Introduction

In this document, we review the features, functions and user manual of the accompanying software, which has been designed and implemented fully using MATLAB 2016a. The software is divided into two main functionalities, namely, discrete convolution and spectrum analyzer. Each part is associated with a number of features and functions that are hereby discussed and explained. The software's main window (figure 1) gives the user two buttons that take him\her directly two the desired part as written on the button.

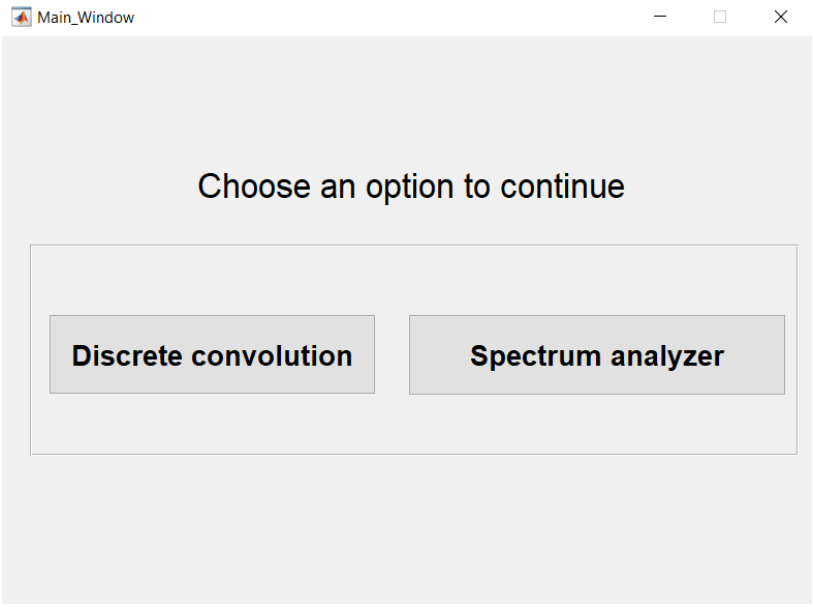


Figure 1: Software's main window

1- Discrete convolution window

Theory

Discrete convolution is a fundamental operation of the study of digital signal processing, yet the concept itself is not exclusively described for discretized systems. Convolution comes in handy mostly when the response of a certain system is of interest. Linear time invariant systems such as filters and communication channels are thereby recognized by their impulse response, which when convolved with a given input signal provides the actual output of that system. Discrete convolution of two given functions f, g can be described by the following formula:

$$(f * g)(n) = \sum f(k)\, g(n-k) \Big|_{k=-\infty:\, k=\infty}$$

This can be visualized by flipping and shifting the function g (n) from –infinity to infinity and summing the multiplication of the two function in each iteration. The processes is commutative as well, meaning that (f ∗ g) = (g ∗ f).

The software provides the ability to carry out this process between two given function of which the parameters are specified by the user, using MATLAB built in function “conv()”. After clicking the button that says “Discrete Convolution” on the main menu, the window in figure 2 opens up. The window contains 9 main components numbered 1:9. The components are:

- ‘1’ is a drop down list of the signal choices provided for the user to perform discrete convolution on. The list contains [rect pulse, ramp function, exponential function].
- ‘2’ are the input data entries in which the user specifies the signal parameters [signal length, amplitude, pulse width\exponential power, amount of circular shift] these 4 text boxes provide all the needed information that directly affect the convolution result.
- ‘3’ contains the figure in which the signal is plotted after clicking “plot” button.
- ‘4’ contains the buttons that initialize the convolution process
- ‘5’ prints the convolution result between the two signals on ‘7’ at once without animation
- ‘6’ initializes the animation process displayed on ‘8’ ‘9’ which visualizes the process with a delay of 0.3 seconds. The convolved signal is inverted and shifted on ’8’ and the corresponding convolution result is displayed on ‘9’.

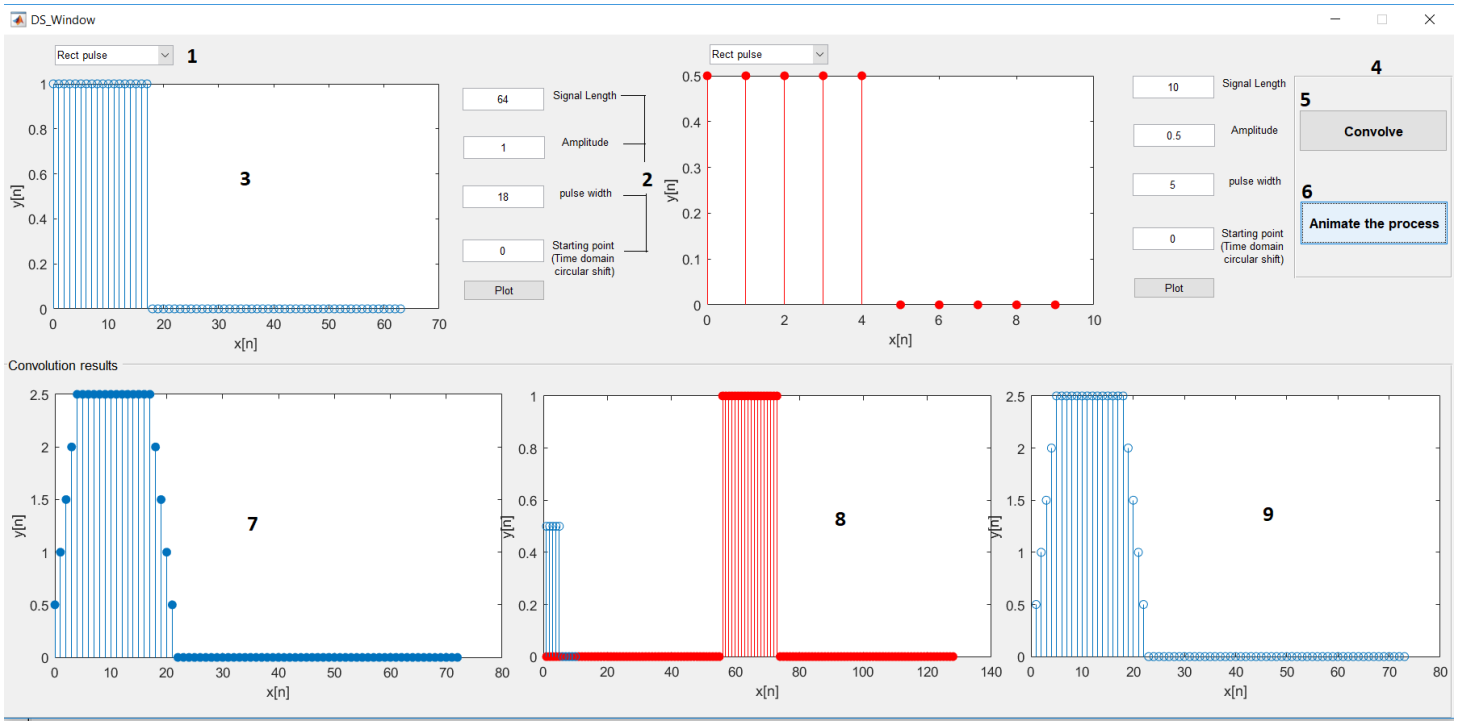


Figure 2: the discrete convolution interface window

Features:

- **Convolution between two of a set of 3 different signal types.**
- **Ability to change the signal parameters**
- **Animation of the process**

Comments:

- Circular shifting was used instead of linear shift for a better visualization and to prevent zooming out from the figure. It's still a valid process especially when dealing with DFT and periodic signals.

Handled errors:

- Convolve and animate buttons are disabled until the two signals are created, to prevent the user from causing a runtime error.
- When the "x" (closing button) in the top right corner is clicked, the main window appears, so that the user always has the option to continue using the software.

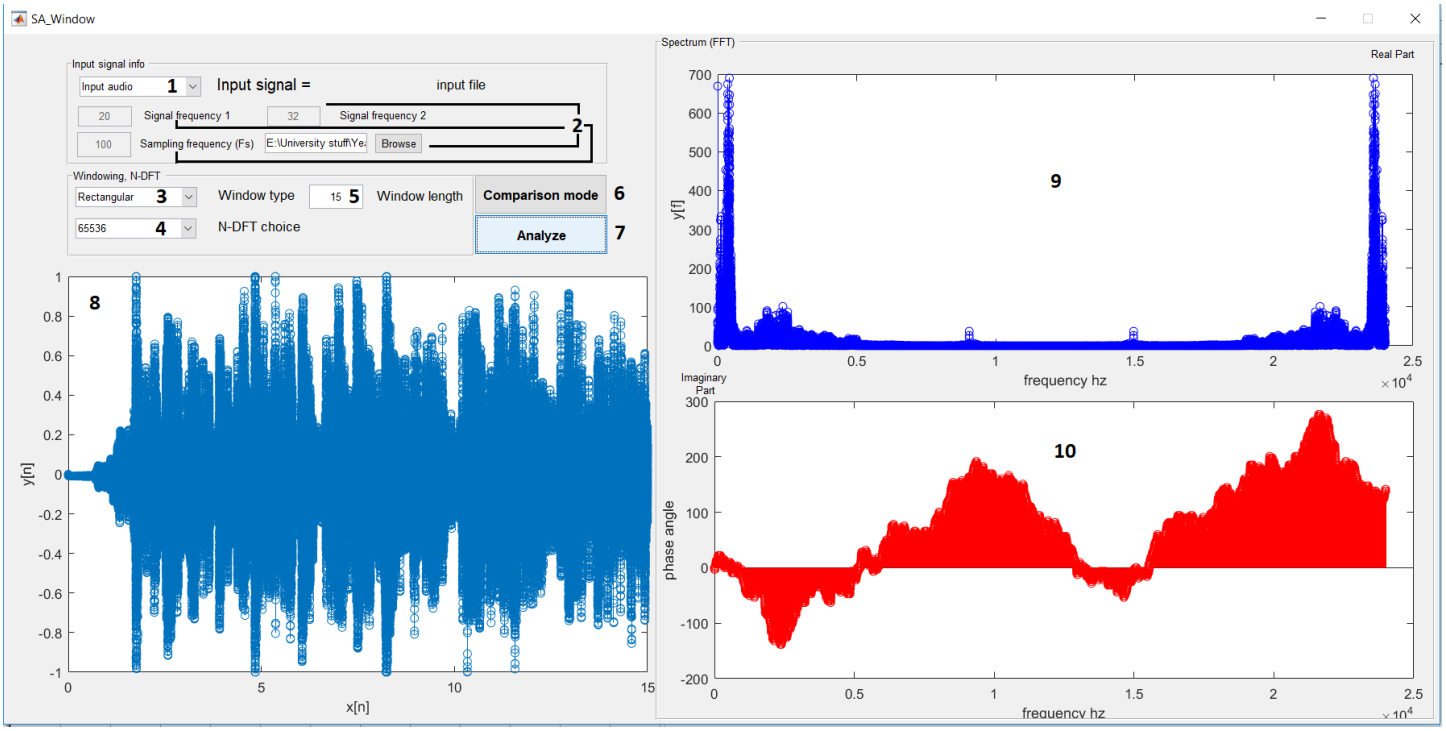
2.1- Spectrum analyzer window

Theory

Spectrum analyzers are essential devices in every signal processing facility. There are two types of spectrum analyzers, heterodyne and fft-based. In this software, we are following the algorithm and principle of Fast Fourier Transform (FFT) to obtain the frequency components of a given signal. Fourier transform itself is a mathematical formulation that aims to represent any given function on the basis of an orthogonal set of functions such as sine and cosine or complex exponentials, as of our case. Hence, we can represent any given signal by a linear combination of its composing frequency components that are easily obtained through the Fourier transform operation. However, computers and other digital systems are only capable of storing and calculating discrete quantities. Besides, the direct Discrete Time Fourier Transform (DTFT) results in a continuous range of frequencies which computers are incapable of working with. Therefore a computer must perform an extra step of discretization in the frequency domain, which is also known as the Discrete Fourier Transform (DFT). DFT itself is such a complex and time consuming process when applied on lengthy signals since its complexity is in the form of $O(n^2)$. Fortunately, a much faster algorithm was found to crack down on the complexity of DFT. FFT cores rely upon the algorithm of divide and conquer which dramatically decreases the complexity of the operation from $O(n^2)$ to $O(n*\log(n))$, which provides a much quicker alternative to DFT with no effects on the accuracy.

This software utilizes MATLAB's built-in function "fft()" as an fft core. The user has the option to either input a signal file, or experiment with 3 demo function [cosine(f1), cosine(f1)+0.5cosine(f2), and rectpulse(pulse width)], where f1, f2 are input frequencies specified by the user as well as the pulse width. The interface in figure 3 contains 10 main components. The components are as follows:

- ‘1’ is a drop down list of the signal options (3 demo functions and input file option)
- ‘2’ signal parameters and specifications [frequencies 1,2, sampling frequency and file directory (disabled when are not in use to prevent possible errors)
- ‘3’ drop down list of the window options
- ‘4’ drop down list of the possible N-DFT options
- ‘5’ Window length entry field
- ‘6’ takes the user to the comparison mode
- ‘7’ displays the time domain signal on ‘8’, real part of the fft on ‘9’ and phase angle on ‘10’



Features:

- **The ability to specify the frequencies and parameters for the demo functions**
- **The ability to choose of 4 different window types [Rectangular, Triangular, Hanning and Hamming] along with a specific window length.**
- **Different options for the N-DFT operation.**
- **The ability to change the sampling frequency (Fs) of the demo functions.**
- **The ability to browse and input any digital file formats supported by MATLAB to perform on.**
- **Visualization of the real and phase components of the frequency domain of a given signal.**
- **Comparison mode (more details are discussed in the next section).**

Comments

The time and frequency axes correspond to the actual timing and actual frequencies in Hz rather than frequency bins.

The signal mathematical representation is displayed next to the signal choice drop-down list.

Handled errors

- The signal parameters entry fields are either enabled or disabled, based on the signal choice.
- Analyze button is disabled in the case of “input file” and does not get enabled until the user browses to a file directory to prevent a run time error.

2.2- Comparison Mode

This window provides the same functionalities of the spectrum analyzer on a given signal, but under different windowing effects and N-DFT choices. It follows the same theory and is divided into pretty much the same structure as that of the previous section. See figure 4.

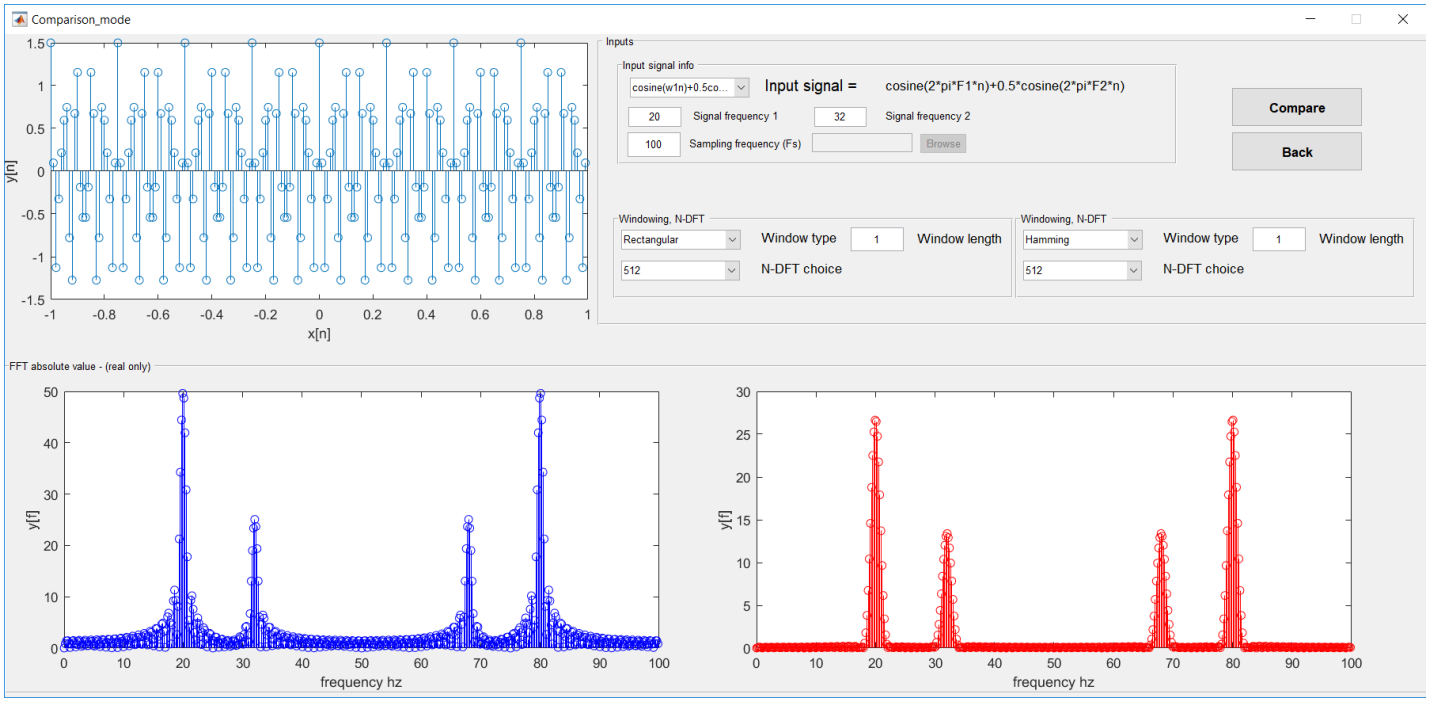


Figure 4: Comparison mode interface

Features:

The user is able to experiment with different windowing effects, N-DFT and window lengths. By pressing “compare”, the chosen signal is created and displayed in both time and frequency domains. The user can choose different window options, lengths for each signal as well as N-point DFT and the immediate results are displayed next to each other.

Handled errors

The compare button is disabled in the case of input file and is not re-enabled until the user browses to an input file to prevent a run time error.