Chapter 2

The Fourier Transform

2.1 Introduction

The sampled Fourier transform of a periodic, discrete-time signal is known as the *discrete Fourier transform* (DFT). The DFT of such a signal allows an interpretation of the frequency domain descriptions of the given signal.

2.2 Derivation of the DFT

The Fourier transform pair of a continuous-time signal is given by Equation 2.1 and Equation 2.2.

$$X(f) = \int_{-\infty}^{\infty} x(t) \cdot e^{-j2\pi ft} dt \tag{2.1}$$

$$x(t) = \int_{-\infty}^{\infty} X(f) \cdot e^{j2\pi ft} df \tag{2.2}$$

Now consider the signal $x(t)$ to be periodic i.e. repeating. The Fourier transform pair from above can now be represented by Equation 2.3 and Equation 2.4 respectively.

$$X(kf_0) = \frac{1}{T_0} \int_{0}^{T_0} x(t) \cdot e^{-j2\pi kf_0 t} dt \tag{2.3}$$

$$x(t) = \sum_{k=-\infty}^{\infty} X(kf_0) \cdot e^{j2\pi kf_0 t} \tag{2.4}$$

where $f_0 = \frac{1}{T_0}$. Now suppose that the signal $x(t)$ is sampled N times per period, as illustrated in Figure 2.1, with a sampling period of T seconds. The discrete signal can now be represented by Equation 2.5, where δ is the dirac delta impulse function and has a unit area of one.

$$x[nT] = \sum_{n=-\infty}^{\infty} x(t) \cdot \delta(t - nT) \tag{2.5}$$

Now replace $x(t)$ in the Fourier integral of Equation 2.3, with $x[nT]$ of Equation 2.5. The Equation can now be re-written as:

$$X(kf_0) = \frac{1}{T_0} \sum_{n=-\infty}^{\infty} \int_{0}^{T_0} x(t) \cdot \delta(t - nT) \cdot e^{-jk2\pi f_0 t} dt \tag{2.6}$$

Since the dirac delta function $\delta(t - nT) = \begin{cases} 1 & \text{for } t = nT \\ 0 & \text{otherwise} \end{cases}$, Equation 2.6 can be modified even further to become Equation 2.7 below:

$$X[kf_0] = \frac{1}{N} \sum_{n=0}^{N-1} x[nT] \cdot e^{-jk2\pi f_0 nT} \tag{2.7}$$

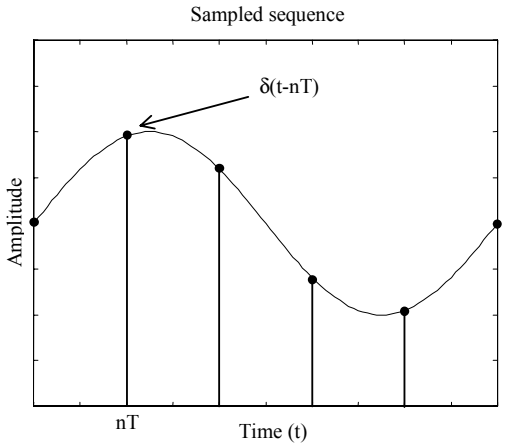


Figure 2.1: Sampled sequence for the DFT.

Since $f_0 = \frac{1}{T_0}$ and $T_0 = NT$, Equation 2.8 is formed by combining these two together.

$$f_0 T = \frac{1}{N} \tag{2.8}$$

By inserting Equation 2.8 into Equation 2.7, the DFT and its inverse, for a periodic signal, is represented by Equation 2.9 and Equation 2.10 respectively.

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-\frac{j2\pi nk}{N}} \tag{2.9}$$

$$x[n] = \frac{1}{N} \sum_{n=0}^{N-1} X[k] \cdot e^{\frac{j2\pi nk}{N}} \tag{2.10}$$

2.3 Digital Frequency

The *N* spectral values correspond to frequencies of

$$kf_0 = \frac{k}{T_0} = \frac{k}{nT} = \frac{kf_s}{N}$$

and hence to digital frequencies (i.e. normalised to the sampling frequency) of:

$$\frac{kf_s}{N} \Big/ f_s = \frac{k}{N} = F$$

Hence for an 8 point transform:

k=	0	1	2	3	4	5	6	7
f=	0	$\frac{1f_s}{8}$	$\frac{2f_s}{8}$	$\frac{3f_s}{8}$	$\frac{4f_s}{8}$	$\frac{5f_s}{8}$	$\frac{6f_s}{8}$	$\frac{7f_s}{8}$
F=	0	$\frac{1}{8}$	$\frac{2}{8}$	$\frac{3}{8}$	$\frac{4}{8}$	$\frac{5}{8}$	$\frac{6}{8}$	$\frac{7}{8}$

2.4 Matrix Interpretation of the DFT

When working with the DFT, it is quite common to make a substitution for the exponential term in Equation 2.9, such that,

$$W_N^{nk} = e^{-\frac{j2\pi nk}{N}} = \left[e^{-\frac{j2\pi}{N}} \right]^{nk} \tag{2.11}$$

Equation 2.9 can now be re-written and the DFT can be presented in a more user-friendly fashion, as illustrated by Equation 2.12.

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot W_N^{nk} \tag{2.12}$$

The term W_N^{nk} is more commonly known as a root of unity and can be represented by an argand or phasor diagram. As an example, Figure 2.2 illustrates the argand diagram for the case of an N = 8 point DFT.

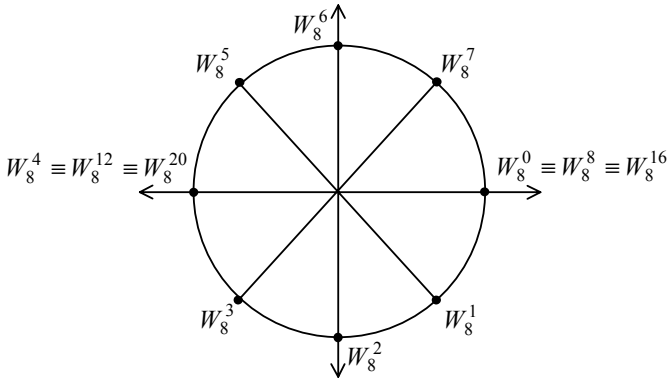


Figure 2.2: Argand diagram illustrating the 8th roots of unity.

As the DFT is a linear operation, Equation 2.12 can therefore be represented by the matrix notation defined by Equation 2.13 below. Such that the output can be derived by multiplying the corresponding Nth root of unity, from the phasor diagram, with the sampled signal $x[n]$.

$$\begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ \downarrow \\ X[N-1] \end{bmatrix} = \begin{bmatrix} W_N^0 & W_N^0 & W_N^0 \ldots\ldots\ldots\ldots W_N^0 \\ W_N^0 & W_N^1 & W_N^2 \ldots\ldots\ldots\ldots W_N^{(N-1)} \\ W_N^0 & W_N^2 & W_N^4 \ldots\ldots\ldots\ldots W_N^{2(N-1)} \\ \downarrow & \downarrow & \downarrow \ldots\ldots\ldots\ldots\ldots \downarrow \\ W_N^0 & W_N^{(N-1)} & W_N^{2(N-1)} \ldots\ldots W_N^{(N-1)^2} \end{bmatrix} \cdot \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ \downarrow \\ x[N-1] \end{bmatrix} \tag{2.13}$$

Each of the terms in the square matrix are unit vectors with a particular angle. These are shown graphically below, and it can be seen that each row consists of the samples of complex sine waves of different frequencies. The DFT output can therefore be interpreted as the result of correlating the input samples with (complex) sine waves with frequencies equal to multiples of the "fundamental" (i.e. the frequency of the second row).

$$\begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ X[3] \\ X[4] \\ X[5] \\ X[6] \\ X[7] \end{bmatrix} = \begin{bmatrix} \rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow \\ \rightarrow & \searrow & \downarrow & \swarrow & \leftarrow & \nwarrow & \uparrow & \nearrow \\ \rightarrow & \downarrow & \leftarrow & \uparrow & \rightarrow & \downarrow & \leftarrow & \uparrow \\ \rightarrow & \swarrow & \uparrow & \searrow & \leftarrow & \nearrow & \downarrow & \nwarrow \\ \rightarrow & \leftarrow & \rightarrow & \leftarrow & \rightarrow & \leftarrow & \rightarrow & \leftarrow \\ \rightarrow & \nwarrow & \downarrow & \nearrow & \leftarrow & \searrow & \uparrow & \swarrow \\ \rightarrow & \uparrow & \leftarrow & \downarrow & \rightarrow & \uparrow & \leftarrow & \downarrow \\ \rightarrow & \nearrow & \uparrow & \nwarrow & \leftarrow & \swarrow & \downarrow & \searrow \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \\ x[4] \\ x[5] \\ x[6] \\ x[7] \end{bmatrix}$$

2.5 The Fast Fourier Transform

The *fast Fourier transform* (FFT) was invented by Cooley and Tukey in 1965. They discovered that the DFT operation could be decomposed into a number of other DFTs of shorter lengths. They then showed that the total number of computations needed for the shorter DFTs was smaller than the number needed for the direct computation. In fact, the number of arithmetic operations (multiplications and additions) for the direct computation of the DFT is apaproximately equal to N^2, but for the FFT algorithm reduced to approximately $N.log_2N$. To take an example, if N=1024, the DFT would require approximately 10^6 multiplications and additions, whilst the FFT would require $<10^3$, more than 1000 times fewer.

2.5.1 Derivation of the FFT

The decomposition of the DFT is achieved by breaking a signal $x[n]$ down into two shorter, interleaved subsequences. This process is more commonly known as *decimation–in-time* (DIT). Suppose a signal exists with N sample values, where N is an integer power of 2. The signal $x[n]$ is first separated into two subsequences with *N/2* samples. One subsequence contains the samples with even-numbered values of n in $x[n]$, and the other contains those with odd-numbered values of n. Writing *n(even) = 2m* and *n(odd) = 2m*+1, the DFT from Equation 2.12 can be modified to:

$$X[k] = \sum_{m=0}^{\frac{N}{2}-1} x[2m] \cdot W_N^{2mk} + \sum_{m=0}^{\frac{N}{2}-1} x[2m+1] \cdot W_N^{(2m+1)k} \tag{2.14}$$

From the Argand diagram in Figure 2.2, it can also be shown that $W_N^2 \equiv W_{N/2}^1$ i.e. $W_8^2 \equiv W_4^1$ etc. Hence, the DFT can now be re-written to show that it can be expressed in terms of two *N/2*-point DFTs.

$$X[k] = \sum_{m=0}^{\frac{N}{2}-1} x[2m] \cdot W_{\frac{N}{2}}^{mk} + W_N^k \sum_{m=0}^{\frac{N}{2}-1} x[2m+1] \cdot W_{\frac{N}{2}}^{mk} \tag{2.15}$$

$$X[k] = X_1[k] + W_N^k \cdot X_2[k] \tag{2.16}$$

$X_1[k]$ is the transform of the even numbered points in $x[n]$, and $X_2[k]$ is the transform of the odd-numbered points in $x[n]$. It is important to note that we must multiply $X_2[k]$ by the additional term W_N^k before adding it to $X_1[k]$. This is because the sub-sequences into which we have decomposed $x[n]$ are displaced from one another in time by one sampling interval. This term W_N^k is often known as the "twiddle factor" since it is a complex number of magnitude 1 but non-zero phase, and hence merely rotates the phase of $X_2[k]$.

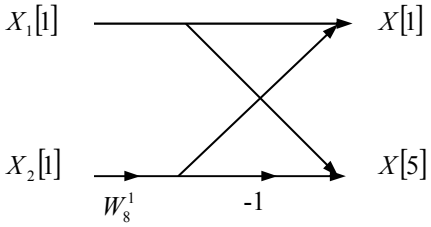
The computation in equation (2.16) is generally broken down into $N/2$ so called "butterfly operations". To illustrate this, suppose $N = 8$, and we consider by way of example the case for $k = 1$ and $k = \frac{N}{2} + 1 = 5$:

$$\begin{aligned} X[1] &= X_1[1] + W_8^1 X_2[1] \\ X[5] &= X_1[1] + W_8^5 X_2[1] = X_1[1] - W_8^1 X_2[1] \end{aligned} \tag{2.16a}$$

where X_1 is the DFT of x_0, x_2, x_4, x_6 and X_2 is the DFT of x_1, x_3, x_5, x_7. The pair of equations in (2.16a) represent the "butterfly operation" whose signal flow diagram is:

2.5.2 Radix-2 FFT

If the N length transform is an integer power of 2, then the transform can be split into two shorter N/2 subsequences. This process can continue until, in the limit, the transform is represented by a series of 2-point subsequences, each of which requires a very simple 2-point DFT. A complete decomposition of this type gives rise to the commonly used time decimated *radix-2* FFT algorithm. A decimation-in-time FFT algorithm divides up the input data into shorter interleaved subsequences. This type of FFT can be performed using many butterfly operations, as illustrated in Figure 2.3 for the case of N = 8. Here it can be seen that the operations are divided up into $\log_2 N$ sections (i.e. 3 for N=8).



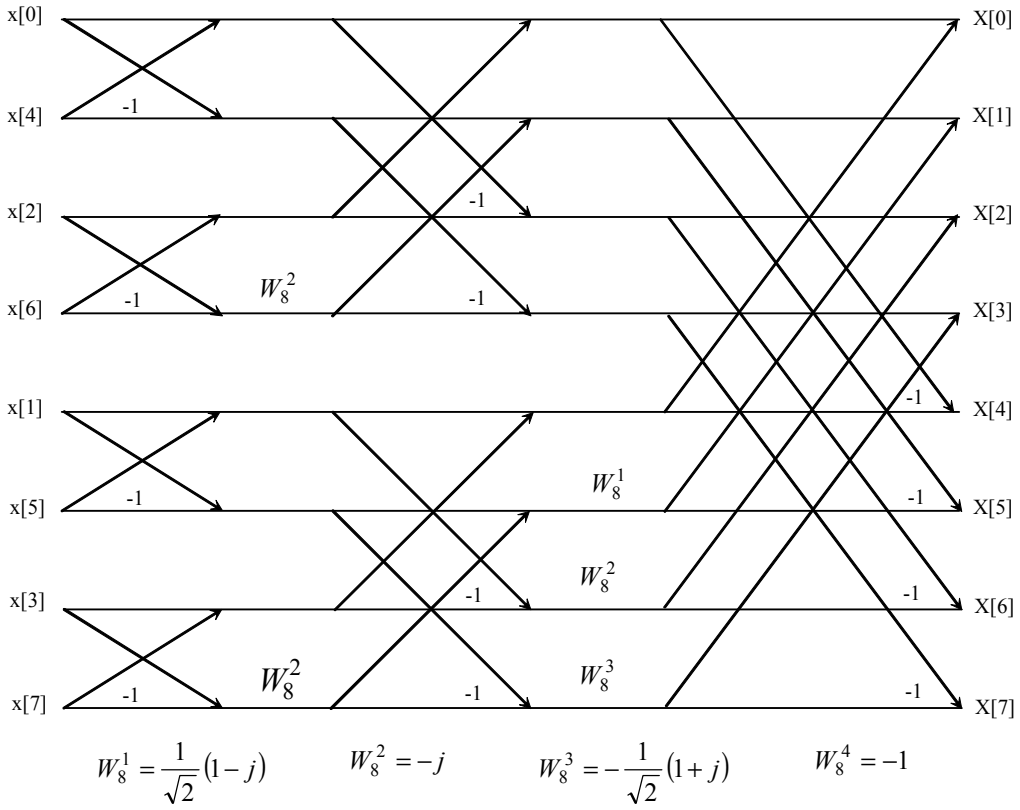


Figure 2.3: Time-decimated radix-2 FFT, N=8.

Decimation-in-frequency FFTs are in a sense the exact opposite of the decimation-in-time algorithms; they are simply the consequence of the symmetry of the Fourier transform. A decimation-in-frequency FFT, illustrated in Figure 2.4 for the case of N = 8, uses the opposite approach to the DIT. Here, the output sequence is decimated rather than the input sequence.

2.5.3 In-Place Computation

Once the output variables for each section have been calculated, there is no longer any need for the input variables. Therefore, the entire algorithm can be performed using *in-place computation*. When in-place computation is used, the output sequence overwrites the input sequence in memory, for each section of the computation. When this is done, in order for the outputs to be in the correct order, it is necessary arrange the inputs to be in *bit reversed* order. This can be achieved by expressing n in $x[n]$ in binary form, reversing the order of the bits, and using the new binary number as the position in which to store that particular sample. Hence for $x[6]$ we have n=6=110, so that the position of $x[6]$ will therefore be 011=3 (4^{th} down from the top in figure 2.3 because the index starts from 0).

One of the problems of *in place* computation is that the order of addressing the data is different for each section of the FFT. There are other methods of organising the storing of data such as the *constant geometry method* that require more memory but which have the same addressing structure for each section.

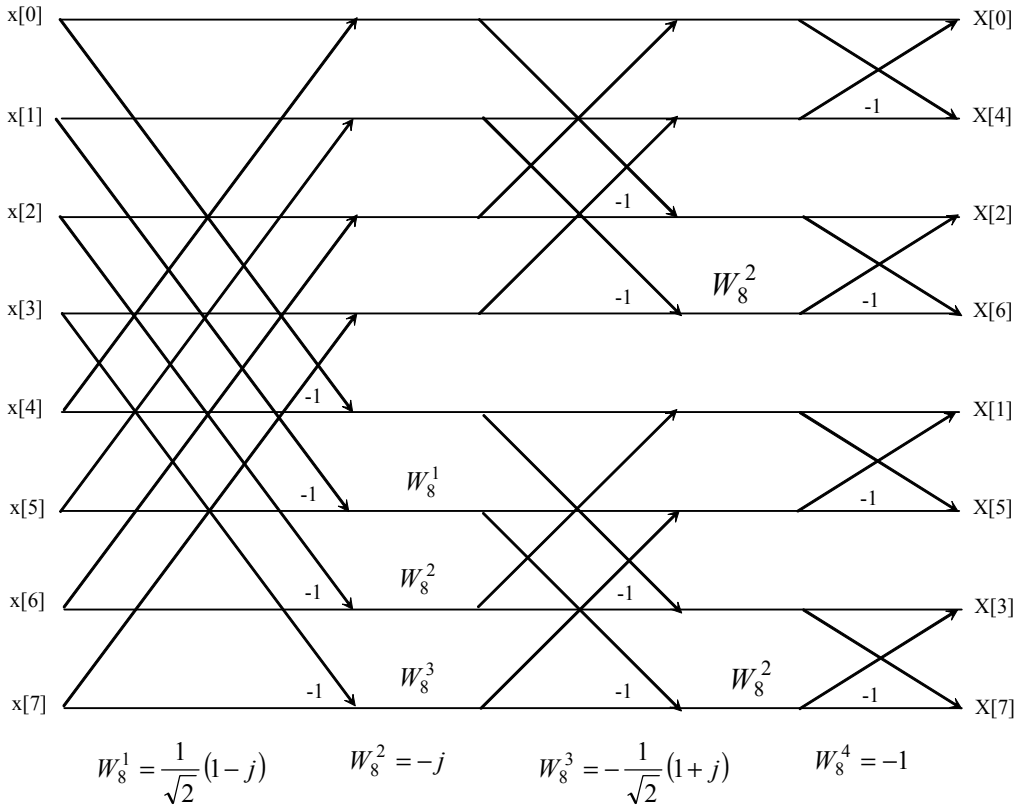


Figure 2.4: Frequency-decimated radix-2 FFT, N=8.

2.6 Windowing

When the DFT is applied to an aperiodic signal it is practical to just take a window of the sequence. A window region can be defined by effectively multiplying the signal $x(n)$ by a *rectangular window* $w(n)$, as shown in Figure 2.5 below.

The windowed function of the signal $x_N(n)$ can be mathematically defined by Equation 2.17 below.

$$x_N(n) = x(n) \cdot w(n) \tag{2.17}$$

The rectangular window function is defined by the following parameters:

$$w(n) = \begin{cases} \mathbf{1} & n_1 \le n \le n_2 \\ \mathbf{0} & \text{otherwise} \end{cases}$$

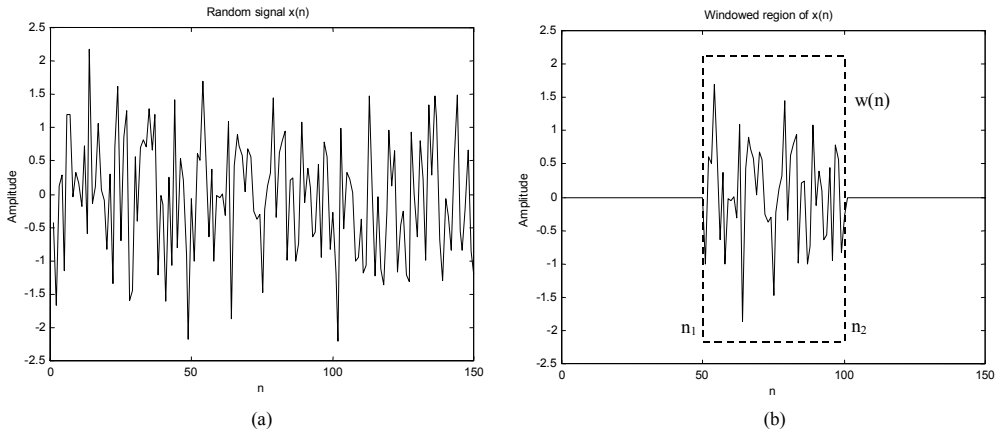


Figure 2.5: (a) Random signal; (b) a rectangular windowed region of x(n).

In DSP theory, multiplication in the time domain of two signals is equivalent to convolution in the frequency domain. Hence, Equation 2.17 can also be expressed by Equation 2.18, where $W(\omega)$ is the frequency spectrum of the window function and $X(\omega)$ is the Fourier transform of the signal.

$$X_N(w) = X(w) * W(w) \tag{2.18}$$

The time and frequency domain representation of a rectangular window function is illustrated in Figure 2.6 (a) and (b) respectively. With reference to Figure (b), the Fourier transform of a rectangular window is the well known *sinc function*.

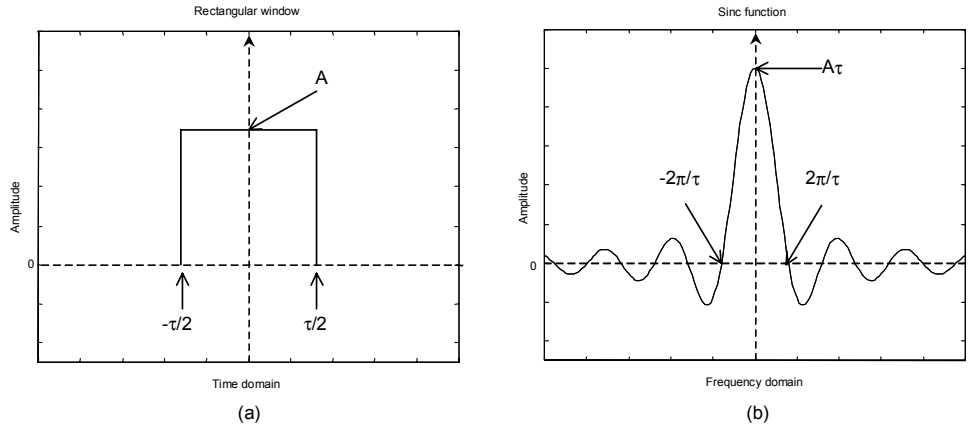


Figure 2.6: A rectangular window function; (a) Time domain. (b) Frequency domain.

2.7 Spectral Leakage

Spectral leakage is generally present when dealing with practical signals, and may lead to problems of interpretation. When the only frequency components present are an integer multiple of the first harmonic of the DFT, then all of the leakage components fall at the nulls of the *sinc* function. However, when at least one of the frequency components falls midway between two bins, then spectral leakage occurs. It results in a smaller peak response, plus a whole series of undesirable side lobe responses corresponding to the sidelobe peaks in the spectrum of the rectangular window.

To reduce the spectral leakage it is common practice to use a different window function from the rectangular window – one that has a more suitable spectrum with lower side lobes. The ideal window function for $W(w)$ is of course a delta function, since the convolution operation will then not distort $X(w)$ at all. However, the inverse Fourier transform of a delta function is $w(n) = 1$, which is of infinite duration Choosing a suitable window always involves some kind of a trade-off between the width of the main lobe and the level of the side lobes. In practice it is desirable to have a narrow main lobe and a low side lobe level. It is also important to realise that the two cannot be achieved simultaneously and a practical trade-off between the two must be tolerated. In addition to the rectangular window there are also other window functions, such as those in the table below. These window functions are covered in more detail in Chapter 4, and their characteristic shapes are illustrated in Figure 4.5.

Typical examples of window functions and their specification are as follows:

Name of window function $w[n]$	Width of main lobe (bin)	Side-lobe level in (dB)
Rectangular	0.9	-13
Bartlett	3.0	-27
Hanning	3.1	-32
Hamming	3.3	-43
Blackman	5.5	-57
Kaiser β=4	2.7	-30
β=8	5.0	-58
β=12	7.5	-90

Name of window function $w(n)$	Mathematical definition
Rectangular	1
Hanning	$0.5-0.5\cos\left[\frac{2\pi n}{N-1}\right]$
Hamming	$0.54-0.46\cos\left[\frac{2\pi n}{N-1}\right]$
Blackman	$0.42-0.5\cos\left[\frac{2\pi n}{N-1}\right]+0.08\cos\left[\frac{2\pi n}{N-1}\right]$
Kaiser	$\frac{I_0\left[\beta\sqrt{1-\left(\frac{\lvert 2n-N+1\rvert}{N-1}\right)^2}\right]}{-I_o(\beta)}$ Where, $I_0(x)=\sum_{k=0}^{\infty}\left(\frac{x^k}{2^k k!}\right)^2$

Example:

Calculate the spectral leakage of $x(t)$, with a rectangular window truncated to N = 512 samples at a sampling frequency f_{sam} = 2.56k Hz. Given that:

$$x(t)=1\cdot\sin(317.5\times 2\pi t)+0.1\cdot\sin(330\times 2\pi t)$$

The frequency normalised to the bin width for each component is given by $\frac{f\cdot N}{f_{sam}}$

Hence, for the frequency component f_1, $b_1=\frac{317.5\times 512}{2.56\times 10^3}=63.5$

And for the frequency component f_2, $b_2=\frac{330\times 512}{2.56\times 10^3}=66$

The frequency spectrum of a rectangular window $W(w)$ is given by the sinc function.

$$W(w)=\frac{\sin(k\pi)}{k\pi}$$

Where the normalised bin width $k = (b_2 - b_1)$. Hence the contribution of spectral leakage for the 317.5 Hz signal into bin 66 is given by:

$$\frac{\sin(66-63.5)\pi}{(66-63.5)\pi}=0.127$$

The frequency spectrum of the resulting leakage is illustrated in Figure 2.7 below.

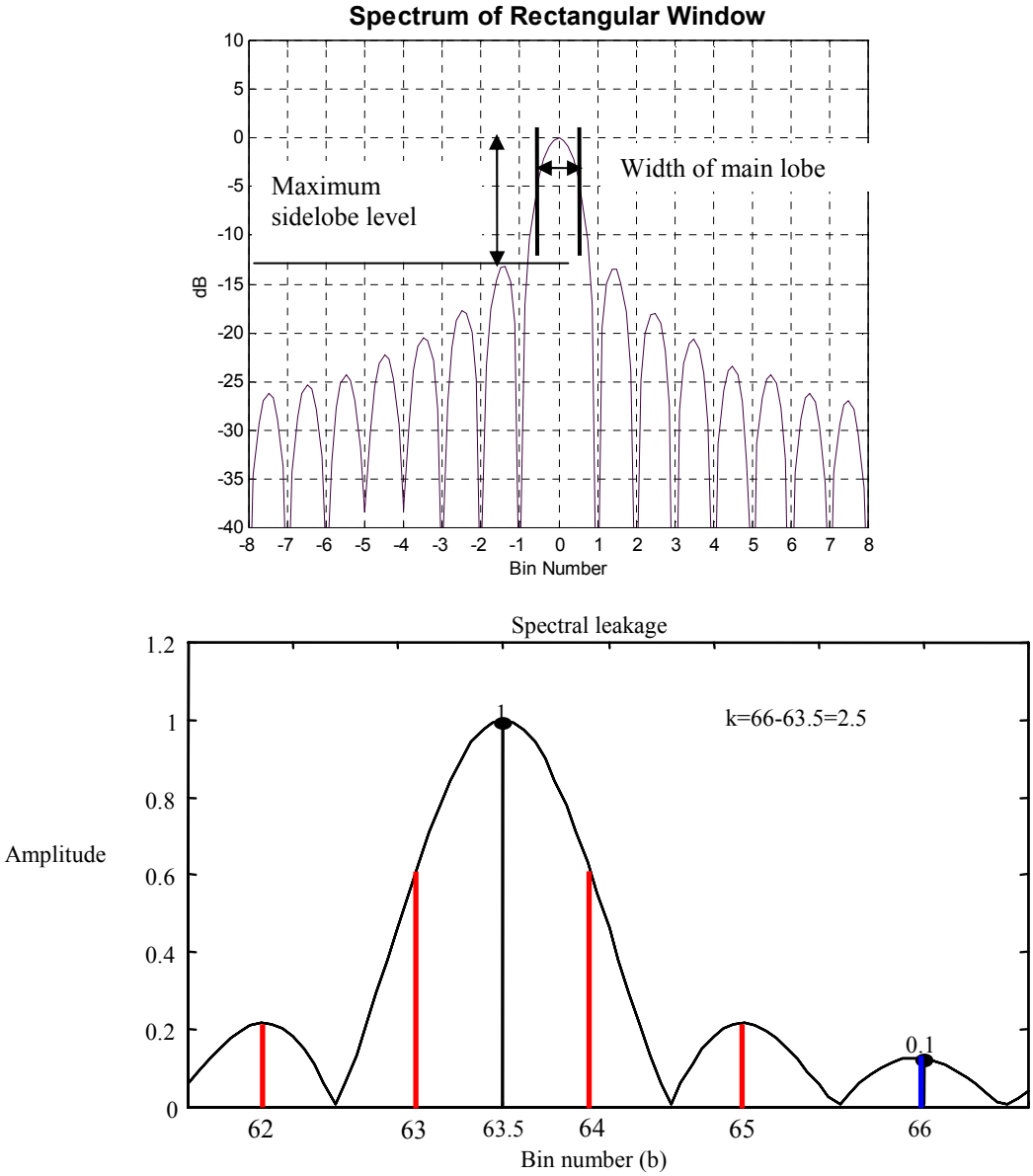


Figure 2.7: Spectral leakage of the 317.5 Hz signal into bin 66.

2.8 Frequency Domain Interpolation Using Zero Padding

If we want to interpolate the DFT output between the frequencies corresponding to the "bins", then we need to evaluate the DFT for more values of Ω. One way of achieving this, but still using the standard DFT or FFT is to add zeros to the end of a data sequence $x[n]$, and just apply the DFT. If we append (M-N) zeros to end of data sequence $x[n]$ to get $xa[n]$, and compute the M-length DFT, then this results in an output that is equivalent to computing the frequency content at M equally spaced frequency points where $M>N$. This is shown below.

The normal DFT is:

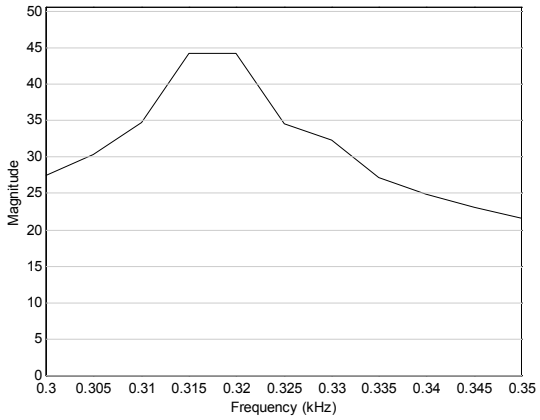
$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot W_N^{nk} \tag{2.19}$$

and the zero padded DFT is:

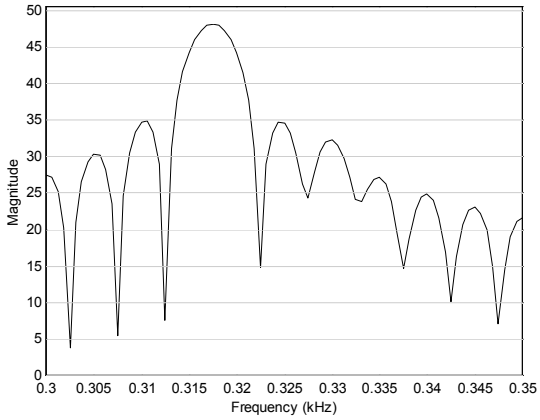
$$X_a[k] = \sum_{n=0}^{M-1} x_a[n] \cdot W_M^{nk} = \sum_{n=0}^{N-1} x_a[n] \cdot W_M^{nk} \tag{2.20}$$

where the second summation has been derived by realising that $x_a[n] = 0$ for $N \le n \le M-1$. In equation (2.20), we are now evaluating the DFT at a frequency spacing of f_s/M rather than at f_s/N for the original sequence.

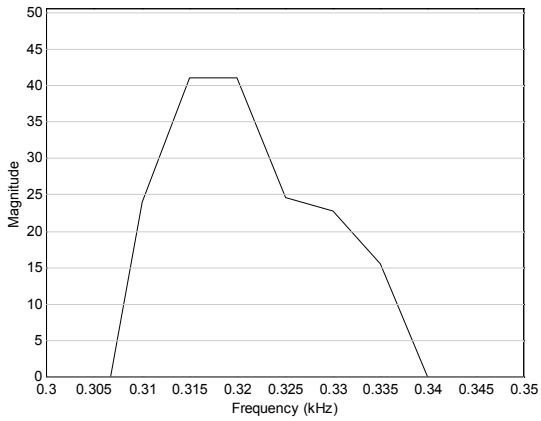
This process can be useful if we are trying to determine a signal component that lies between two bins, and is only a little larger than the underlying noise or spectral leakage floor – illustrated below with the example in section 2.7.



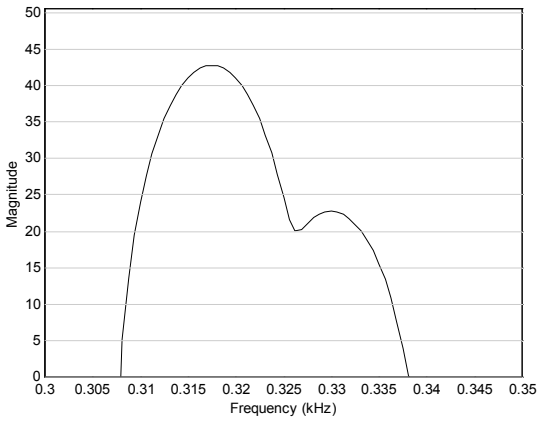
(a) Rectangular window, N=M=512



(b) Rectangular window, N=512, M=4096



(a) Hamming window, N=M=512



(b) Hamming window, N=512, M=4096

Figure 2.8: Simulink output for example in section 2.7.