Turk J Elec Engin, VOL.14, NO.1 2006, © TÜBİTAK

A Labview-Based Virtual Instrument for Engineering Education: A Numerical Fourier Transform Tool

Levent SEVGİ, Çağatay ULUIŞIK
Doğuş University, Electronics and Communication Engineering Department,
Zeamet Sok. No. 21, Acıbadem / Kadıköy, 34722 İstanbul-TURKEY
e-mail: lsevgi@dogus.edu.tr

Abstract

Engineering is based on practice. The minima of this practice should be given during the university education. This has become more and more comprehensive and expensive parallel to high-technology devices developed and presented to societies. Computers, microprocessor-based devices, programmable systems on Chip (PSoC), etc., make engineering education not only very complex but interdisciplinary as well. Building undergraduate labs has become more and more expensive if only physical experimentation and hands-on training are targeted. On the other hand, simple, comparatively much cheaper software may turn a regular personal computer (PC) into a virtual lab. The key question therefore is, to establish a balance between virtual and real labs, so as to optimize cost problems, while graduating sophisticated engineers with enough practice. This article introduces a virtual instrumentation (a LabVIEW package DOGUS_FFT.VI) for numerical Fourier transform calculations, which may also be used as an educational tool.

Key Words: *Fourier transformation, FFT, DFT, LabView, simulation, engineering education, visualization, virtual instrumentation.*

1. Introduction

The analysis of real world signals is a fundamental problem for many engineers and scientists, especially for electrical engineers since almost every real world signal is changed into electrical signals by means of transducers, e.g., accelerometers in mechanical engineering, EEG electrodes and blood pressure probes in biomedical engineering, seismic transducers in Earth Sciences, antennas in electromagnetics, and microphones in communication engineering, etc.

Traditional way of observing and analyzing signals is to view them in time domain. Baron Jean Baptiste Fourier [1], more than a century ago, showed that any waveform that exists in the real world can be represented (i.e., generated) by adding up sine waves. Since then, we have been able to build (break down) our real world time signal in terms of (into) these sine waves. It is shown that the combination of sine waves is unique; any real world signal can be represented by only one combination of sine waves [2].

The Fourier transform (FT) has been widely used in circuit analysis and synthesis, from filter design to signal processing, image reconstruction, etc. The reader should keep in mind that the time domain -

frequency domain relations in electromagnetics are very similar to the relations between spatial and wave number domains. A simplest propagating (e.g., along z) plane wave is in the form of

$$\Phi(r,t) \propto e^{j(\omega t - kz)} \tag{1}$$

(where k and z are the wavenumber and position, respectively) and, whatever characteristics $\exp(j\omega t)$ have, are also applicable to $\exp(-jkz)$. Figure 1 is given for this illustration. It illustrates that:

- A rectangular time (frequency) window corresponds to a beam type (Sinc(.) function) variation in frequency (time) domain.
- Similarly, a rectangular aperture (array) in spatial domain corresponds to a beam type (Sinc(.) function) variation in wavenumber domain.
- The wider the antenna aperture the narrower the antenna beam; or, the narrower the pulse in time domain the wider the frequency band.

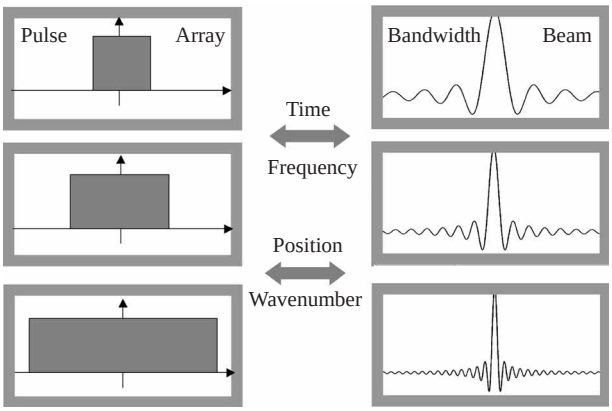

Figure 1. Time-frequency, and position - wavenumber relations

Therefore, FT has also been used in electromagnetics from antenna analysis to imaging and non-destructive measurements, even in propagation problems. For example, the split-step parabolic equation method (which is nothing but the beam propagation method in optics) [2-4] has been in use more than decades and is based on sequential FT operations between the spatial and wavenumber domains. Two and three dimensional propagation problems with non-flat realistic terrain profiles and inhomogeneous atmospheric variations above have been solved with this method successfully.

2. Fourier Transformation

The principle of a transform in engineering is to find a different representation of a signal under investigation. The FT is the most important transform widely used in electrical and computer (EC) engineering.

2.1. Fourier transform

The transformation from the time domain to the frequency domain (and back again) is based on the Fourier transform and its inverse, which are defined as

$$S(\omega) = \int_{-\infty}^{\infty} s(t)e^{-j2\pi f t}dt \tag{2a}$$

$$s(t) = \int_{-\infty}^{\infty} S(f)\, e^{j2\pi f t}df \tag{2b}$$

Here, *s(t), S(ω), and f* are the time signal, the frequency signal and the frequency, respectively, and $j = \sqrt{-1}$. We, the physicists and engineers, sometimes prefer to write the transform in terms of angular frequency $\omega = 2\pi f$, *as*

$$S(\omega) = \int_{-\infty}^{\infty} s(t)e^{-j\omega t}dt \tag{3a}$$

$$s(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} S(\omega)\, e^{j\omega t}d\omega \tag{3b}$$

which, however, destroys the symmetry. To restore the symmetry of the transforms, the convention

$$S(\omega) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} s(t)e^{-j\omega t}dt \tag{4a}$$

$$s(t) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} S(\omega)\, e^{j\omega t}d\omega \tag{4b}$$

is sometimes used. The FT is valid for real or complex signals, and, in general, is a complex function of ω (or f).

The FT is valid for both periodic and non-periodic time signals that satisfy certain minimum conditions. Almost all real world signals easily satisfy these requirements (It should be noted that the Fourier series is a special case of the FT). Mathematically,

- FT is defined for continuous time signals.
- In order to do frequency analysis, the time signal must be observed infinitely.

Under these conditions, the FT defined above yields frequency behavior of a time signal at every frequency, with zero frequency resolution. Some functions and their FT are listed in Table 1.

Table 1. Some functions and their Fourier transforms

Time domain	Fourier domain
Rectangular window	Sinc function
Sinc function	Rectangular window
Constant function	Dirac Delta function
Dirac Delta function	Constant function
Dirac comb (Dirac train)	Dirac comb (Dirac train)
Cosine function	Two, real, even Delta f.
Sine function	Two, imaginary, odd Delta f.
Gauss function	Gauss function

2.2. Discrete Fourier transform (DFT)

To compute the Fourier transform numerically on a computer, discretization plus numerical integration are required. This is an approximation of the true (i.e., mathematical), analytically-defined FT in a synthetic (digital) environment, and is called discrete Fourier transformation (DFT). There are three difficulties with the numerical computation of the FT:

- *Discretization* (introduces periodicity in both the time and the frequency domains)
- *Numerical integration* (introduces numerical error, approximation)
- *Finite time duration* (introduces maximum frequency and resolution limitations)

The DFT of a continuous time signal sampled over the period of T, with a sampling rate of Δt can be given as

$$S(m\Delta f) = \frac{T}{N} \sum_{n=0}^{N-1} s(n\Delta t) e^{-j2\pi\, m\Delta f\, n\Delta t} \tag{5}$$

where $\Delta f = 1/T$, and, is valid at frequencies up to $f_{max} = 1/(2\Delta t)$. Table 2 lists a simple Matlab m-file that computes (8) for a time record $s(t)$ of two sinusoids whose frequencies and amplitudes are user-specified. The record length and sampling time interval are also supplied by the user and DFT of this record is calculated inside a simple integration loop.

2.3. Fast Fourier transform (FFT)

The fast Fourier transform (FFT) is an algorithm for computing DFT, before which the DFT required excessive amount of computation time, particularly when high number of samples (N) was required. The FFT forces one further assumption, that N is an integer multiple of 2. This allows certain symmetries to occur reducing the number of calculations (especially multiplications) which have to be done.

Table 2. A Matlab module for DFT calculations.

```
%-------------------------------------------------------------------
%   Program : DFT.m
%   Author : Çağatay Uluışık, Levent Sevgi
%   Purpose : To calculate DFT of a time signal without Matlab function
%   Function : two sinusoids
%-------------------------------------------------------------------

%   Get the input parameters
    fr1=input('Frequency of the first sinusoid [Hz] = ?');
    a1=input('Amplitude of the first sinusoid [ ] = ?');
    fr2=input('Frequency of the second sinusoid [Hz] = ?');
    a2=input('Amplitude of the first sinusoid [ ] = ?');
    T=input ('Time record length [s] = ?');
    dt=input ('sampling time interval [s] = ?');
    fmax=input('maximum frequency for DFT [Hz] = ?');
    df=input('frequency sampling interval for DFT [Hz] = ?');

    N=T/dt; w1=2*pi*fr1; w2=2*pi*fr2; M=fmax/df;

%   Build the input time series
    for k=1:N
      st(k)=a1*sin(w1*dt*k)+a2*sin(w2*dt*k);
    end

%   Apply the DFT with M points
    for k=1:M
      Sf(k)=complex(0,0);
      for n=1:N
        Sf(k)=Sf(k)+st(n)*exp(-i*2*pi*n*dt*k*df);
      end
      Sf(k)=Sf(k)*dt;
    end

%   Prepare the frequency samples
    for k=1:M
      F(k)=(k-1)*df;
    end

%   Plot the output
    plot(F,abs(Sf));
    title('The DFT of the Sum of two Sinusoids')
    xlabel('Frequency [Hz]'); ylabel('Amplitude')
%------------------------------ End of DFT.m ------------------------------
```

To write an FFT routine is not as simple as DFT routine, but there are many internet addresses where one can supply FFT subroutines (including source codes) in different programming languages, from Fortran to C++. Therefore, the reader does not need to go into details, rather to include them in their codes by simply using "include" statements or "call" commands. In Matlab, the calling command is *fft(s,N)* for the FFT and *ifft(s,N)* for the inverse FFT, where s is the recorded N-element time array. A simple m-file is included in Table 3, which computes and plots FFT of a sine-modulated Gaussian function whose

parameters are supplied by the user.

Table 3. A Matlab module for FFT calculations

```
%-------------------------------------------------------------------
%  Program : FFT.m
%  Author : Çağatay Uluışık, Levent Sevgi
%  Purpose : To calculate FFT of a time signal with Matlab function
%  Function : sine modulated Gauss function
%-------------------------------------------------------------------
%  Get the input parameters
   N=input ('N = ?');
   fr=input('Modulation frequency [Hz] = ?');
   dt=input ('time step, dt [s] = ?');
   w=2*pi*fr;

%  Build the input time series
   for k=1:N
     X(k)=sin(w*dt*k)*exp(-(dt*N/(10*dt*k))^2);
   end

%  Apply the FFT
   X2=fft(X,N);
   X2=fftshift(X2); % swaps the left and right halves
   X2=abs(X2);

%  Prepare the frequency samples
   fmax=1/dt; df=1/(N*dt);
   for k=1:N
     F(k)=-fmax/2+(k-1)*df;
   end

%  Plot the output
   plot(F,X2);
   title('The FFT of a Time Signal')
   xlabel('Frequency [Hz]'); ylabel('Amplitude')
% ------------------------- End of FFT.m -------------------------
```

2.4. Aliasing effect, spectral leakage and scalloping loss

As stated above, performing FT in a discrete environment introduces artificial effects. These are called aliasing effects, spectral leakage and scalloping loss.

It should be kept in mind when dealing with discrete FT that:

- Multiplication in the time domain corresponds to a convolution in the frequency domain.
- The FT of an impulse train in the time domain is also an impulse train in the frequency domain with the frequency samples separated by $T_0 = 1/f_0$.
- The narrower the distance between impulses (T_0) in the time domain the wider the distance between impulses (f_0) in the frequency domain (and vice versa).

- The sampling rate must be greater than twice the highest frequency of the time record, i.e., $\Delta t \geq 1/(2f_{\max})$ (Nyquist sampling criterion).
- Since *time – bandwidth* product is constant, narrow transients in the time domain possess wide bandwidths in the frequency domain.
- In the limit, the frequency spectrum of an impulse is constant and covers the whole frequency domain (that's why an impulse response of a system is enough to find out the response of any arbitrary input).

If the sampling rate in the time domain is lower than the Nyquist rate *aliasing* occurs. Two signals are said to alias if the difference of their frequencies falls in the frequency range of interest, which is always generated in the process of sampling (aliasing is not always bad; it is called mixing or heterodyning in analog electronics, and is commonly used in tuning radios and TV channels). It should be noted that, although obeying Nyquist sampling criterion is sufficient to avoid aliasing, it does not give high quality display in time domain record. If a sinusoid existing in the time signal not bin-centered (i.e., if its frequency is not equal to any of the frequency samples) in the frequency domain *spectral leakage* occurs. In addition, there is a reduction in coherent gain if the frequency of the sinusoid differs in value from the frequency samples, which is termed *scalloping loss*.

3. Windowing and Window Functions

Using a finite-length discrete signal in the time domain in FT calculations is to apply a rectangular window to the infinite-length signal. This does not cause a problem with the transient signals which are time-bounded inside this window. But, what happens if a continuous time signal like a sine wave is of interest? If the length of the window (i.e., the time record of the signal) contains an integral number of cycles of the time signal, then, periodicity introduced by discretization makes the windowed signal exactly same as the original. In this case, the time signal is said to be periodic in the time record. On the other hand, there is a difficulty if the time signal is not periodic in the time record, especially at the edges of the record (i.e., window). If the DFT or FFT could be made to ignore the ends and concentrate on the middle of the time record, it is expected to get much closer to the correct signal spectrum in the frequency domain. This may be achieved by a multiplication by a function that is zero at the ends of the time record and large in the middle. This is known as *windowing* [5].

It should be realized that, the time record is tempered and perfect results shouldn't be expected. For example, windowing reduces spectral leakage but does not totally eliminate it. It should also be noted that, windowing is introduced to force the time record to be zero at the ends; therefore transient signals which occur (starts and ends) inside this window do not require a window. They are called *self-windowed* signals, and examples are impulses, shock responses, noise bursts, sine bursts, etc.

Other window functions (as opposed to the natural rectangular window which has the narrowest mainlobe, but the highest sidelobe level) are used to obtain a compromise between a narrow main lobe (for high resolution) and low sidelobes (for low spectral leakages). High frequency resolution provides accurate estimation of the frequency of an existing sinusoid and results in the separation of two sinusoids that are closely spaced in the frequency domain. Low spectral leakage improves the detectability of a weak sinusoid in the presence of a strong one that is not bin-centered [6].

Examples of common window functions are (*n = 0, 1,2, ... N-1*):

$$\textit{Rectangular:}\ \ W(n\Delta t) = 1 \tag{6a}$$

$$\textit{Hanning:}\ \ W(n\Delta t) = \frac{1}{2} - \frac{1}{2}\cos\left(\frac{2\pi n}{N}\right) \tag{6b}$$

$$\textit{Hamming:}\ \ W(n\Delta t) = 0.54 - 0.46\ \cos\left(\frac{2\pi n}{N}\right) \tag{6c}$$

$$\textit{Gaussian:}\ \ W(n\Delta t) = \exp\left\{\frac{-1}{2}\left(\frac{\alpha n}{N/2}\right)^2\right\} \tag{6d}$$

$$\textit{Blackman-Harris:}\ \ W(n\Delta t) = 0.42323 - 0.49755\ \cos\left(\frac{2\pi\, n}{N}\right) + 0.07922\ \cos\left(\frac{4\pi\, n}{N}\right) \tag{6e}$$

All of these window functions act as a filter with a very rounded top. If a sinusoid in the time record is centered in the filter then it will be displayed accurately. Otherwise, the filter shape (i.e., the window) will attenuate the amplitude of the sinusoid (scalloping loss) by up to a few dB (15-20 %) when it falls midway between two consecutive discrete frequency samples. The solution of this problem is to choose a flat-top window function [6];

$$\textit{Flat-top:}\ \ W(n\Delta t) = 0.2810639 - 0.5208972\ \cos\left(\frac{2\pi\, n}{N}\right) + 0.1980399\ \cos\left(\frac{4\pi\, n}{N}\right) \tag{6f}$$

which reduces the amplitude loss to a value less than 0.1 dB (1 %). However, this accuracy improvement does not come without its price; it widens the mainlobe in the frequency domain response (i.e., a small degradation in frequency resolution). It should be remembered that there is always a pay off between accuracy and resolution when applying a window function.

4. Digital filters (DF)

In electronics and communication, a device is said to be a filter if it transmits only part of the incident energy and thereby changes its spectral distribution. High-pass (low-pass) filters transmit energy above (below) a certain frequency. On the other hand, bandpass filters transmit energy of a certain bandwidth, while band-stop filters transmit energy outside a specific frequency band. The filtration may be performed both in frequency and time domains. Frequency domain filtration is performed in the following way:

- accumulate the time signal for a certain observation period,
- apply FFT and obtain frequency domain signal,
- suppress the unwanted components,

- apply inverse FFT and obtain filtered time domain signal.

For an online process, these steps must be completed before the next cycle of operation starts. If the electronic equipment is not fast enough to meet this time requirement, then the filtration has to be done directly in the time domain. Digital filters (DF) are used in this case.

4.1. The difference equations of the DF

A set of linear differential equations describes the behavior of a continuous-time, time-invariant linear system. The solution to the differential equations establishes the system's response to a given input signal. To model a discrete-time, time-invariant linear system using a computer, the corresponding differential equations as a system of difference equations must be set up. The solution to the difference equations establishes the discrete-time system response.

A discrete-time system has a difference equation of the form:

$$y(n) = \sum_{l=1}^{N} a_l \, y(n-l) + \sum_{k=0}^{M} b_k \, x(n-k) \tag{7}$$

One can develop digital bandpass, notch, or highpass filters that mimic the behavior of the RLC filters. Furthermore, high-order digital filters are possible. The type of filter and its performance depend on the values selected for the coefficients a_l and b_k.

4.2. A Simple Notch Filter

A simple way to obtain a notch filter is to select $a_l = 0$ for all $l, b_0 = 0.5$, $b_d = 0.5$ and to set the remaining b_k coefficients to zero. Then, the output of the digital filter is given by

$$y(n) = 0.5x(n) + 0.5x(n-d) = 0.5[x(n) + x(n-d)] \tag{8}$$

Thus, each input sample is delayed in time by $\Delta\mathrm{t} \times \mathrm{d}$ and added to the current sample. ($\Delta\mathrm{t}$ represents interval between samples.) Finally, the sum of the input and its delayed version is multiplied by 0.5. To see that this results in a notch filter, consider a sine wave delayed by an interval $\Delta\mathrm{t} \times \mathrm{d}$. In this case, one can write

$$A\cos\left[w\ (t - \Delta t\, d)\right] = A\cos(wt - w\Delta t\, d) = A\cos(wt - \theta) \tag{9}$$

Thus, a time delay of $\Delta\mathrm{t} \times \mathrm{d}$ amounts to a phase shift of $w\Delta t\, d$ radians or $f\, \Delta t\, d \times 360°$. For low frequencies, the phase shift is small, so the low-frequency components of $x(n)$ add nearly in phase with those of $x(n-d)$. On the other hand, for the frequency

$$f_{notch} = \frac{1}{2\,\Delta t\, d} = \frac{f_s}{2\,d} \tag{10}$$

the phase shift is $180°$. Of course, when the phase of the sine wave is shifted by $180°$ and added to the original, the sum is zero. Thus, any input component having the frequency f_{notch} does not appear in the output. This simple notch filter is just one of many possible digital filters that can be realized by selection of the coefficient values in equation (8).

5. A LabVIEW-based Virtual FFT Instrument (DOGUS_FFT)

With what and to what extend a student will be equipped in engineering education should be specified according to the rapid shifts in societal needs. This is a challenge especially for newly founded universities and institutes because of the limited budgets. Fore example, Electronics and Communication Engineering (ECE) Department at Doğuş University, Istanbul, is a newly founded department and is in the process of establishing undergraduate as well as graduate level labs. It is an extremely hard optimization problem to establish low cost, highly effective labs which provide the rich spectrum of experimentation as required for ECE students. The first level labs are almost standard and no serious problems were encountered in establishing them. However, this is a real problem for the higher level labs. After serious investigation optimization resulted in favor of the newly developed National Instrument set, Educational Lab Virtual Instrumentation suit (NI-ELVIS) which is shown in Figure 2.

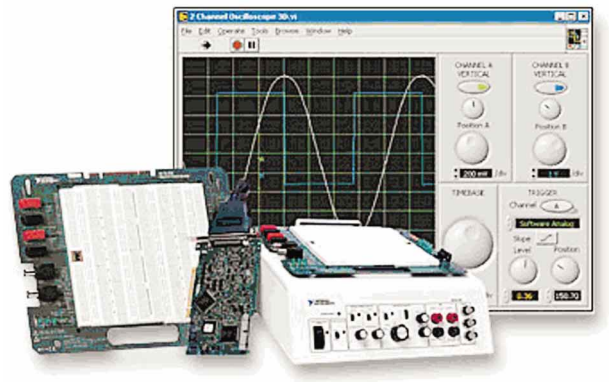

Figure 2. A novel lab experimental setup NI-ELVIS (a DAC card for a PC, a custom-designed benchtop workstation, and prototyping board)

NI-ELVIS consists of LabVIEW-based virtual instruments, a multifunction data acquisition (DAC) device, a custom-designed benchtop workstation, and prototyping board. This combination provides a ready-to-use suite of instruments found in regular educational laboratories. Because it is based on LabVIEW and provides complete data acquisition and prototyping capabilities, the system is "*good*" for simple experimentations, for hands-on training, and academic courses from lower-division classes to advanced project-based curriculum. The major difficulty in using NI-ELVIS (or similar setups) is to prepare suitable experiments that balance analog parts for the benchtop workstation prototyping board against digital study in the connected computer. Fortunately, this does not cause serious problems for some of the virtual instruments that are used in education as well as research, such as FT tools, digital receiver blocks, etc.

A virtual instrument has been prepared as an FFT tool with the LabVIEW Express module. It can generate and display time and frequency domain behaviors of (a) two sinusoids, (b) a rectangular pulse, (c) a pulse train, (d) a Gaussian function, (e) a sine modulated Gaussian function, and finally (f) any set of real data from benchtop, all of whose parameters can be specified by the user. Its block diagram and front window are given in Figures 3 and 4, respectively. The block diagram in Figure 3a consists of mainly *a while-loop*, *two case-structure* and *one for-loop*. The while-loop contains time domain data generation blocks, and the process repeats until the *exit button* on the front panel is pressed. One of the time domain data generation blocks is presented in Figure 3b, which is used for the summation of two sinusoids. The integration period T, and the sampling interval Δt are supplied by the user from the front panel. The

Table 4. A Matlab module for signal generation with a given SNR.

```
%-------------------------------------------------------------------
%  Program : SNR.m
%  Author : Çağatay Uluışık, Levent Sevgi
%  Purpose : To generate N-element array of a sinusoid with a
%  given SNR and uniform noise distribution
%-------------------------------------------------------------------

%  Get the input parameters
   SNRdB=input('Signal to noise ratio (SNR) [dB] =');
   SNR=10^(SNRdB/10);
   Vm=input('Amplitude of the sinusoid (Vm) =');
   f=input('Sinusoid frequency (f) =');
   T=input('Observation Period (T) =');
   dt=input('Sampling Rate (dt) =');
   N=T/dt;

%  Calculate Signal Power Ps :
   Ps=0.5*Vm^2;

%  Calculate Noise Power Pn :
   Pn=0.5*Vm^2/SNR;

%  Calculate the fluctuation limits of the uniformly distributed noise Nm :
   Nm=sqrt(1.5*Vm^2/SNR);

%  Generate an N element array of uniformly distributed random numbers Uk :
   Uk=rand(1,N);

%  Generate Signal
   for k=1:N
     t(k)=k*dt;
    Sk(k)=Vm*sin(2*pi*f*k*dt);
   end

%  Change the variance of the Noise
   Nk=2*Nm*Uk-Nm;

%  Add Noise to the Signal
   Vk=Sk+Nk;

%  Plot the outputs
   subplot(3,1,1)
   plot(t,Nk); ylabel('Noise')
   subplot(3,1,2)
   plot(t,Sk); ylabel('Signal')
   subplot(3,1,3);
   plot(t,Vk); xlabel('Time [s]'); ylabel('Signal+Noise')
%------------------------- End of SNR.m -------------------------
```

for-loop gets the ratio of $N=T/\Delta t$ (the number of iterations) as an input. The counter k, ranges from 0 to n-1. The frequencies of the two sinusoids f_1 and f_2 are also specified by the user on the front panel. The products "$2\times\pi\times f_1\times k\times\Delta t$" and "$2\times\pi\times f_2\times k\times\Delta t$", *are fed into* the sine-block at each iteration in the loop. The amplitude of the first sinusoid is A_1 and is supplied by the user, but the second one is fixed to unity. The sum of these two forms the time record (the output array) of the loop and is fed into the standard LabVIEW FFT block. The second case-structure is reserved for the windowing. Currently, five cases are used; (0) None, (1) Hanning, (2) Hamming, (3) Gaussian, (4) Blackman-Harris, (8) FlatTop. Another block is reserved for the noise generation and addition. Finally, a block is reserved for a notch filter.

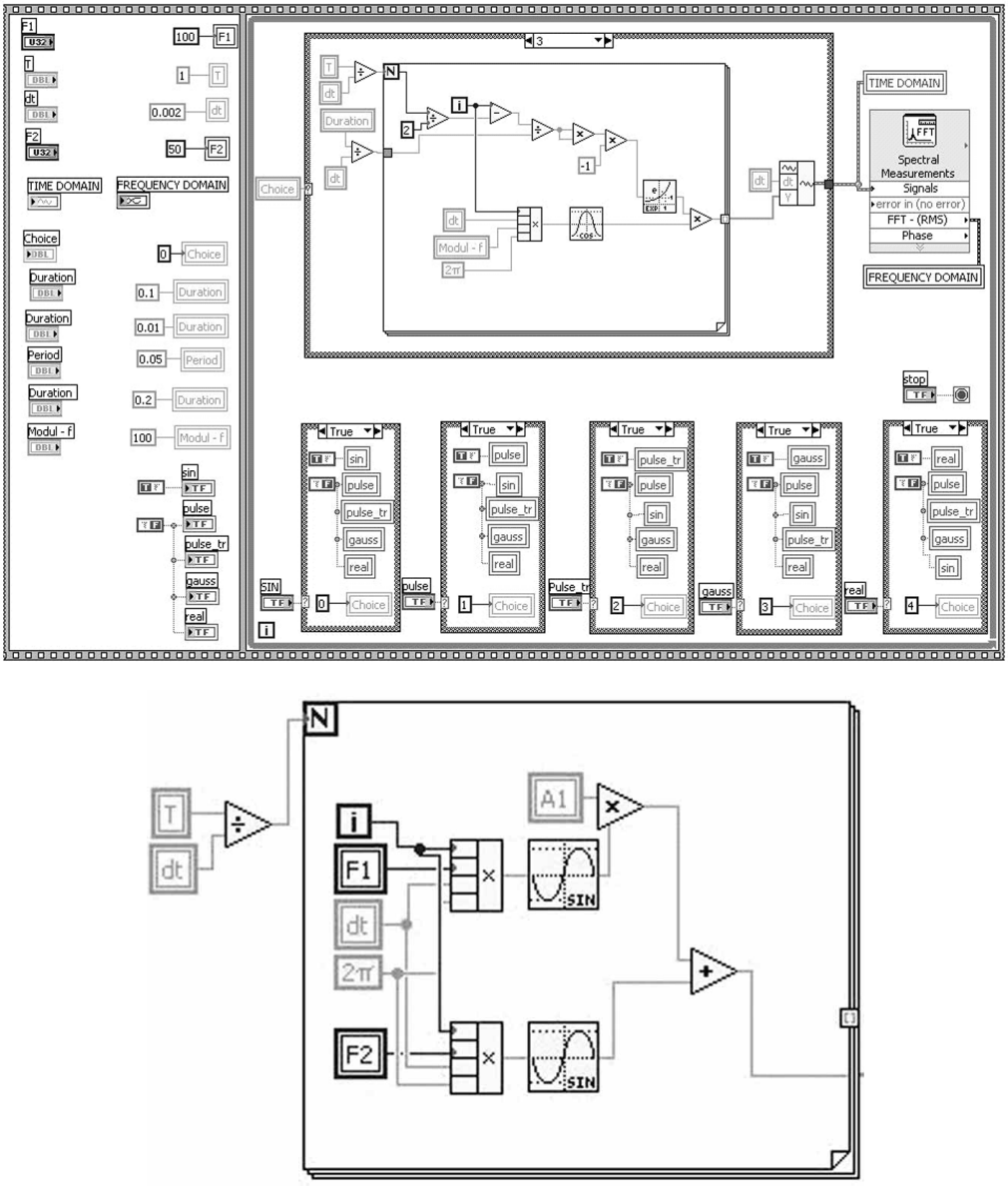

Figure 3. (a) The block diagram of the FFT virtual instrument, (b) The sub-block for generation of the time record that contains two sinusoids

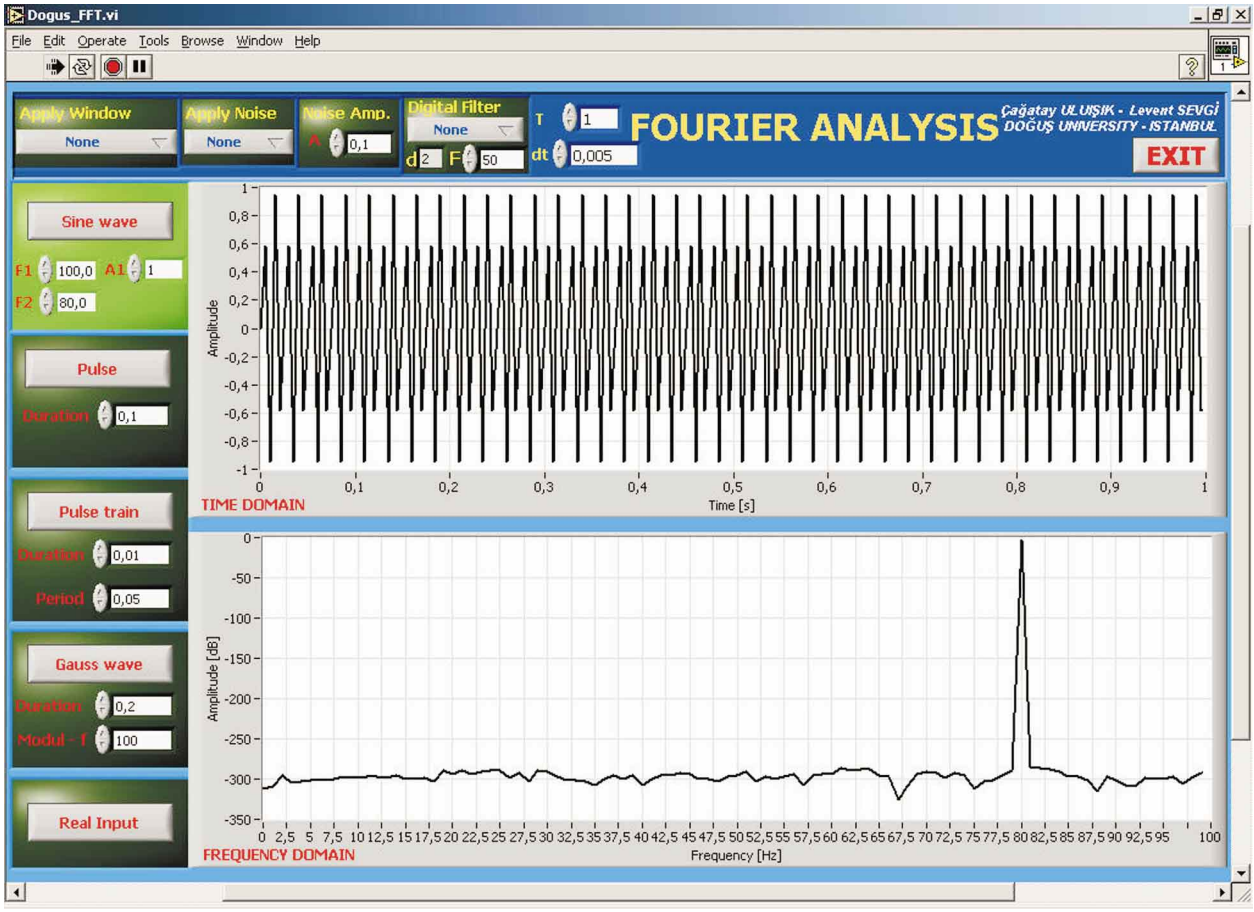

Figure 4. The front panel of the FFT virtual instrument.

In Figure 4, the front panel is shown with an example for two sinusoids. The frequencies of the sinusoids are $f_1 = 100\ Hz$ and $f_2 = 80\ Hz$, respectively, with equal amplitudes. The top window presents time plot of this function. The integration time and the sampling interval are shown to be $T = 1\ s$ and $\Delta t = 0.005\ s$, respectively. The bottom window displays the FFT of this function. It is interesting to see that although there are two sinusoids, only one of them appears in the FFT plot, since maximum frequency overlaps with one of the sinusoids and Nyquist criterion is not satisfied. The reason of this and more are given in the next section with characteristic examples and illustrations.

The FFT VI has also a module for real inputs. A real input may be received from the function generator on benchtop workstation as well as from a user-built circuit on the NI-ELVIS prototyping board via the DAC card installed in the PC. Therefore, the virtual instrument may also be used as an experimentation tool for real data.

5.1. Noise modeling in terms of Signal to Noise ratio (SNR)

The signals we are interested in are time-varying and most of them are without DC component (i.e., average power is zero). The primary factor in determining the floor for the signal spectrum in the frequency domain is the noise, and is called the receiver sensitivity. The noise may be produced in the receiver itself, since every electronic element (active devices, resistances, etc.) causes noise (thermal noise, shot noise, contact or

flicker noise, etc.), or it may be environmental. Which ever is the stronger, it limits the sensitivity of the receiver. For example, environmental noise at HF frequencies (*3 - 30 MHz*) dominates the receiver noise (30 to 70 dB higher) therefore we do not need to concern for the latter. On the other hand, receiver noise becomes to dominate at VHF, UHF and above, which means environmental noise is negligible.

Noise $n(t)$ is a random process and it may be simulated statistically. The complete statistical description of the noise (i.e., a one-dimensional random variable) can be specified and numerically generated by using its probability distribution function (PDF), and/or probability density function (pdf) [7]. Some well-known random variables are uniform-, Gaussian-, and Rayleigh-distributed, and they can be given as

Uniform: (mean: m=(a+b)/2, variance: σ^2 =(b-a)^2/12)

$$\text{Variable range:} \quad a < n < b \tag{11a}$$

$$\text{PDF} \quad F(n) = \begin{cases} 0 & n < a \\ \frac{n-a}{n-b} & a \le n \le b \\ 1 & n > b \end{cases} \tag{11b}$$

$$\text{Pdf} \quad f(n) = \frac{1}{b-a} \tag{11c}$$

Gaussian (Normal) (m^2: DC power, σ^2: AC power)

$$\text{Variable range:} \quad -\infty < n < +\infty \tag{12a}$$

$$\text{PDF} \quad F(n) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{n} \exp\left\{-\frac{(t-m)^2}{2\sigma^2}\right\} dt \tag{12b}$$

$$\text{Pdf} \quad f(n) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left\{-\frac{(n-m)^2}{2\sigma^2}\right\} dt \tag{12c}$$

Rayleigh (σ^2: variance)

$$\text{Variable range:} \quad 0 < n < +\infty \tag{13a}$$

$$\text{PDF} \quad F(n) = \begin{cases} 0 & n \le 0 \\ 1 - \exp\left\{\frac{-n^2}{2\sigma^2}\right\} & n > 0 \end{cases} \tag{13b}$$

$$\text{Pdf} \quad f(n) = \frac{n}{\sigma^2} \exp\left\{-\frac{n^2}{2\sigma^2}\right\} \tag{13c}$$

5.2. Generation of Signal and Noise as random signals

The commercial software packages and compilers (e.g., Matlab, Fortran, C++, etc.) have standard built-in functions that generate a random variable with uniform distribution. If U is a random variable with uniform distribution (between zero and one) then random variables with Gaussian and Rayleigh distributions may be generated as follows:

Uniform (with the lower and upper limits of a and b, respectively)

Step 1: Give the lower and upper boundaries (a, b)

Step 2: Generate a random number with uniform distribution (u)

Step 3: Transform to the new uniform number via

$$n = (b - a)u + a \tag{14}$$

Gaussian

Step 1: Give the mean and variance $(m,\ \sigma^2)$

Step 2: Generate two random numbers with uniform distribution $(u_1$ and u_2

Step 3: Transform to Standard Gaussian random number n_0 via

$$n_0 = \sqrt{-2\ln(u_1)}\cos(2\pi u_2) \text{ (zero mean, unit variance)} \tag{15a}$$

Step 4: Transform to Gaussian random number n_g with mean m and variance σ.

$$n_g = m + \sigma\, n_0 \tag{15b}$$

Rayleigh

Step 1: Give the variance (σ^2)

Step 2: Generate a random number with uniform distribution (u)

Step 3: Transform to Standard Rayleigh random number n_0 via

$$n_0 = \sqrt{-2\ln(u)} \tag{16a}$$

Step 4: Transform to Rayleigh random number n_r with variance σ^2.

$$n_r = \sigma\, n_0 \tag{16b}$$

A sequence of random numbers is also called a random signal. The average (or the mean) and the variance of the random signal can be calculated from

$$m = \frac{1}{N}\sum_{k=1}^{N} S_k \tag{17}$$

$$\sigma^2 = \frac{1}{N}\sum_{k=1}^{N} S_k^2 - m^2 \tag{18}$$

respectively. The theoretical mean and variance of a random signal with uniform distribution can be expressed as [7]

$$m_u = \frac{\text{upper boundary - lower boundary}}{12} \tag{19a}$$

$$\sigma_u^2 = \frac{(\text{upper boundary - lower boundary})^2}{12} \tag{19b}$$

5.3. Generating a sequence with a desired SNR and noise characteristics

It is often required to generate a sequence that contains both the signal and the noise. The amount of noise added to the signal can be specified with a signal to noise ratio (SNR). The SNR is defined in terms of power of a signal. The mean power in a (of a sequence of) signal s can be calculated from

$$power \approx \frac{1}{N}\sum_{k=1}^{N} S_k^2 \tag{20}$$

If a signal is a sinusoid then the signal power is equal to one-half the square amplitude of the sinusoid (e.g., the power of $10 \times sin(\omega\ t)$ is 50). SNR is the ratio of the power in a signal to the power in the noise. For example $SNR = 1$ specifies that the noise and the signal have equal powers. An array of random numbers with a sinusoid, band limited white noise and a given SNR [dB] may be generated as follows:

Step 1: Give *SNR [dB]*

Step 2: Calculate SNR from $SNR=10\ \hat{}\ (SNR_{dB}/10)$

Step 3: Calculate signal power from $s(t) = V_m\, sin(\omega_o t)$ as $P_s = (V_m)^2/2$

Step 4: Calculate noise power P_n from $SNR = P_s/P_n$ as $P_n = (V_m)^2/(2SNR)$

Step 5: Calculate the fluctuation limits of the uniformly distributed noise from $P_n = (2N_m)^2/12$ as

$$N_m = \sqrt{\frac{3\ V_m^2}{2\,SNR}}$$

Step 6: Generate an N-element array of uniformly distributed random numbers u_k *(k=1,2, ...N)*

Step 7: Change the variance of the noise : $n_k = 2N_m u_k$ - N_m

Step 8: Generate the signal as $s_k = V_m\, sin(\omega_o k\, \Delta\, t)$

Step 9: Add noise to the signal $v_k = s_k + n_k$

To illustrate, suppose it is desired to generate a noisy voltage sinusoid $v(t)=s(t)+n(t)$ with 1 V amplitude and SNR of 10 dB with (zero mean) uniform noise distribution. The signal power is equal to 0.5. A 10 dB SNR means the noise power is ten times less than the signal power; therefore the noise power will be .05. By equating this value to the variance in (14) (which is the noise power) one can calculate the range of noise amplitude fluctuations from $0.05 = (2a_0)^2/12$ (since the noise is uniform and ranges between $\pm a_0$) as $a_0 \approx 0.39V$. Finally, the noise sequence will be generated from (14) with $b = a_0 =$ *0.39*, $a = -a_0 = -$ *0.39*.

6. Numerical Illustrations

The first example in this section belongs to the correct parameterization of the previous figure. Figure 5 illustrates the two equal-amplitude sinusoids (f_1 =*100 Hz* and f_2 =*80 Hz)* correctly displayed in the time and the frequency domains. Here, the integration time and the sampling interval are chosen as $T = 1s$ and Δt = *0.002 s*, respectively. This yields a maximum frequency of 250 Hz, which covers both of the sinusoids. The corresponding frequency resolution is 1 Hz, which results in oversampling compared to the Nyquist criterion, but is good for high quality time display. Finally, the sinusoids are exactly at their correct locations in the frequency domain, since the integration time is integer multiple of the periods of both sinusoids.

The second example also belongs to the two sinusoid choice, but in this case with insufficient frequency resolution. Here, the sinusoids are separated by only 5 Hz (f_1 =*100 Hz* and f_2 = *97 Hz*). The sampling interval is specified as Δt = *0.0025 s*, and it corresponds to the maximum frequency of *200 Hz*. The integration time is T = *0.4 s* and yields a resolution of Δf = *2.5 Hz*. Although the frequency resolution is enough to discriminate the two sinusoids *5 Hz* apart from each other, they can not be identified in the display as shown in the figure. The reason for that is the integration time which is the integer multiple of the period of the first sinusoid (*100 Hz*) but not the other (97 *Hz*).

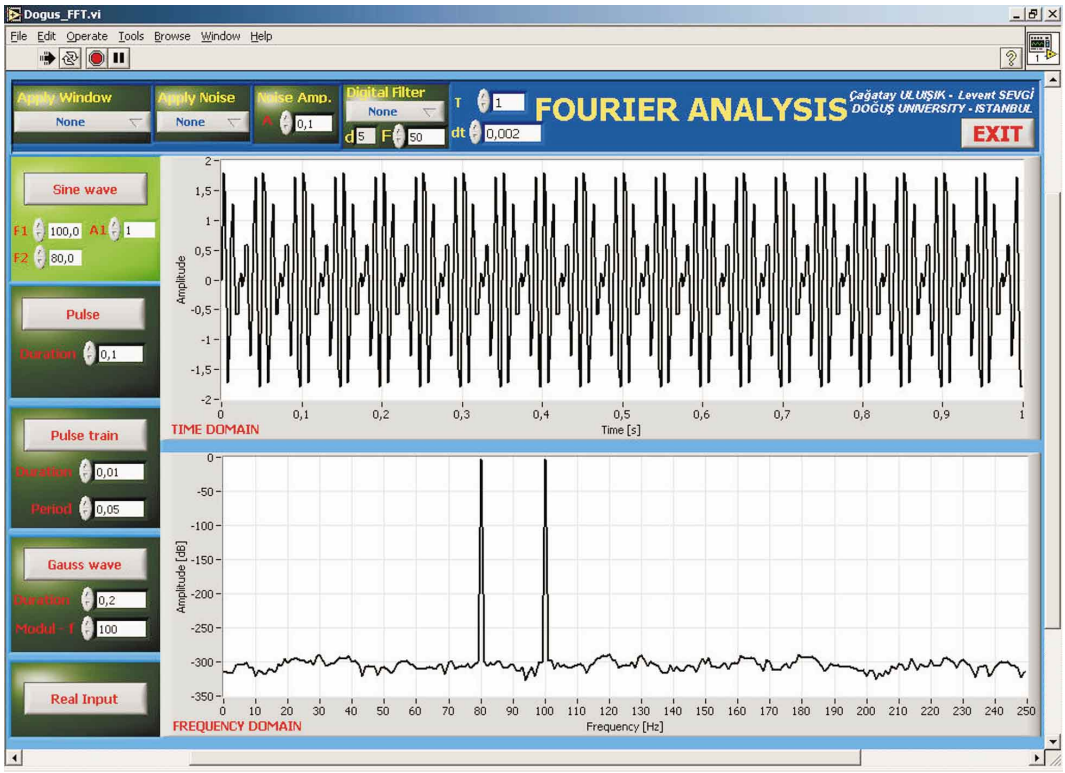

Figure 5. The front panel for the two sinusoid problem (f_1 =*100 Hz,* f_2 =*80 Hz,* Δt = *0.002 s, corresponds to* f_{max} = *250 Hz,* T = *1 s, corresponds to* Δf = *1 Hz).*

The third example belongs to the rectangular pulse and pulse train functions. Figure 7 shows a single rectangular pulse and its FFT spectrum. As given above, the FFT of a rectangular pulse is a Sinc. function, and amplitude of the first sidelobe is 13 dB below the mainlobe. The width of the mainlobe is controlled by the pulse duration and the proportionality between them is inversely. Figure 8 illustrates the situation for a rectangular pulse train. In this case, the frequency spectrum is a train of Sinc. functions inside the

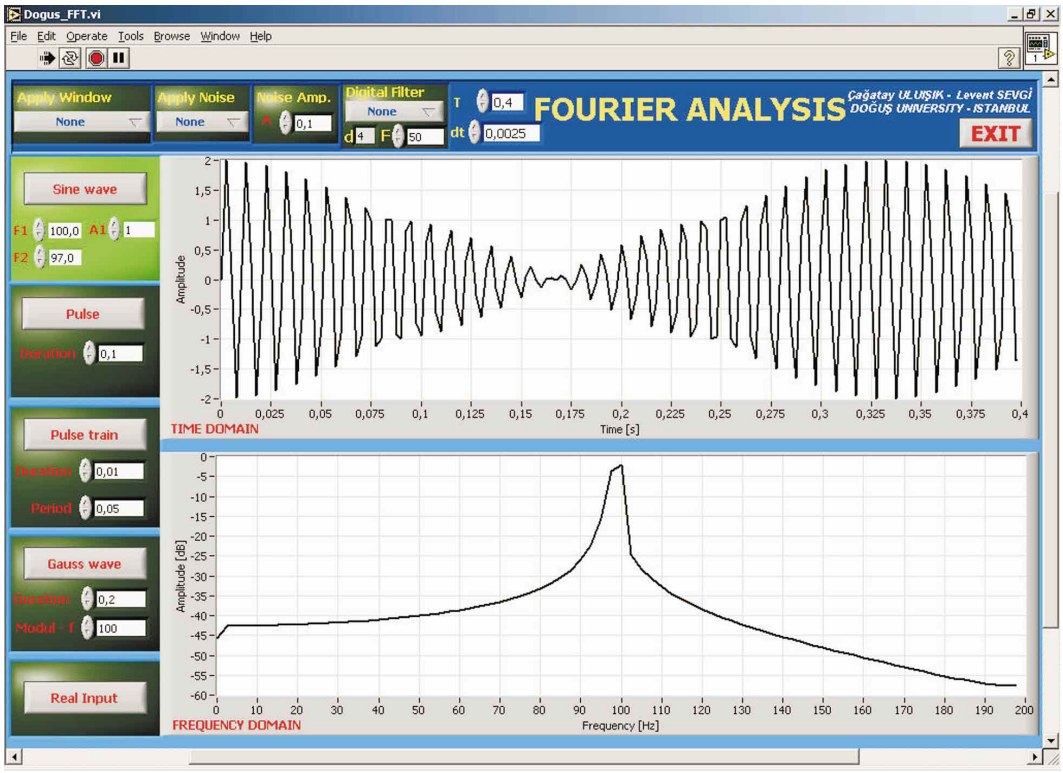

Figure 6. Another display for the two sinusoid problem with insufficient frequency resolution (f_1 =*100 Hz,* f_2 =*97 Hz,* Δt = *0.0025 s, corresponds to* f_{max} = *200 Hz,* T = *0.4 s, corresponds to* Δf = *2.5 Hz).*

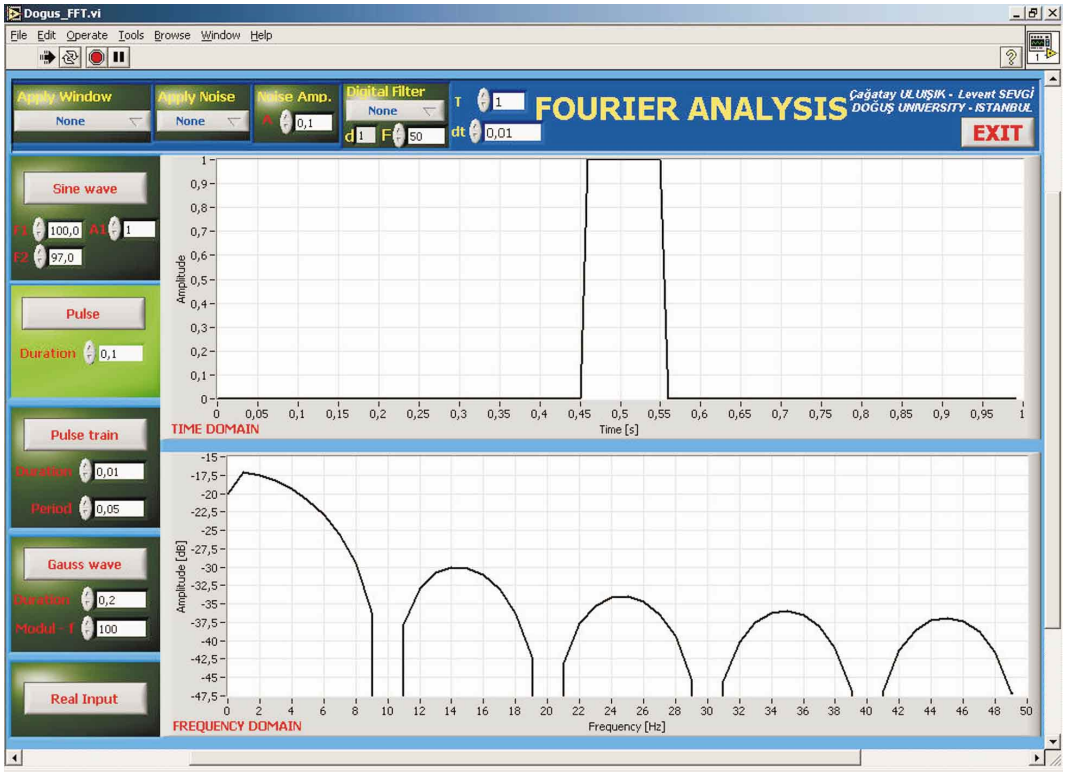

Figure 7. The front panel for the rectangular pulse and its spectrum (T = *1 s,* Δt = *0.01 s,* pulse duration = *0.1 s* and it yields *10 Hz* lobe widths in the frequency domain).

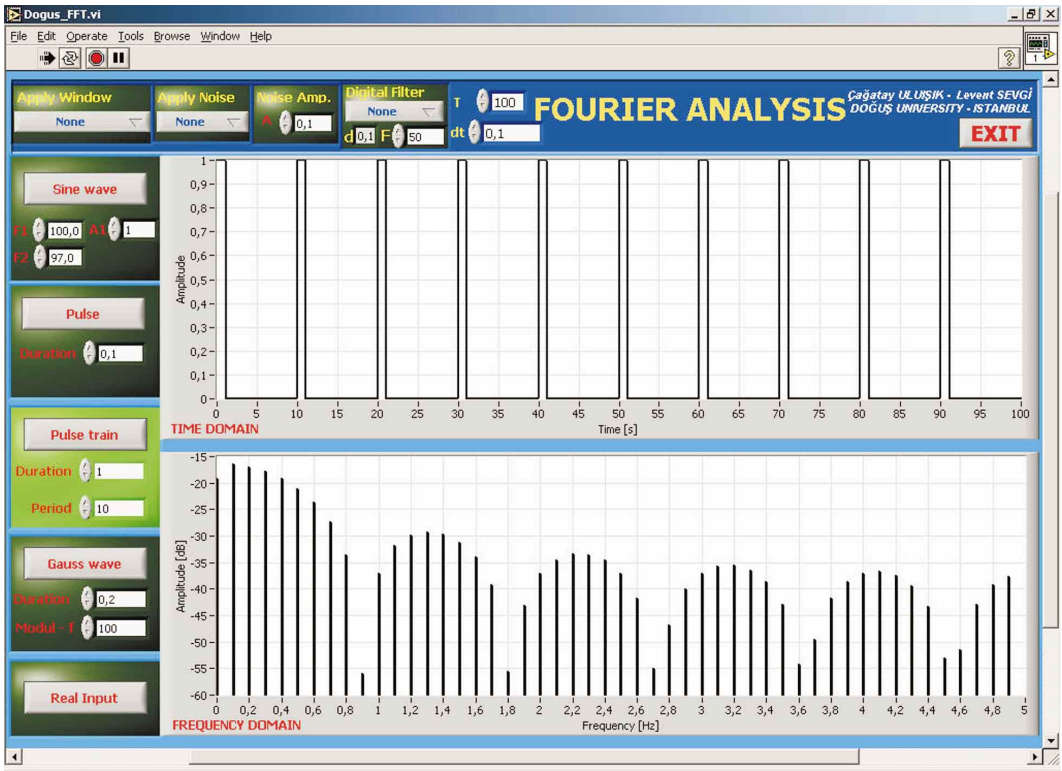

Figure 8. The front panel for the rectangular pulse train and its spectrum ($T = 100\ s$, $\Delta t = 0.1\ s$, pulse duration $= 1\ s$).

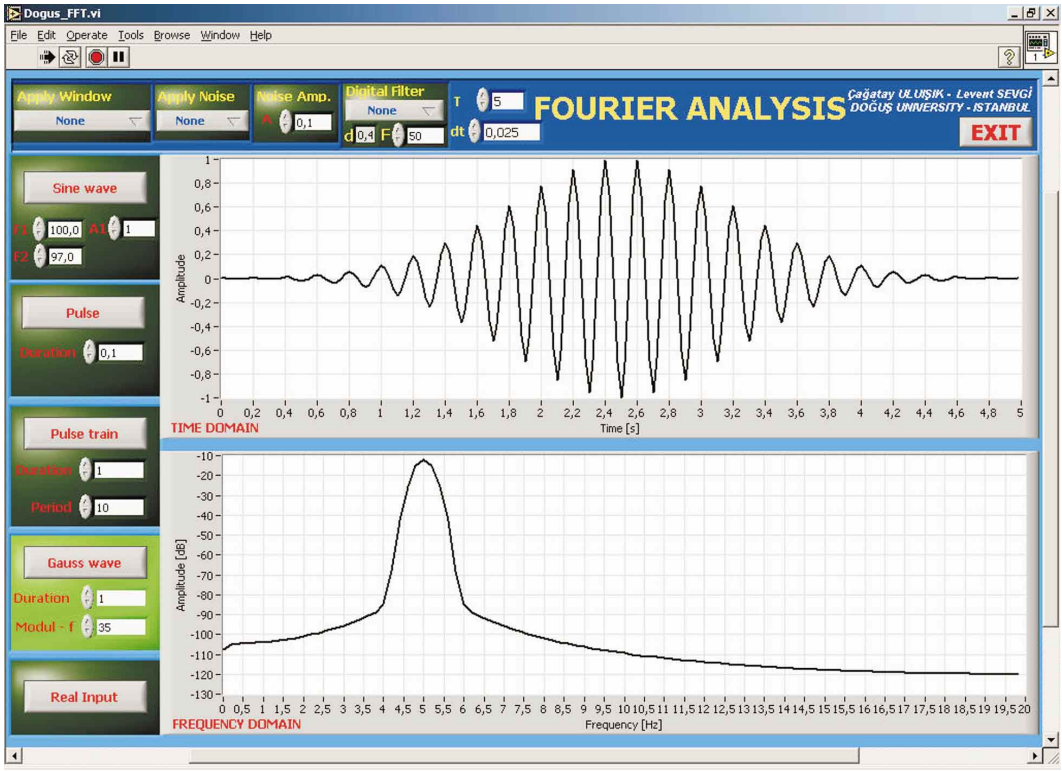

Figure 9. The front panel for a sine-modulated Gaussian functions ($T = 5\ s$, $\Delta t = 0.025\ s$, Gaussian function duration $= 1\ s$, modulation frequency $f = 35\ Hz$). A sampling rate of 0.025 s corresponds to $f_{max} = 20\ Hz$, therefore aliasing occurs and instead of $35\ Hz$, the frequency spectrum is centered at $5\ Hz$.

main Sinc. function. Again the width of the mainlobe is controlled by the pulse duration, but the distance between the Sinc train is specified by the period of the rectangular pulse train. The Sinc train is hardly noticed in the display because of the parameters chosen. One can play with the parameters and obtain better display in the frequency domain.

The fourth example is given in Figure 9 for a sine-modulated Gaussian function. Although the Gaussian function has an infinite extend, its duration may be controlled by its parameters and a finite-length record may be generated with a small error. The FT of a Gaussian function is also a Gaussian function; its width in the time domain is inversely proportional with the width (i.e., the band) in the frequency domain. Modulating the Gaussian function with a sine wave shifts the spectrum to the positive and negative sinusoid frequencies.

One example concerning the selection of a window function is the compromise between narrow mainlobe and low sidelobe level [8]. The effect of sidelobe levels on the ability to detect weak sinusoidal signal in the presence of strong sinusoidal signal that is not bin-centered is given in Figure 10. Here, two sinusoids are present in the time record; the weak sinusoid's frequency and amplitude are *107 Hz* and 0.01, respectively, while the strong one is at 101.5 Hz with a unit-amplitude. Since, integration time and sampling interval are chosen as $T = 0.987\ s$ and $\Delta t = 0.003\ s$, the frequency samples in the frequency domain starts from $-f_{\max} = 166.670\ Hz$ and present at all points separated by $\Delta f = 1.013\ Hz$. Therefore, the two sinusoids are not bin-centered. For a sinusoid not bin-centered (that is, its frequency is not equal to any of the discrete frequencies) in the frequency domain results in non-zero output at each bin. This spectral leakage effect is clearly observed in Figure 10a. Figure 10b shows the ability of various window functions to discern the presence of the second signal.

Another example belongs to the illustration of the effects of the added noise. Figure 11 shows a display for a single *40 Hz* sinusoid having 1 V- amplitude, and added noise with uniform distribution (having zero mean and a variance of 1 V). From above equations *SNR* is found to be 1.5 (or *1.8 dB*). It is clearly observed that although the sinusoid is hardly identified in the time domain, the peak at 30 Hz in the frequency domain can still be identified. It should be noted that the vertical axis is in dB-scale and the average floor is at *-320 dB (theoretically, it must be* $-\infty$*)* and *-30 dB*, without and with the noise added, respectively. It should be noted that, SNR yields relative power difference between the signal and the noise levels; absolute noise level (the detection threshold) is determined from the physical characteristics of the noise. For example, if the thermal noise is dominant then average noise floor can be calculated from $N_0 = kTB$, where $k = 1.38 \times 10^{-23}$ [Joules/°K] is the Boltzman's constant, T is the temperature [°K]in degrees Kelvin, and B [Hz] is the signal bandwidth. The effect of added noise is also shown in Figure 12 for a single rectangular pulse signal. Here, the amplitude and the duration of the pulse are 1 V and *0.25 s, respectively.* The noise amplitude (i.e., variance) is chosen 0.5 V.

The last example is reserved for the notch filter demonstration. A 50 Hz sine modulated Gaussian signal is passed through the notch filter of 50 Hz and its time and frequency domain variations are given in Figure 13. The sharp filtration at 50 Hz is clearly observed.

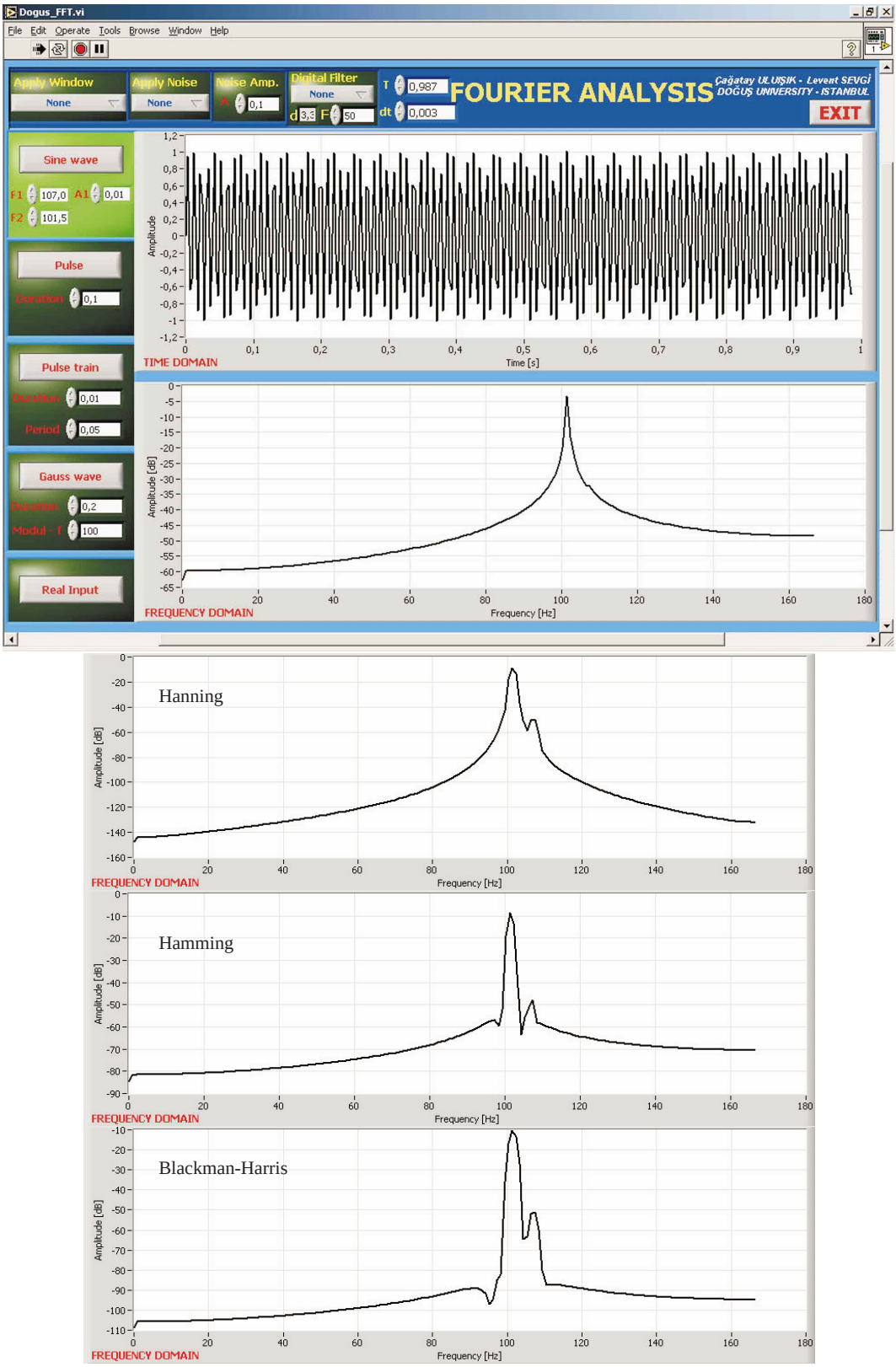

Figure 10. Response of various window functions to a time record in the presence of two sinusoidal signals (a) none, (b) Hanning, Hamming, and Blackman-Harris ($f_1 = 107\ Hz$, $A_1 = 0.01$, $f_2 = 101.5\ Hz$, $A_2 = 1.0$, $T = 0.987\ s$, $\Delta t = 0.003\ s$)

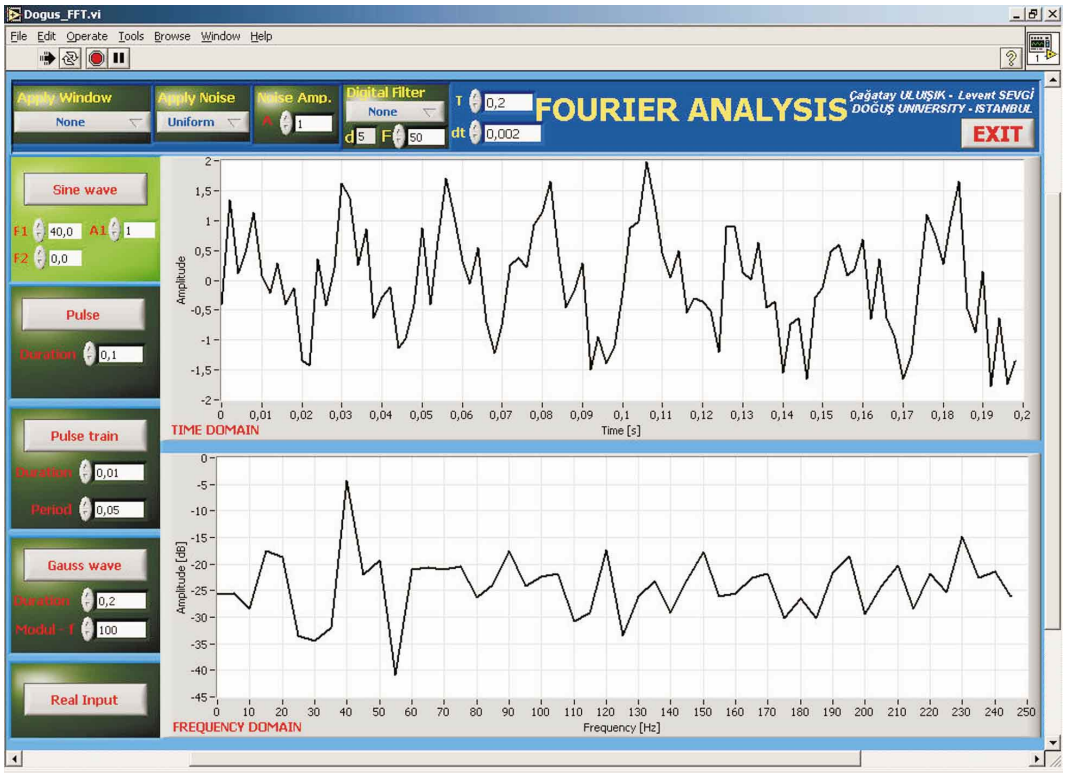

Figure 11. The front panel to illustrate signal (a sinusoid) plus (uniform) noise effects ($f_1 = 40$ *Hz*, $A_1 = 1$, $f_2 = 0$ *Hz*, $T = 0.2$ *s*, $\Delta t = 0.002$ *s*, uniform noise, noise amplitude is *1 V*).

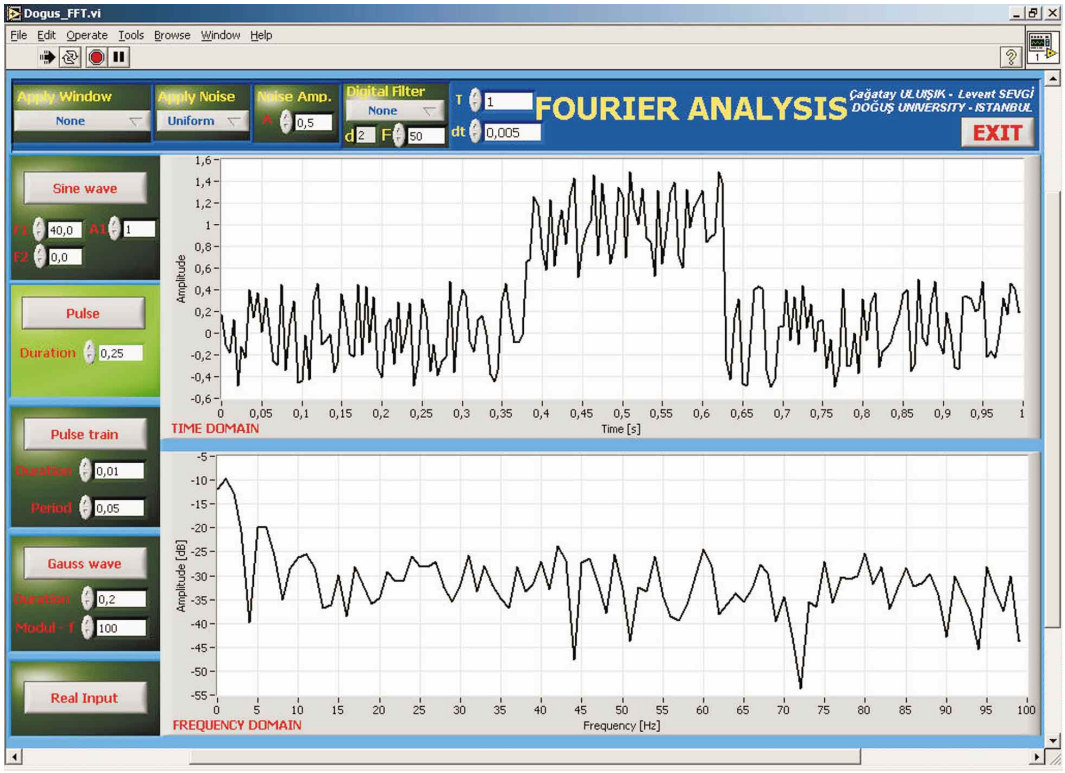

Figure 12. The front panel to illustrate signal (rectangular pulse) plus (uniform) noise effects (pulse duration = *0.25 s*, pulse amplitude = *1 V*, $T = 1$ *s*, $\Delta t = 0.005$ *s*, uniform noise, noise amplitude is *0.5 V*).

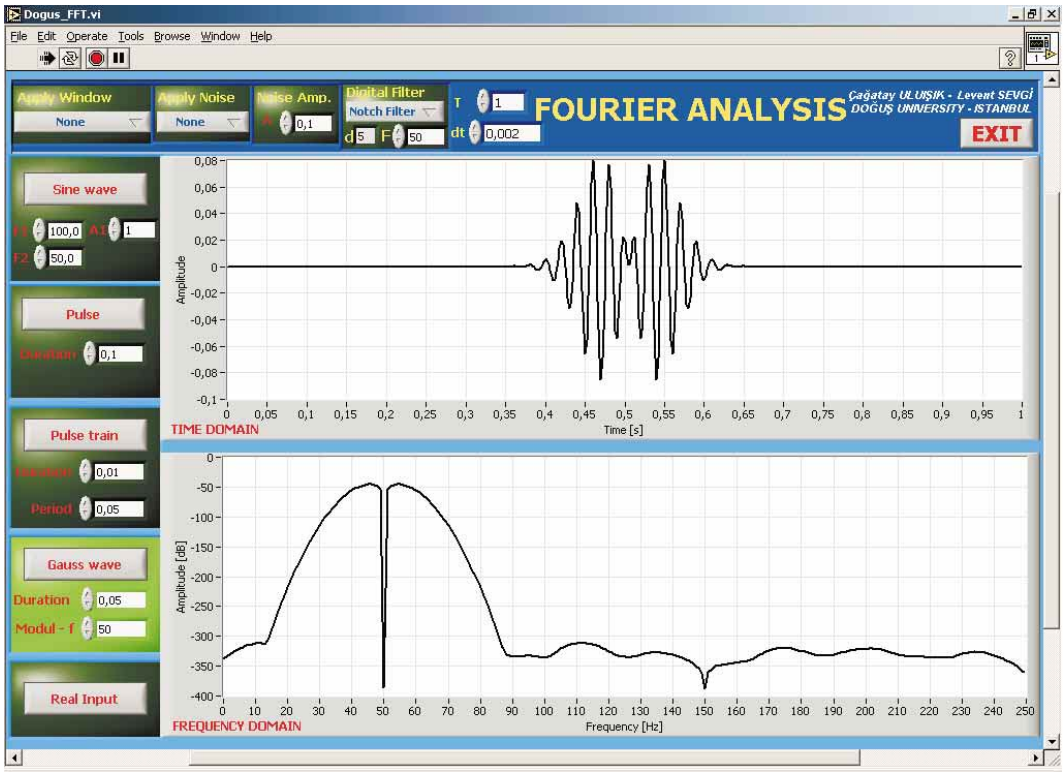

Figure 13. The front panel for the modulated Gaussian signal at the output of a notch filter ($f_{notch} = 50\ Hz$, $T = 1\ s$, $\Delta t = 0.002$ s, Gaussian function duration $= 0.05\ s$, modulation frequency $f = 50\ Hz$). A sampling rate of $0.002\ s$ corresponds to $f_{max} = 250\ Hz$, therefore aliasing do not occur). The sharp filtration at $50\ Hz$ is clearly observed.

7. Conclusions

A LabVIEW-based virtual instrument (VI) is designed for Fourier transformation (FT) operations in this paper. The VI may be used for both computer generated time data and real experimental data that are recorded by means of a DAC card inserted in the connected PC. The VI is an extremely handy tool for the lecturers who teach FT, as well as for the students (even for the researchers) who need to do many tests to understand FT problems such as aliasing, spectral leakage, and scalloping loss. The reader may enjoy doing experiments with the VI and then may discuss the results. It is really educating to parameterize a time record and write down the expectations first, and then to run the experiment with the presented VI and comment on the results (or vice versa). It is our experience from our environment that without these kinds of numeric tests one can *learn* but not *understand* the FT in detail. We have met people who have textbooks on FT, but still miss some of the concepts (which can not be well-understood without practice), since they only deal mathematical definitions and derivations of the FT.

Today, engineering education possesses challenges such as physics-based modeling, observation-based parameterization, computer-based simulations and code verification against canonical problems, etc. As phrased by Einstein - *in the matter of physics, first lessons should contain nothing but what is experimental and interesting to see* - experimentation and hands-on training are the key issues in engineering education. On the other hand, with the development of new computer technologies, interactive multimedia programming languages (e.g. JAVA), and the World Wide Web, it is now possible to build virtual engineering and science laboratories that can be accessed from all around the world. Experiment-oriented problems can be offered without the overhead incurred when maintaining a full laboratory. At this point the question arises: should an intelligent balance be established between real and

virtual experimentations and how [9-12]?

Acknowledgment

The authors acknowledge Mr. Serdar Bölükbaşıoğlu from E3TAM, Istanbul, for his assistance in preparation of some of the block diagrams of the *vi*-file.

References

[1] Baron Jean Baptiste Fourier (see, e.g., http://bartleby.com/65/fo/Fouriers.html)

[2] HP, "The Fundamentals of Signal Analysis", Hewlett Packard Application note 243, 1994

[3] L. Sevgi, F. Akleman, L. B. Felsen, "Ground Wave Propagation Modeling: Problem-matched Analytical Formulations and Direct Numerical Techniques", IEEE Antennas and Propagation Magazine, Vol. 44, No.1, pp.55-75, Feb. 2002

[4] L. SEVGI, *Complex Electromagnetic Problems and Numerical Simulation Approaches*, IEEE Press – John Wiley and Sons, June 2003

[5] M. Levy, *Parabolic equation methods for electromagnetic wave propagation*, IEE, Institution of Electrical Engineers, 2000

[6] F. J. Harris, "On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform", Proc. IEEE, Vol. 66, no. 1, pp.51-83, Jan 1978

[7] D. M. Etter, *Engineering Problem Solving with Matlab*, Cleve Moler, The MathWorks, Inc.

[8] R. O. Nielsen, Sonar Signal Processing, Artech House, Boston, USA, 1990

[9] L. B. Felsen, L. Sevgi, "Electromagnetic Engineering in the 21^{st} Century: Challenges and Perspectives", (introductory paper) Special issue of ELEKTRIK, Turkish J. of Electrical Engineering and Computer Sciences, Vol. 10, No.2, pp.131-145, Feb 2002

[10] L. Sevgi, "EMC and BEM Engineering Education: Physics based Modeling, Hands-on Training and Challenges", IEEE Antennas and Propagation Magazine, Vol. 45, No.2, pp.114-119, April 2003

[11] L. Sevgi, İ. C. Göknar, "How to Establish an Intelligent Balance Between Numerical and Physical Experimentation?", ICEER 2004, International Conference on Engineering Education and Research, Olomouc and Bouzov Castle, Czech Republic, June 27-30, 2004

[12] L. Sevgi, İ. C. Göknar, "An Intelligent Balance in Engineering Education", IEEE Potentials, Vol. 23, No.4, pp.40-41, Oct/Nov 2004