EE 215 Semester Project

Signal Analysis using Discrete Fourier Transforms

Department of Electrical and Computer Engineering
Missouri University of Science and Technology

Abstract

The purpose of this project is to bridge the relationship between the Continuous time and the Discrete time domains. The relationship will be shown through the use of Discrete Fourier analysis. The essential idea of Fourier analysis is the use of Fourier Transforms to convert from the time domain signal to its frequency domain equivalent. In this project the Transforms to be used are the DTFT, and the DFT. Using MATLAB as the tool to compute the transforms, relationships between the transforms will be realized and ultimately a connection between the Discrete time domain and the Continuous time domain.

Introduction

Two of the most rapidly evolving and advancing areas of Electrical Engineering are the areas of signal processing and control systems. These areas both rely heavily on the ability to analyze signals and systems. While all ‘real’ physical signals and systems are functions of time, it is often helpful to consider them as functions of frequency. When analyzing a signal, the calculations will generally become more simplified in when analyzed with respect to the frequency. That said however, typically the transformation equation is more complicated and is usually computed using a computer based program.

Background and Theory

The first step in analyzing a signal with respect to frequency is transforming the time domain signal into the frequency domain. Such a transformation can be completed using several different Fourier transforms, however the transform chosen depends on properties of the time domain signal, and the desired frequency domain signal.

In this project the signal analysis was done on the computer using MATLAB. Because of the limitations of MATLAB and the computer based analysis in general, signals cannot be considered an infinitely long continuous time signal, but instead as finite amounts of data points that were inherently ‘sampled’ from the real continuous time signal. The sampled signals are called Discrete time signals. Due to the discrete time nature of the signal, the Fourier Transforms used in the analysis processes are

modified to fit such a type of signal. The transforms used in this project for these Discrete signals were the “Discrete Time Fourier Transform” (DTFT) and the “Discrete Fourier Transform” (DFT). As the names would imply, the transforms are extremely similar in definition.

The definition of the Discrete Time Fourier Transform (DTFT) is shown in Figure 1 of Appendix A. The DTFT is essentially a transform that is only realizable for discrete signals with an ‘infinite’ number of samples. When brought into the frequency domain, the DTFT result will be an inherently ‘infinite’ amount of complex discrete frequency values representing the amplitude and phase of the signal at the respected frequencies. Due the resulting amount of infinite discrete frequencies the DTFT result can be considered in continuous frequency, that is being defined at all frequencies. Also because it is physically impossible to have an infinite number of samples, the DTFT is generally regarded as a theoretical transform.

The Discrete Fourier Transform (DFT) on the other hand, is more practical in nature than the DTFT. The definition of the DFT is shown in Figure 2 of Appendix A. The essential difference between the DFT and the DTFT coincides with the fact that the DFT is defined over a finite length of time and consists of a finite amount of equally spaced values. Due to this fact, the DFT is more practical to compute on a computer than the DTFT. As one may realize, the larger the number of values, or samples, included in the DFT computation, the closer the result of the DFT becomes to the values given by the DTFT.

The computation required for the definition of the DFT is considerably complex, and lengthy. While a computer can solve this much quicker than by hand, the computation time by the computer will increase exponentially as the number of samples is increased. It is for this reason that a faster computer algorithm to compute the DFT was created. This algorithm is called the “Fast Fourier Transform”. The actual details of this algorithm are beyond the scope of this report. However it can be said that the FFT algorithm reduces computation time by removing N^2 complex multiplications and N(N-1) complex additions. These reductions in computation are noticeable even with a small amount of samples.

Considering the importance in the number of discrete samples taken from a Continuous time function to obtain an accurate waveform, a sampling criterion has been formed to provide a rule of thumb for sampling frequencies in periodic functions. The name of this criterion is the Nyquist Criterion. Simply put the Nyquist Criterion states that for a Continuous time periodic signal to be accurately represented as a Discrete time signal, the sampling frequency must be greater than two time the fundamental frequency of the signal.

Procedure

Through preforming the DTFT, DFT, and the FFT on two different continuous time functions the relationship between the discrete and continuous time domains will be validated. Two functions were tested in this project. These signals were the cosine function and the rectangle function shown in Figure 3 of Appendix A. Furthermore, as it was mentioned previously, the transforms were computed using MATLAB.

The first step was converting the Continuous time signal into the Discrete time signal. This process was simplified immensely through MATLAB. The MATLAB codes for both of the functions are shown in Appendix B. Several sampling frequencies were chosen below, above and at the Nyquist criterion for the cosine function. Because the rectangle function is not periodic, a sampling frequency of 10 was chosen and increased accordingly. The time step was quickly found to be the reciprocal of the sampling frequency. Using this, the complete time-frame and time steps of the Discrete time signal were declared and the function was computed for its respected time-frame and time step.

With the newly found discrete time version of the signal found, the next logical step was to compute the Discrete Time Fourier Transform (DTFT). Using the definition shown previously in Figure 1 of Appendix A, the summation was implemented into MATLAB as a set of two nested ‘for’ loops. From this calculation the resulting frequency

domain signal had both an amplitude and phase associated with its respected frequencies. The phase portion of the signal will be ignored throughout the remainder of the project. By computing the frequency for one period and then scaling the axis and amplitudes appropriately, the results are represented appropriately and are computed much faster.

In order to verify the results of the computed DTFT, a theoretical or ‘modified’ version of the DTFT will be used. Computing the ‘modified’ version of the DTFT involves convoluting the analytically calculated or ‘theoretical’ DTFT of the original discrete signal with the DTFT of a windowing function that represents the desired width. The result from this convolution should be nearly identical to the computed DTFT just before. This cross-referenced DTFT function will be referred to as the ‘modified’ DTFT. For this project the windowing function was chosen to be a rectangular function.

Following the completion of the DTFT, the DFT will be computed using the original definition shown in Figure 2 of Appendix A. The computation involved with the DFT is nearly identical to the DTFT. For this reason, the resulting frequency domain signal will be found by using a set of two nested ‘for’ loops. Once again the key difference between the DFT and the DTFT is that the DFT is in discrete frequency, while the DTFT is in continuous frequency. Hence the visible differences in the graphs.

The last portion of the procedure involves computing the DFT, not by using the definition, but by the Fast Fourier Transform (FFT) algorithm. The results of this

transform will be similar to the results of the DFT from the definition. An apparent difference however is the method of implementation. Due to the way the FFT algorithm is built into MATLAB, a function called ‘fftshift’ is needed in conjunction with the ‘fft’ command. By using the ‘fftshift’ command, the resulting frequency signal is shifted in such a way that the zero frequency component of the FFT array is moved to the middle. By having the zero frequency component in the center of the array, the plots are a more accurate representation of the resulting frequency signal.

Results & Analysis

The results from the procedure were, for the most part, accurate with the theoretical background. Starting with the cosine function, it became apparent that the higher the sampling frequency was above the Nyquist criterion the closer the discrete time function became to the real continuous time signal. With a lower sampling frequency, the discrete time signal was not a good representation of the Continuous time function. The results of the conversion for the cosine function into Discrete time is shown in Figure 4 of Appendix A.

Calculating the DTFT was the second step in the procedure. From the theory, the output for the cosine function should have a peak amplitude value equal to the absolute value of the maximum value of the frequency calculated for. In this case that value was 0.5. Furthermore, these peak values should occur at the fundamental frequency of the

periodic function. This value was 1250 for the cosine function. The results of this calculation are shown in Figure 5 of Appendix A. Next, the modified DTFT was calculated to confirm the previous results by using the analytical derived results convoluted with a rectangular window function. The results of the modified DTFT are shown in Figure 6 of Appendix A.

Following the results for both of the DTFT calculations, the DFT of the cosine was found in the same manner. That is, gradually increasing the sampling frequency above the Nyquist criterion until the result matched theoretical results. The results of the DFT calculations using the DFT definition are shown in Figure 7 of Appendix A. To validate these results the same calculation was preformed again, only using the FFT algorithm. Doing so confirmed that the resulting signal found from the DFT definition, was indeed correct. The results from the FFT calculation are shown in Figure 8 of Appendix A.

After the Fourier analysis of the cosine function, the rectangular function from Figure 3 of Appendix A was analyzed in a similar manner. The main difference with the rectangular function is that it is not a periodic signal. As such, the Nyquist criterion was not applicable. Therefore the sampling frequency was chosen and increased until waveform converged to a general form or appeared similar to the theoretical results. After converting the Continuous time signal into the Discrete time version, the resulting waveform is shown in Figure 9 of Appendix A.

Once the Discrete time version of the signal had been found, the next step was to calculate the DTFT by using the definition. In the same manner that was done with the cosine function, the DTFT on the rectangle function was realized and found to result in the graphs shown in Figure 10 of appendix A. In cross-referencing the results found from the DTFT the modified DTFT calculations were also computed. A similar rectangular windowing function was used in this calculation as well. The waveforms from this computation are shown in Figure 11 of Appendix A.

The DFT calculations were also found to match the results of the DTFT for the rectangular function. The results are shown in Figure 12 of Appendix A. Furthermore the FFT matched the DFT results. The FFT output is shown in Figure 13 of Appendix A

Conclusions

Following the completion of the project, it was clear that as the sampling frequency increased, the more accurate the discrete time signal and frequency transforms represented their continuous time counterpart. Furthermore it became obvious that the resulting waveforms were more accurate once the sampling frequency surpassed the Nyquist criterion. Given a sufficient amount of samples the DTFT and DFT computations will converge to the same waveform. It is this waveform that, given a sufficiently large amount of samples, will become the same waveform that would result from computations in the Continuous time domain.

Having completed the project, relationships between the Fourier transforms, and sampling frequencies have been found. It is the relationship between these two that determine the accuracy, computation time, and similarity to the continuous time domain function. It is the knowledge of this relationship in Fourier analysis that will aid in future electrical engineering endeavors such as Signal Processing and Control system involving computer based analysis.

Appendices

Appendix A (Figures)

$$\mathcal{F}\{x[n]\} = X(f) = \sum_{n=\infty}^{\infty} x[n]e^{-j2\pi fn}$$

Figure 1. Definition of the Discrete Time Fourier Transform (DTFT)

$$\mathcal{F}\{x[n]\} = X[k] = \sum_{n=0}^{N-1} x[n]e^{\frac{-j2\pi kn}{N}}$$

where N is the number of samples taken

Figure 2. Definition of the Discrete Fourier Transform (DFT)

$$x(t) = \cos(2\pi 1250t) \quad \text{and} \quad x(t) = \text{rect}(t/60)$$

Figure 3. The functions to be analyzed

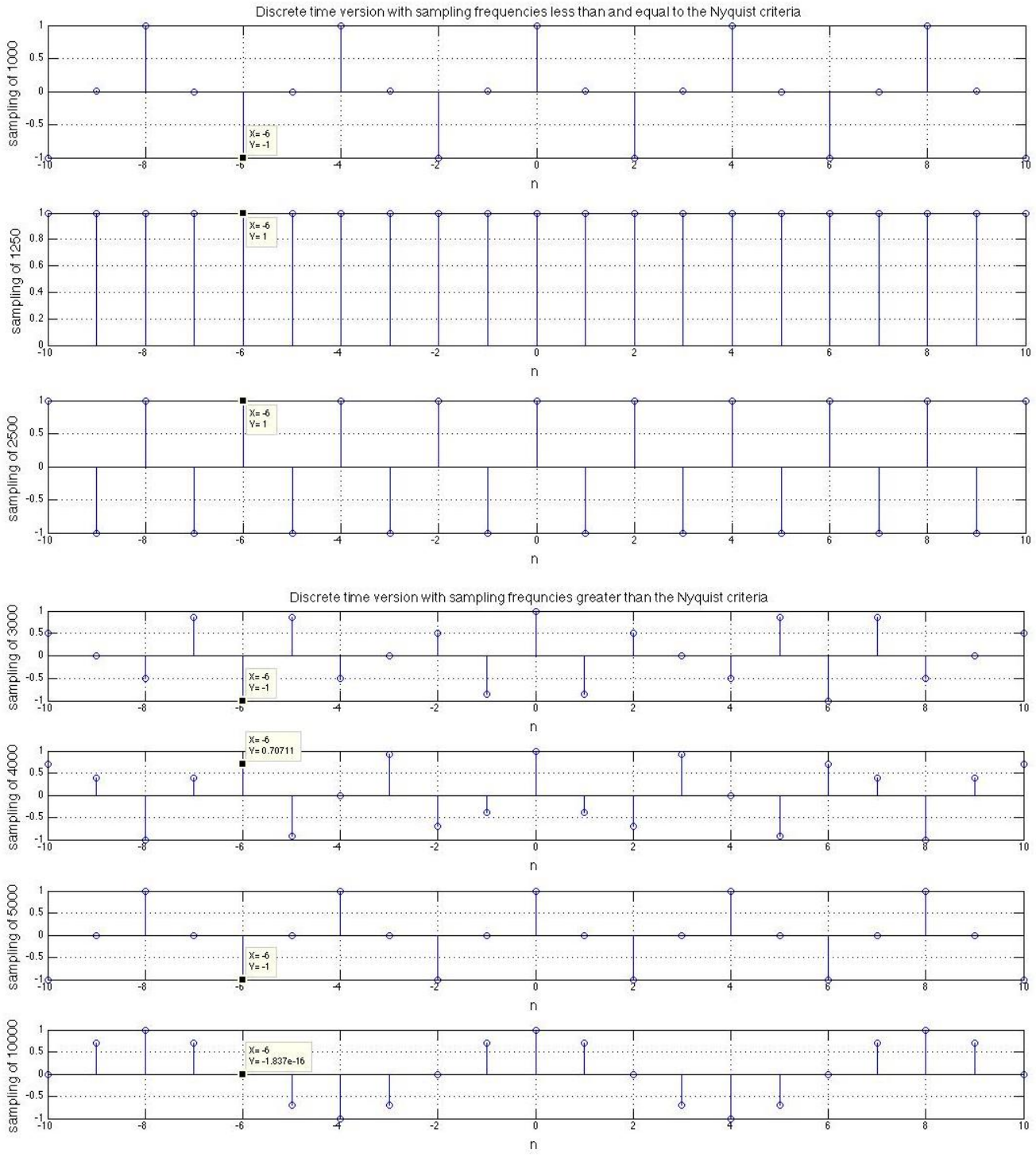

Figure 4. Discrete time version of the Cosine function

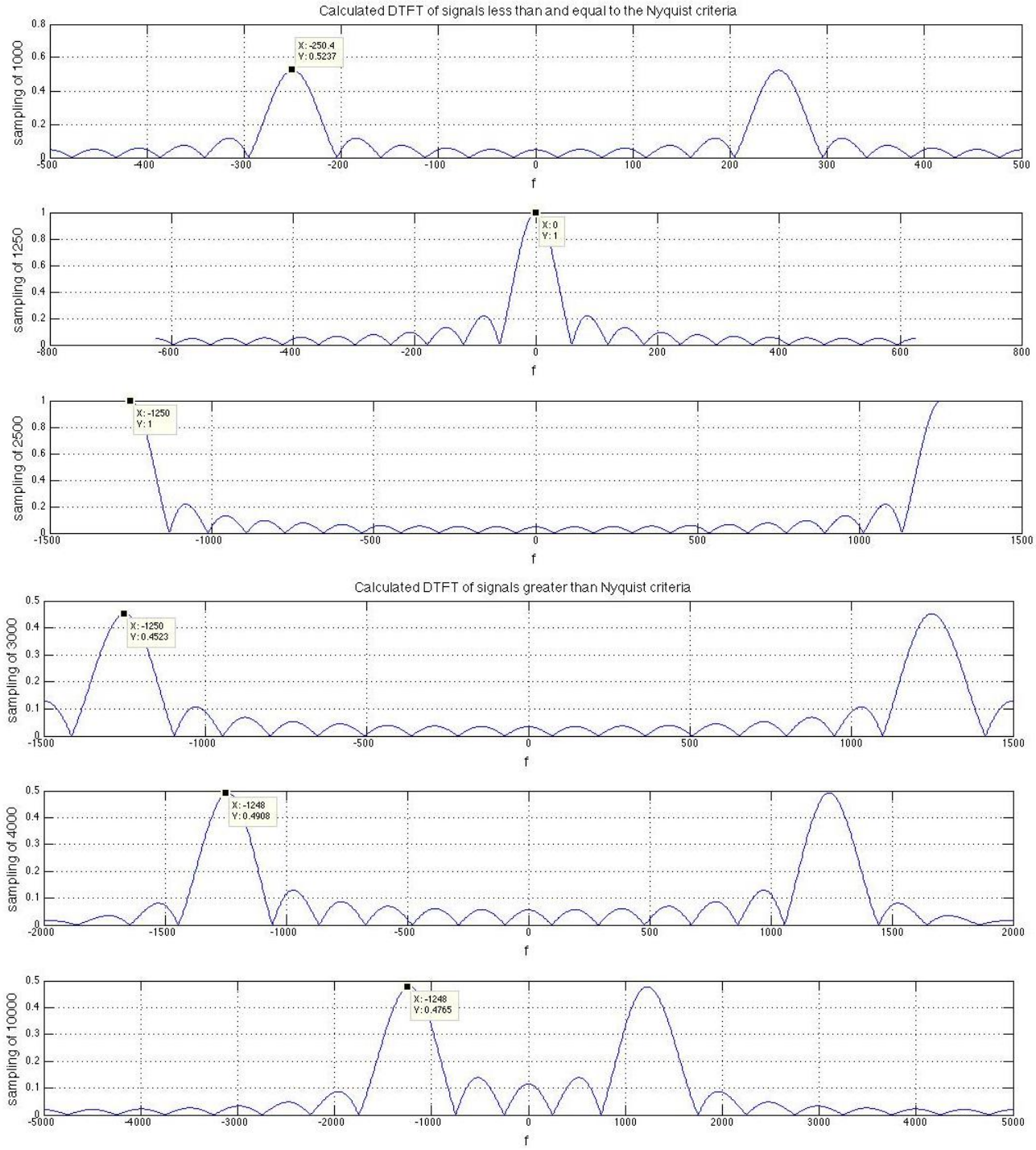

Figure 5. DTFT of the Cosine function

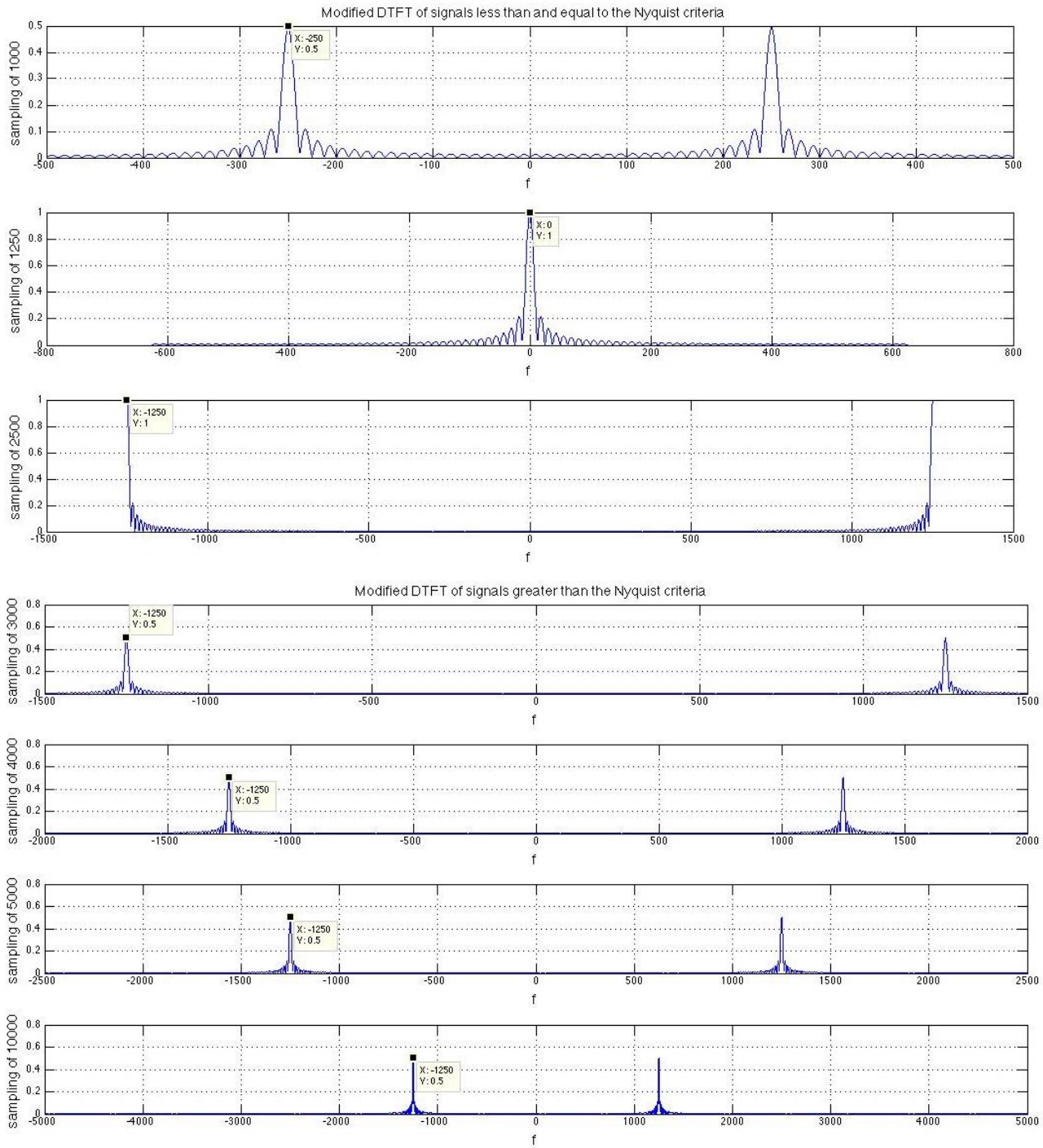

Figure 6. Modified DTFT of the Cosine function

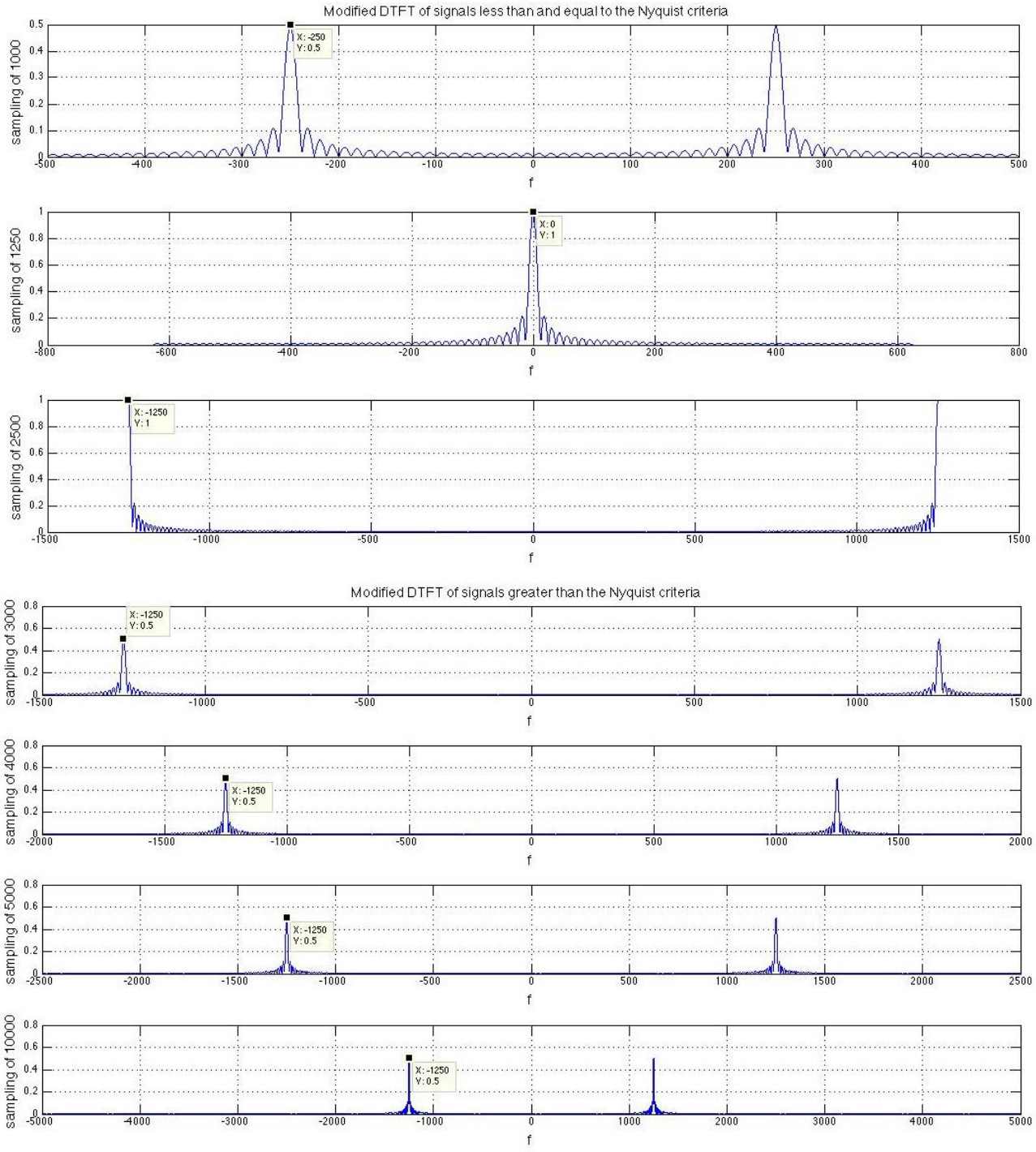

Figure 7. DFT of the Cosine function

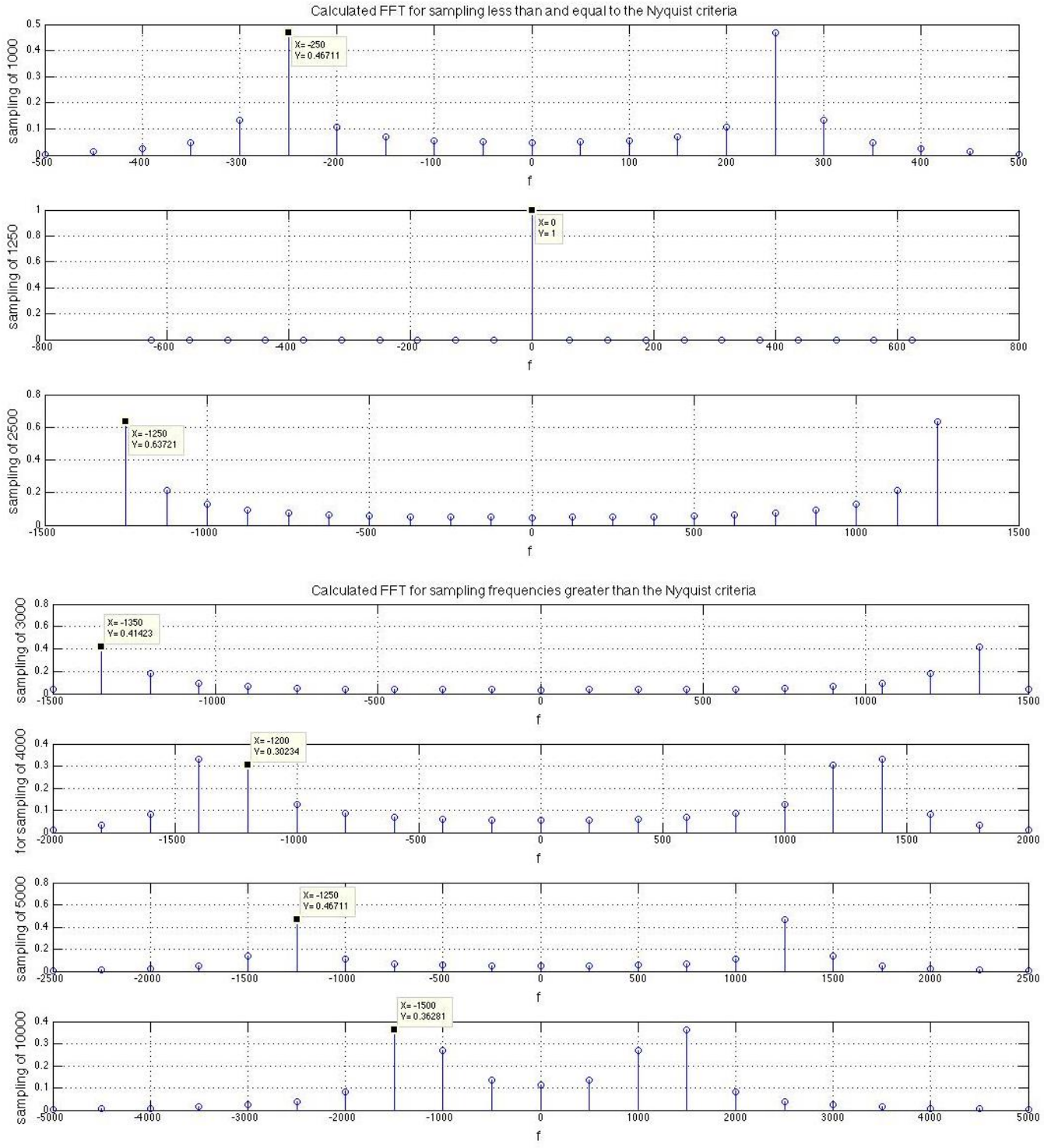

Figure 8. FFT of the Cosine function

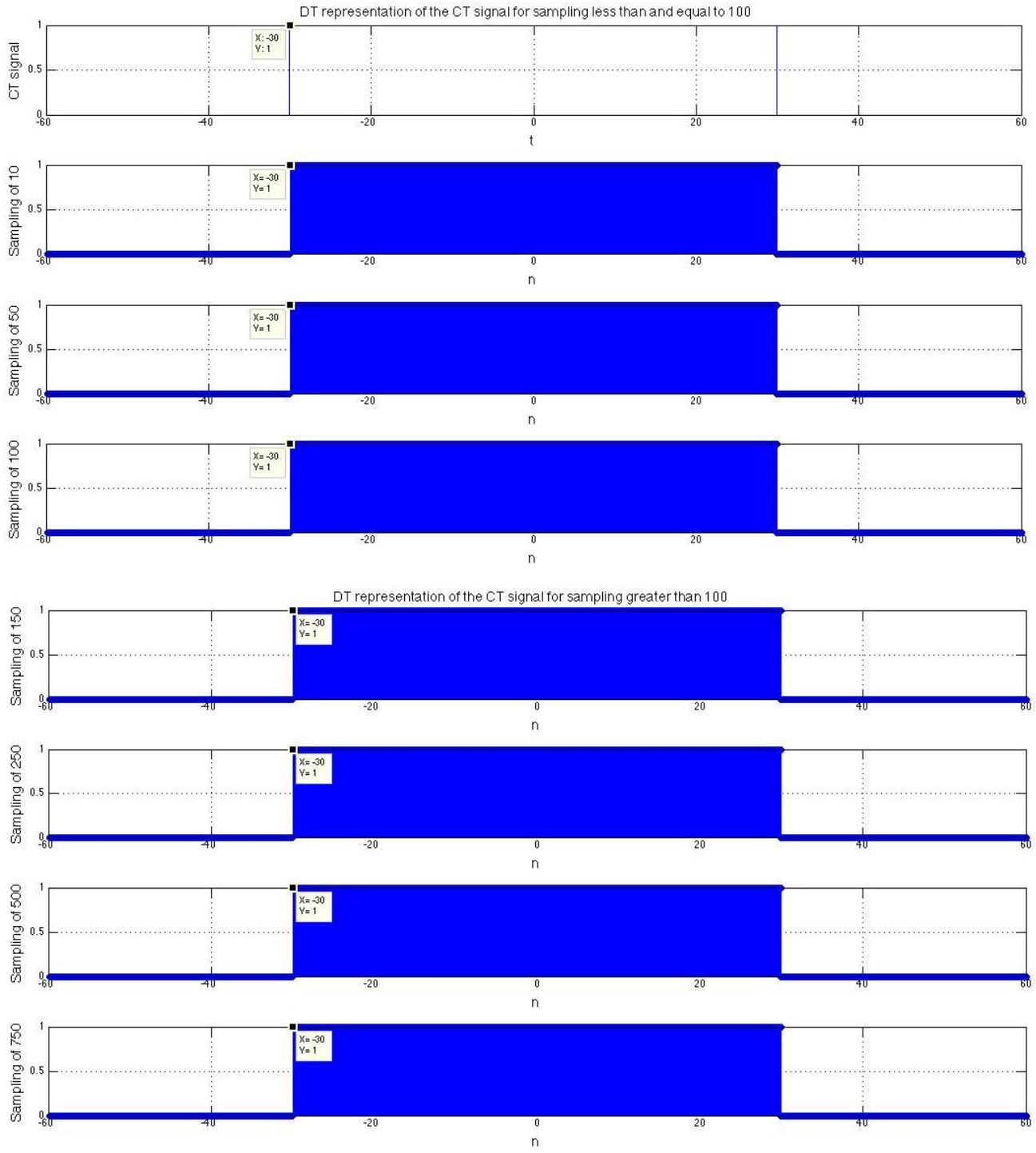

Figure 9. Discrete time version of the rectangular function

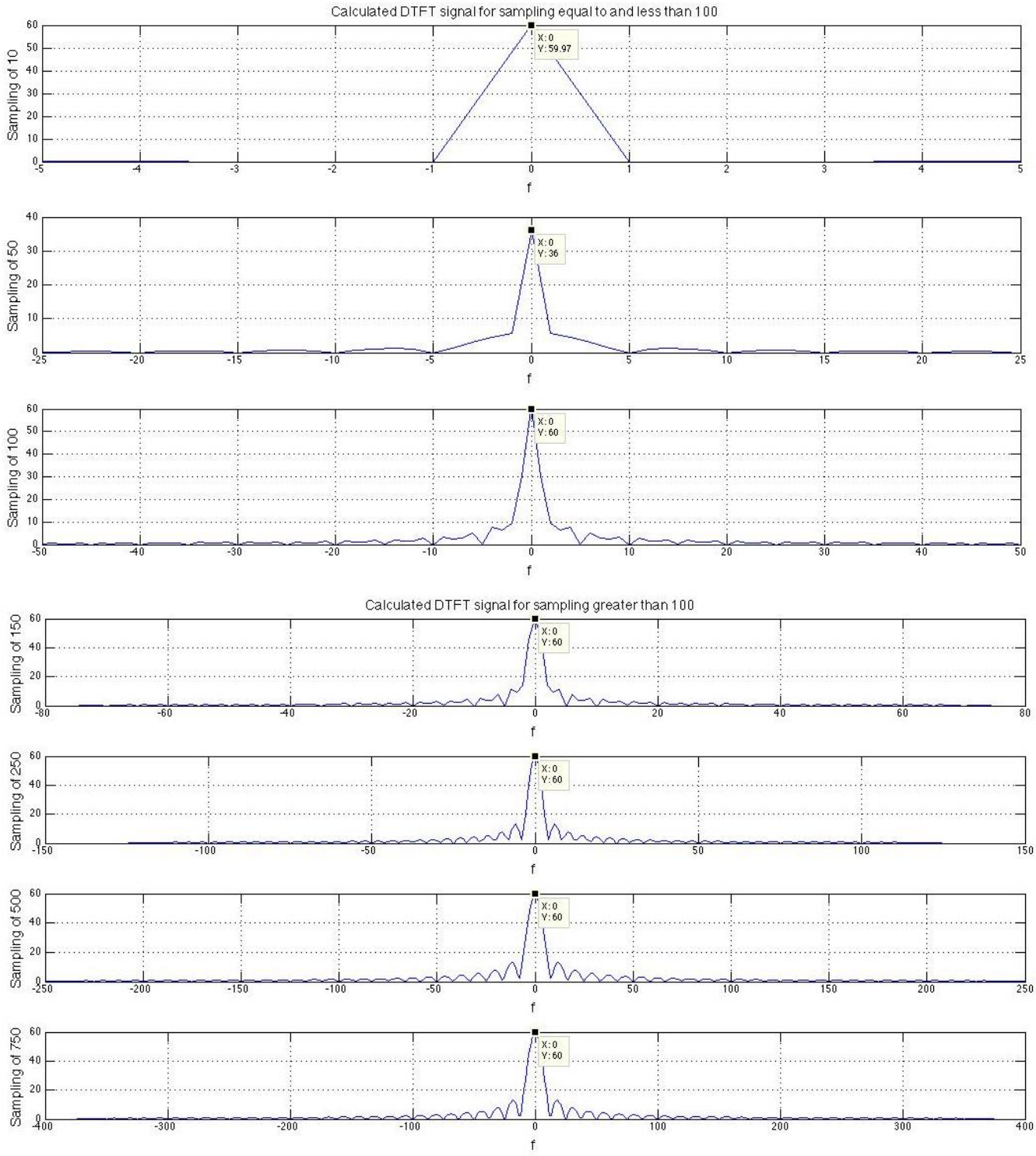

Figure 10. DTFT of the rectangular function

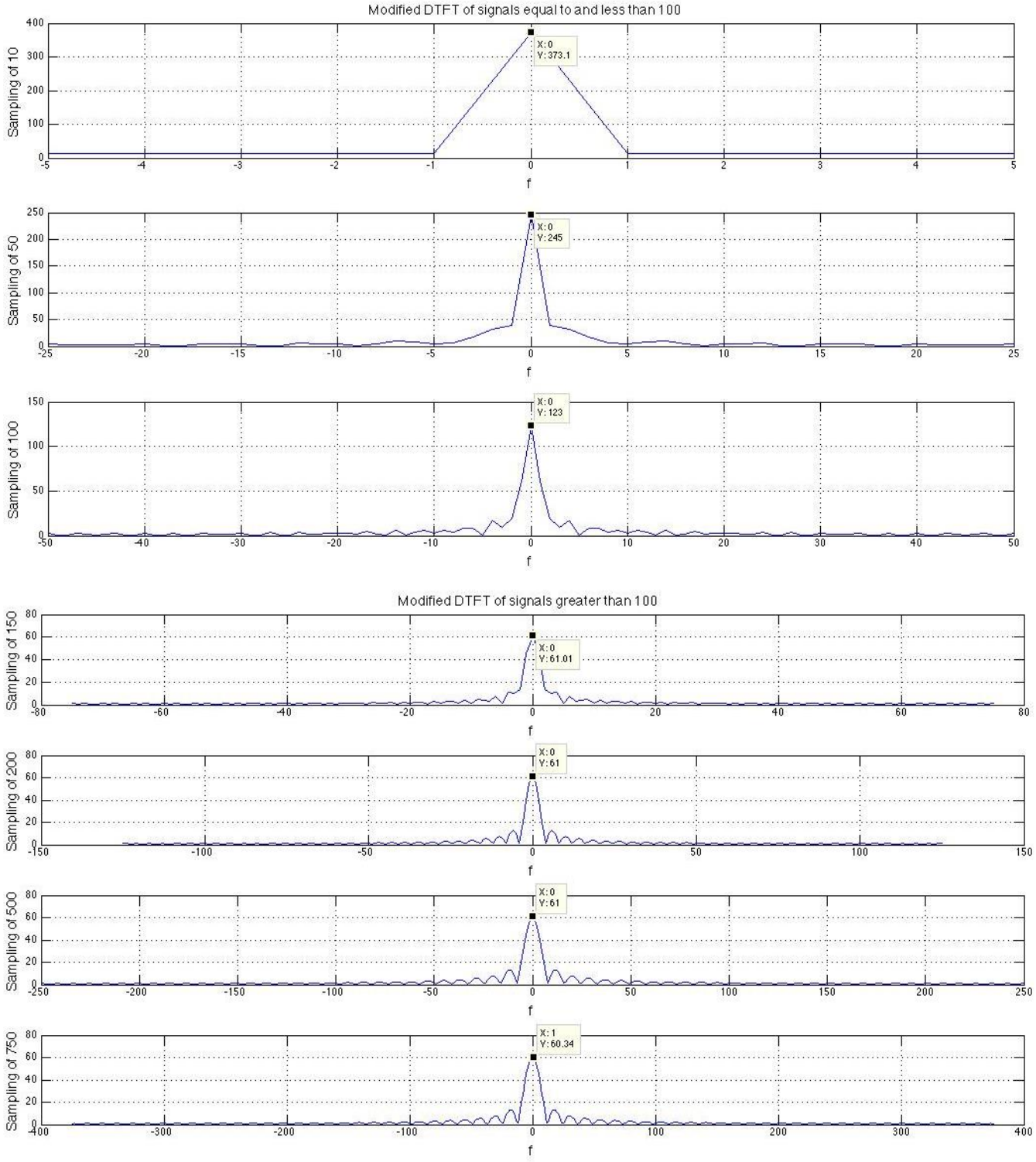

Figure 11. Modified DTFT of the rectangular function

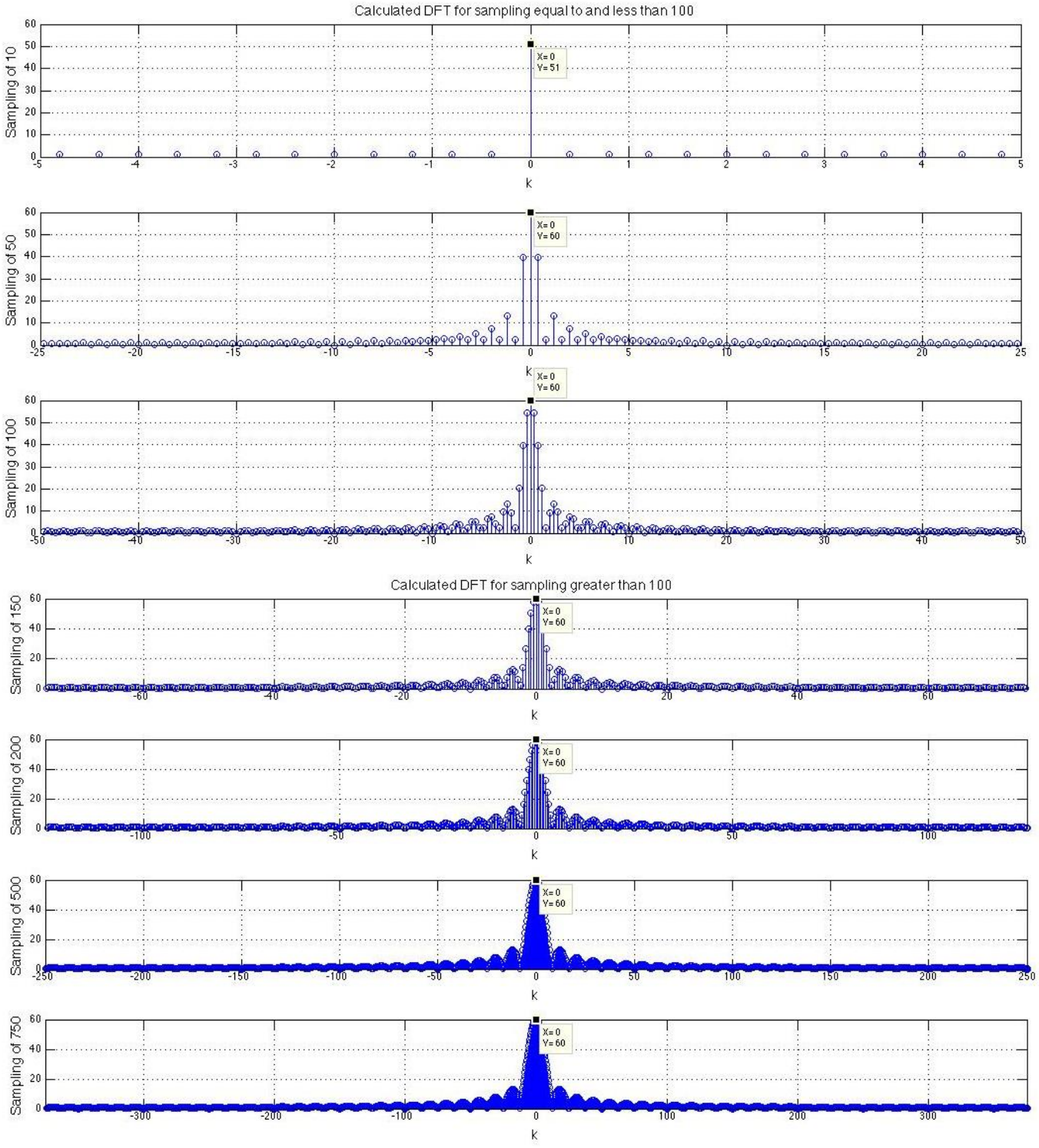

Figure 12. DFT of the rectangular function

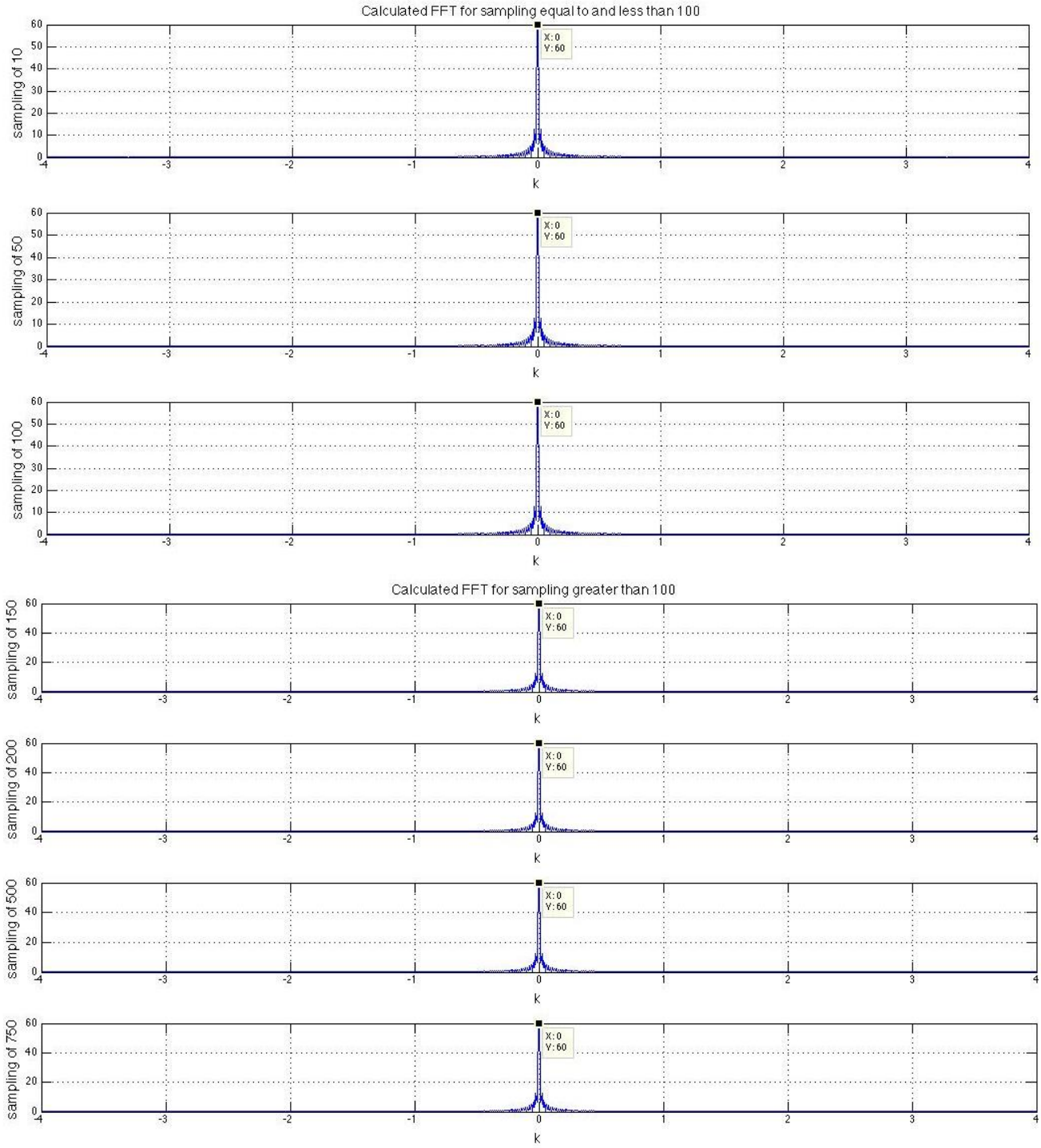

Figure 13. FFT of the rectangular function

Appendix B (MATLAB code)