spectrum, represented by rectangular window in this case.

**Figure 20 Implementation of block-averaging and windows on the square wave**



**Figure 21 Zoomed-in view of implementation of block-averaging and window-effects**

SUMMARY AND CONCLUSIONS

* A 1500 Hz signal is generated at a sampling rate of 10240Hz using the signal generator and the data is acquired by the NI-cDAQ system
* Signal with 2 is generated by changing the block-size N and it is noticed that the signal exhibits maximum leakage for this block-size
* FFT is performed on the signal to notice leakage of energy into adjacent spectral bins. It has been noticed that log amplitude plot makes leakage apparent
* Rectangular, hanning and flat-top windows have been implemented in MATLAB and it has been noticed that rectangular window gives maximum frequency resolution and flat-top window gives maximum amplitude resolution. Hanning window posses characteristics of both rectangular and flat-top windows
* MATLAB is used to successfully simulate the actual signal by using a 24-bit virtual ADC. Signals have been compared and are found to be matching
* A179hz, 1000mV(rms) signal is processed using MATLAB’s virtual ADC and sampling parameters have been calculated so that the signal is obtained without leakage. It has also been found that using windows on the data which does not have leakage does not change the energies/amplitudes that the processed signal possess
* MATLAB code is generated to implement a block-averaging technique. A 50Hz square-wave is processed using this code and the results of the spectra have also been presented. Different windows are implemented on this data to observe the variations in spectrum. It has been observed that the spectrum does not change significantly when windows are implemented

REFERENCES

1. S.W.Smith, Digital Signal Processing for scientists and engineers, p. 43, Newness, 2003

Appendix A

Calculation of sampling frequency using M-number

M-Number – 94245207

5th digit is “5”

From Equation 10,

Since n is an integral value, sampling frequency is fixed at 10240Hz for the experiment

Appendix B

Calculation of block-size for maximum leakage

Fs – Sampling rate

N – Blocksize

- time step = 1/Fs

- frequency resolution

T – Time period

Frequency resolution is a function of Fs and N. Maximum leakage occurs at . Hence, either N or Fs can be altered to half the value of frequency resolution. In this experiment, block size is doubled to get a signal of maximum leakage.

Appendix C

MATLAB code for implementation of block averaging

clc;

clear all;

close all;

%Sampling parameters

Fs = 10240;

delta\_t = 1/Fs;

data = csvread('C:\Accost\Mich Tech\Course work\Dynamic Systems and measurements\Lab Assignments\Assignment-3\square\_wave.csv');

time\_history = data(:,2);

N = 1024; %Block-size

N\_total = length(time\_history); %Total number of points

x = time\_history;

T\_total = N\_total\*delta\_t; %Total time of measurement

X=x;

count = 1;

for i=1:N:N\_total-1024

i;

block = i+1023;

u = uencode(X(i:i+1023),24);

d = udecode(u,24);

x = d;

w2 = hann(length(x));

w1 = flattopwin(length(x));

x1 = w2.\*x;

x2 = w1.\*x;

ACF\_1 = 1/mean(w2);

ACF\_2 = 1/mean(w1)

FFT\_1 = ACF\_1\*fft(x1,length(x1))/length(x1);

FFT\_2 = ACF\_2\*fft(x2,length(x2))/length(x2);

FFT = fft(x,length(x))/length(x);

block\_spectrum\_1(count,:)=(2\*abs(FFT\_1(1:length(FFT\_1)/2))); %Store each spectrum as a row

block\_spectrum\_2(count,:)=(2\*abs(FFT\_2(1:length(FFT\_2)/2)));

block\_spectrum\_3(count,:)=(2\*abs(FFT(1:length(FFT)/2)));

count = count+1;

end

freq = Fs/2\*(linspace(0,1,N/2));

%Calculating block averages

block\_average\_1 = mean(block\_spectrum\_1(1:50,:),1); %Consider first 50-blocks only

block\_average\_2 = mean(block\_spectrum\_2(1:50,:),1);

block\_average\_3 = mean(block\_spectrum\_3(1:50,:),1);

%Plots

hold on

p1 = plot(block\_average\_1,'-r')

p2 = plot(block\_average\_2,'-g')

p3 = plot(block\_average\_3,'-b')

xlabel('frequency(Hz)')

ylabel('Volts V')

legend('Hanning','Flattop','Rectangular')

grid

Appendix D

Matlab code for comparing actual signal with virtual ADC

%MATLAB-SIMULATION

clc;

clear all;

close all;

Fs = 10240;

delta\_t = 1/Fs;

T = 0.01;

N = ceil(T/delta\_t);

delta\_f = Fs/N;

% N = ceil(Fs\*2/delta\_f);

f1 = 1500;

A = 0.08;

t = (0:(N-1))\*delta\_t;

X = A\*sin(2\*pi\*f1\*t);

u = uencode(X,24);

d = udecode(u,24);

x = d;

w1 = hann(length(x))';

w2 = flattopwin(length(x))';

w3 = rectwin(length(x))';

x = w2.\*x;

ACF = 1/mean(w2);

ECF = 1/rms(w2);

figure

% plot(x,'-x')

% NFFT = 2^nextpow2(N);

NFFT = N;

FFT = ECF\*fft(x,NFFT)/NFFT;

FFT\_f = Fs/2\*linspace(0,1,NFFT/2);

hold on

p1 = plot(FFT\_f,log(2\*abs(FFT(1:length(FFT)/2))),'-r')

%LAB-DATA

dataat100 = csvread('C:\Accost\Mich Tech\Course work\Dynamic Systems and measurements\Lab Assignments\Assignment-3\lab\_3\_data\_at\_1.csv');

x\_actual = dataat100(1:103,2);

N\_actual = 103;

hann\_actual = hann(N\_actual);

rect\_actual = rectwin(N\_actual);

flat\_actual = flattopwin(N\_actual);

x\_actual = x\_actual.\*flat\_actual;

ACF\_actual = 1/mean(flat\_actual);

ECF\_actual = 1/rms(flat\_actual);

FFT\_actual = ECF\_actual\*fft(x\_actual,N\_actual)/N\_actual;

FFT\_f\_actual = 10240/2\*linspace(0,1,N\_actual/2);

FFT\_a\_actual = 2\*abs(FFT\_actual(1:length(FFT\_actual)/2));

p2 = plot(FFT\_f\_actual,log(FFT\_a\_actual),'-g')

set(p1,'linewidth',2);

set(p2,'linewidth',2);

set(gca, 'fontname', 'Calibri', 'fontsize', 16);

xlabel('Frequency(s)');

ylabel('log Amplitude');

legend('Simulated','Actual')

Appendix D

Matlab code used for using different windows on virtually generated 179Hz sine-wave

clc;

clear all;

close all;

%Sampling parameters

f = 179;

n = 31;

Fm = 13.1072\*1e6;

Fs = ceil((Fm/256)/n);

delta\_t = 1/Fs;

N = 64;

A = 1\*sqrt(2);

T = N\*delta\_t;

t = (0:N-1)\*delta\_t;

x = A\*sin(2\*pi\*f\*t);

u = uencode(x,24,5);

x = udecode(u,24,5);

%Windows

w1 = hann(length(x))';

w2 = flattopwin(length(x))';

w3 = rectwin(length(x))';

ACF\_1 = 1/mean(w1);

ACF\_2 = 1/mean(w2);

ECF = 1/rms(w3);

x1 = x.\*w1;

x2 = x.\*w2;

figure

% p1=plot(t,x,'-x'); grid on

FFT = fft(x,N)/N;

FFT\_1 = ACF\_1\*fft(x1,N)/N;

FFT\_2 = ACF\_2\*fft(x2,N)/N;

FFT\_f = Fs/2\*linspace(0,1,N/2);

% figure

hold on

p1 = plot(FFT\_f,(2\*abs(FFT(1:length(FFT)/2))),'-r');

p2 = plot(FFT\_f,(2\*abs(FFT\_1(1:length(FFT)/2))),'-b');

p3 = plot(FFT\_f,(2\*abs(FFT\_2(1:length(FFT)/2))),'-g');

grid on

set(p1,'linewidth',2);

set(p2,'linewidth',2);

set(p3,'linewidth',2);

set(gca, 'fontname', 'Calibri', 'fontsize', 16);

xlabel('Frequency(Hz)');

ylabel('Voltage(V)');

legend('Rectangular','Hanning','Flattop')

End of the document