Chapter 6

Practical Spectral Analysis

We illustrate the topic of this chapter by an example from musical signal processing. Suppose we are given a recording of, say, the fourth Symphony of Brahms, as a digitized waveform. We want to perform spectral analysis of the audio signal. Why would we want to do that? For the sake of the story, assume we want to use the signal for reconstructing the full score of the music, note by note, instrument by instrument. We hasten to say, lest you develop misconceptions about the power of digital signal processing, that such a task is still beyond our ability (in 1996 at least). However, the future may well prove such tasks possible.

Let us do a few preliminary calculations. The music is over 40 minutes long, or about 2500 seconds. Compact-disc recordings are sampled at 44.1 kHz and are in stereo. However, to simplify matters, assume we combine the two channels into a single monophonic signal by summing them at each time point. Our discrete-time signal then has a number of samples N on the order of 10^8. So, are we to compute the DFT of a sequence one hundred million samples long? This appears to be both unrealistic and useless. It is unrealistic because speed and storage requirements for a hundred million point FFT are too high by today's standards. It is useless because all we will get as a result is a wide-band spectrum, covering the range from about 20 Hz to about 20 kHz, and including all notes of all instruments, with their harmonics, throughout the symphony. The percussion instruments, which are naturally wide band and noiselike, will contribute to the unintelligible shape of the spectrum. True, the frequency resolution will be excellent—on the order of 0.4×10^{-3} Hz—but there is no apparent use for such resolution. The human ear is far from being able to perceive frequency differences that are fractions of millihertz, and distances between adjacent notes in music scales are three to six orders of magnitude higher.

If not a full-length DFT, then what? A bit of reflection will tell us that what we really want is *a sequence of short DFTs.* Each DFT will exhibit the spectrum of the signal during a relatively short interval. Thus, for example, if the violins play the note E during a certain time interval, the spectrum should exhibit energies at the frequency of the note E (329.6 Hz) and at the characteristic harmonics of the violin. In general, if the intervals are short enough, we may be able to track the note-to-note changes in the music. If the frequency resolution is good enough, we may be able to identify the musical instrument(s) playing at each time interval. This is the essence of spectral analysis. The human ear-brain is certainly an excellent spectrum analyzer. A trained musician can identify individual notes and instruments in very complex musical compositions.

Continuing our example, each DFT would have a length of, say, 4096, corresponding to a time interval of about 92.9 milliseconds. The frequency resolution will be about 11 Hz, which is quite adequate. There are about 26,000 distinct intervals in the symphony [$= (40 \times 60)/(92.9 \times 10^{-3})$], so we will need to compute 26,000 FFTs of length 4096. We may choose to be conservative and overlap the intervals to a certain extent (say 50 percent), to ensure that nothing important that may occur at the border between adjacent intervals is missed. Then we will need about 52,000 FFTs of length 4096. With today's technology, the entire computation can be done in considerably less time than the music itself.

We complete our Brahms example by showing what some of the music looks like in the time and frequency domains. The fourth movement of the symphony starts with eight chords, each about 1.7 seconds. Table 6.1 gives the frequencies of the individual notes in each chord.[1] Figures 6.1 and 6.2 show eight segments of the signal, one for each chord, of length 92.9 milliseconds. Figures 6.3 and 6.4 show the magnitudes of the Fourier transforms of the eight segments, in the range 0 to 2000 Hz. We also mark, on each spectrum, the frequencies of the notes of the chord, taken from Table 6.1, as well as all their harmonics up to 2000 Hz. As we see, the note frequencies and their harmonics fall on peaks of the corresponding spectra in most cases.

Bar No.	1	2	3	4	5	6	7	8
note 1	130.8	110.0	82.4	130.8	92.5	98.0	87.3	82.4
note 2	261.6	220.0	164.8	261.6	164.8	164.8	123.5	164.8
note 3	329.6	261.6	246.9	329.6	185.0	196.0	174.6	207.6
note 4	440.0	370.0	329.6	440.0	277.2	329.6	220.0	246.9
note 5	523.2	440.0	392.0	523.2	329.6	493.9	246.9	329.6
note 6	659.2	523.2	493.9	659.2	466.2	987.8	311.1	415.2
note 7	—	740.0	659.2	880.0	554.4	—	440.0	493.9
note 8	—	—	784.0	—	932.4	—	493.9	659.2

Table 6.1 Frequencies (in Hz) of the notes in the first 8 bars of the fourth movement of Brahms's Symphony No. 4.

Dividing a signal having long duration into short segments and performing spectral analysis on each segment is known as *short-time spectral analysis*. Practical short-time spectral analysis requires more than just DFT (or FFT) computations. In this chapter we study the principal technique used for short-time spectral analysis, that of *windowing*. We first describe the basic windowing operation and interpret it in the time and frequency domains. We then list the most common windows and examine their properties. Finally, we apply the windowing technique to the problem of measuring the frequencies of sinusoidal signals, first without and then with additive noise. Short-time spectral analysis is by no means limited to sinusoidal signals, however. In Section 13.1 we shall study the use of short-time spectral analysis to random signals; see page 513.

6.1 The Effect of Rectangular Windowing

Assume we are given a long, possibly infinite-duration signal $y[n]$. We pick a relatively short segment of $y[n]$, say

$$x[n] = \begin{cases} y[n], & 0 \le n \le N-1, \\ 0, & \text{otherwise.} \end{cases} \tag{6.1}$$

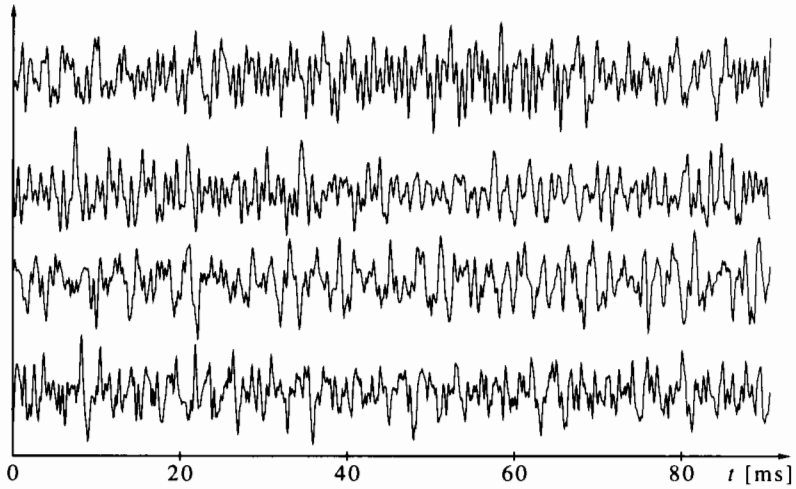

Figure 6.1 Brahms's Symphony No. 4, fourth movement, bars 1 through 4 (top to bottom), a 92.9-millisecond segment of each bar.

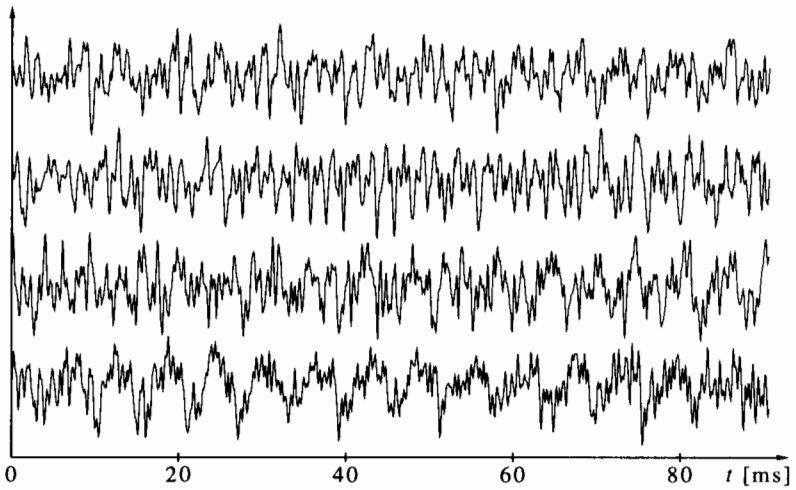

Figure 6.2 Brahms's Symphony No. 4, fourth movement, bars 5 through 8 (top to bottom), a 92.9-millisecond segment of each bar.

We can describe the operation of getting $x[n]$ from $y[n]$ as a multiplication of $y[n]$ by a *rectangular window*, that is,

$$x[n] = y[n]w_{\mathrm{r}}[n], \quad \text{where } w_{\mathrm{r}}[n] = \begin{cases} 1, & 0 \le n \le N-1, \\ 0, & \text{otherwise.} \end{cases} \tag{6.2}$$

How is the Fourier transform of a rectangular-windowed signal related to that of the original signal? Before we give a general answer, let us look at an example. Consider an exponential signal and its Fourier transform,

$$y[n] = \begin{cases} a^n, & n \ge 0, \\ 0, & n < 0, \end{cases} \qquad Y^{\mathrm{f}}(\theta) = \frac{1}{1 - ae^{-j\theta}}, \tag{6.3}$$

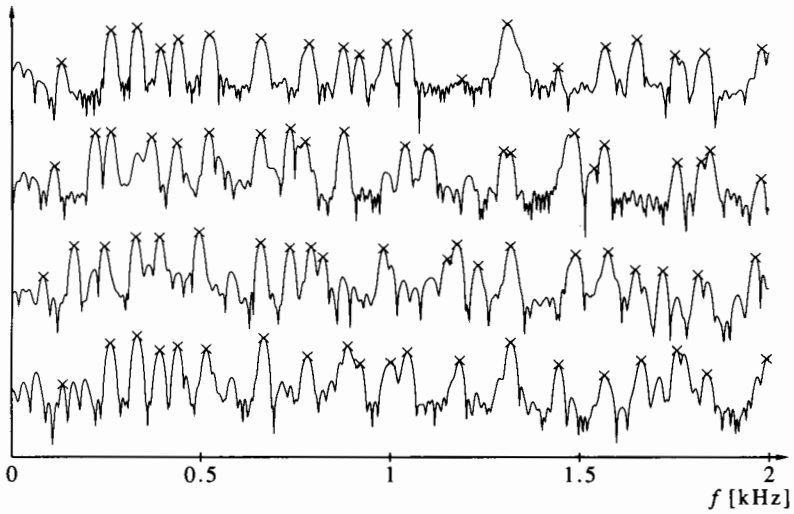

Figure 6.3 Spectra of the signals shown in Figure 6.1. x's denote frequencies of chord notes and their harmonics.

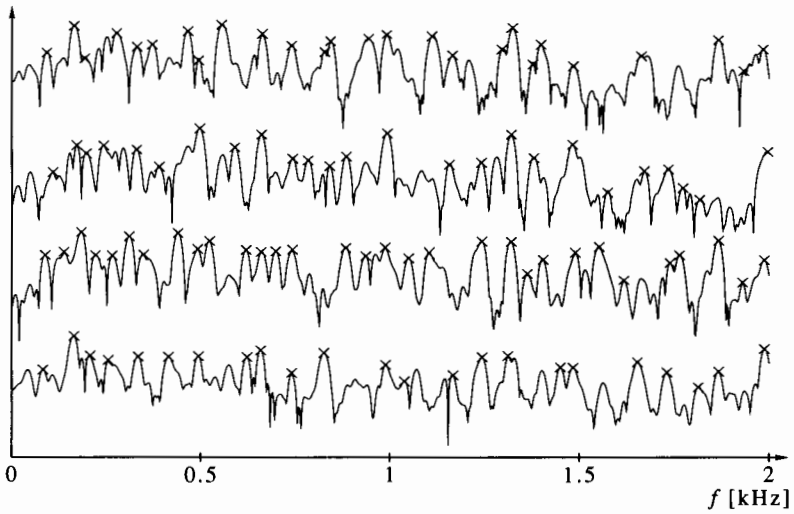

Figure 6.4 Spectra of the signals shown in Figure 6.2. x's denote frequencies of chord notes and their harmonics.

assuming $|a| < 1$. Now consider the Fourier transform of the rectangular-windowed signal,

$$X^{\mathrm{f}}(\theta) = \sum_{n=0}^{N-1} a^n e^{-j\theta n} = \frac{1 - a^N e^{-j\theta N}}{1 - a e^{-j\theta}}. \tag{6.4}$$

Figure 6.5 shows the magnitudes of the Fourier transforms of the original and the rectangular-windowed signals, with $N = 16$ and $a = 0.9$. The transform of $y[n]$ is smooth, that of $x[n]$ is wiggly, but the latter approximately follows the former.

Now return to (6.2); recall that multiplication in the time domain translates to

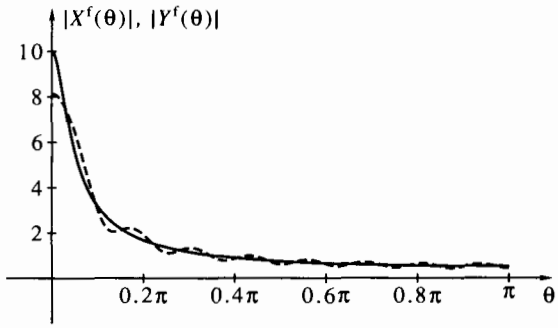

Figure 6.5 The Fourier transform (magnitude) of an exponential sequence. Solid line: unwindowed; dashed line: windowed.

convolution in the frequency domain, so

$$X^f(\theta) = \frac{1}{2\pi}\{Y^f * W_r^f\}(\theta), \tag{6.5}$$

where $W_r^f(\theta)$ is the Fourier transform of the rectangular window, given by

$$W_r^f(\theta) = \sum_{n=0}^{N-1} e^{-j\theta n} = \frac{1 - e^{-j\theta N}}{1 - e^{-j\theta}} = \frac{\sin(0.5\theta N)}{\sin(0.5\theta)} e^{-j0.5\theta(N-1)}. \tag{6.6}$$

The function

$$D(\theta, N) = \frac{\sin(0.5\theta N)}{\sin(0.5\theta)} \tag{6.7}$$

is called the *Dirichlet kernel.* It is shown in Figure 6.6 for $N = 40$. The main properties of the Dirichlet kernel are as follows:

1. Its maximum value is N, occurring at $\theta = 0$.
2. Its zeros nearest to the origin are at $\theta = \pm 2\pi/N$. The region between the two zeros is called the *main lobe* of the Dirichlet kernel.
3. There are additional zeros at $\{\theta = 2m\pi/N,\ m = \pm 2, \pm 3, \ldots\}$. Between every pair of adjacent zeros there is a maximum or a minimum, approximately at $\theta = (2m + 1)\pi/N$. The regions between these zeros are called *side lobes.*
4. The highest side lobe (in absolute value) occurs at $\theta = \pm 3\pi/N$ and its value (for large N) is approximately $2N/3\pi$. The ratio of the highest side lobe to the main lobe is $2/(3\pi)$, or about -13.5 dB.

Had $W_r^f(\theta)$ been an impulse function $\delta(\theta)$, we would have obtained $X^f(\theta) = Y^f(\theta)$. Since $W_r^f(\theta)$ is not an impulse, we get distortions. These distortions are of two kinds:

1. Smearing and, as a consequence, loss of resolution due to the finite width of the main lobe. Any impulselike feature in $Y^f(\theta)$ (such as one resulting from a periodic component in $y[n]$) will have a width approximately $\pm 2\pi/N$ in the spectrum of the rectangular-windowed signal $x[n]$. Two periodic components in $y[n]$ whose frequency separation is less than $2\pi/N$ will blend with each other and may appear as one component in $X^f(\theta)$.
2. Side-lobe interferences. Suppose $y[n]$ contains a strong sinusoidal component at frequency θ_0 and weak components at other frequencies. The side lobes of the strong component may mask the main lobe of the weak component. This masking is worst when the frequency of the weak component differs from θ_0 by an odd multiple of π/N.

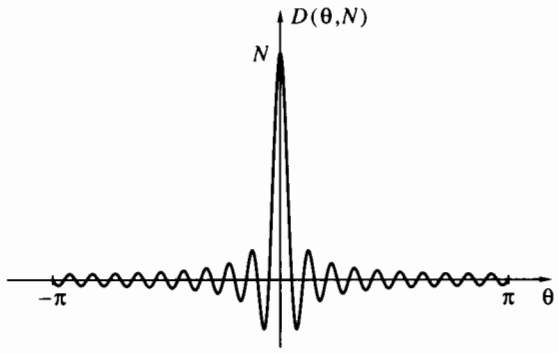

Figure 6.6 The Dirichlet kernel for $N = 40$.

In summary, rectangular windowing has undesirable side effects, so it is generally not a good way of performing short-time spectral analysis. The solution to this problem is presented in the next section.

6.2 Windowing

As we have seen, the undesirable side effects of rectangular windowing are due to the convolution of the Fourier transform of the original signal with the Dirichlet kernel. Suppose, instead, that we wish to obtain $X^f(\theta)$ as the convolution of $Y^f(\theta)$ with another function $W^f(\theta)$. The time-domain multiplication property of the Fourier transform tells us how to do it: We must multiply $y[n]$ by the sequence $w[n]$, the inverse Fourier transform of $W^f(\theta)$. Thus, we must perform the operation

$$x[n] = y[n]w[n], \tag{6.8}$$

and then we will get

$$X^f(\theta) = \frac{1}{2\pi}\{Y^f * W^f\}(\theta). \tag{6.9}$$

However, the sequence $w[n]$ cannot be arbitrary. First, it must have finite duration, else it will not select a finite segment of $y[n]$. Second, its length N must agree with the desired length of the finite segment we wish to analyze. Third, the elements of $w[n]$ should be nonnegative, because it is not reasonable to invert the polarities of the signal values. Within these limitations, we would like $W^f(\theta)$ to have:

1. A main lobe that is as narrow as possible.
2. Side lobes that are as low as possible.

The sequence $w[n]$ is called a *window*, and the operation (6.8) is called *windowing*. Windowing amounts to truncating the signal $y[n]$ to a finite length N while reshaping it through multiplication. The special case of rectangular windowing corresponds to truncation without reshaping. The transform $W^f(\theta)$ of the window is called the *kernel function* of the window [hence the name Dirichlet kernel for (6.7)]. Figure 6.7 depicts a hypothetical desired shape of a kernel. The kernel of a good window should be a good approximation to a delta function, since then the convolution operation (6.9) will not distort $Y^f(\theta)$ too much. However, the inverse Fourier transform of $W^f(\theta) = 2\pi\delta(\theta)$ is $w[n] \equiv 1$, and this does not have a finite duration. Choosing a window always involves trade-off between the width of the main lobe and the level of the side lobes. In general, the narrower the main lobe, the higher the side lobes, and vice versa. The rectangular

window has the narrowest possible main lobe of all windows of the same length, but its side lobes are the highest. The side-lobe level of the rectangular window, which is -13.5 dB, is undesired in most applications. We are almost always ready to increase the main-lobe width (and degrade the frequency resolution) in order to reduce the side-lobe level.

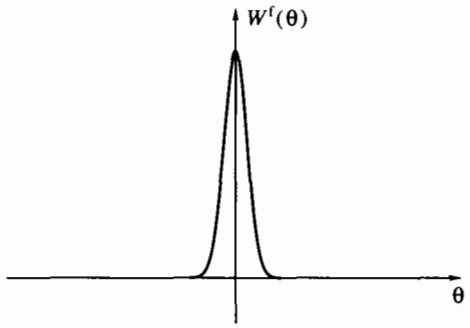

Figure 6.7 A hypothetical window kernel function.

To reiterate:

> The main parameters of a window by which we judge its suitability to a given task are (1) its main-lobe width relative to the width of the main lobe of a rectangular window of the same length N and (2) its side-lobe level relative to the amplitude of the main lobe.

6.3 Common Windows

6.3.1 Rectangular Window

The rectangular window was discussed previously, so we just repeat its main properties for completeness. The width of its main lobe is $4\pi/N$ and its side-lobe level is -13.5 dB. Figure 6.8 depicts a rectangular window with $N = 41$, in the time and frequency domains. The magnitude response of the kernel function is shown in decibels, for better visualization of the side-lobe level.

6.3.2 Bartlett Window

Bartlett's method of side-lobe level reduction is based on squaring of the kernel function of the rectangular window, thereby reducing the side-lobe level by a factor of 2 (in dB). Squaring in the frequency domain is equivalent to convolving a rectangular window with itself in the time domain. Suppose that the desired window length N is odd and let $w_r[n]$ be a rectangular window of length $(N+1)/2$. Define

$$w_t[n] = \frac{2}{N+1}\{w_r * w_r\}[n] = 1 - \frac{|2n - N + 1|}{N+1}, \quad 0 \le n \le N-1. \tag{6.10}$$

The corresponding kernel function is then

$$W_t^f(\theta) = \frac{2}{N+1}D^2(\theta, 0.5(N+1))e^{-j0.5\theta(N-1)} = \frac{2\sin^2[0.25\theta(N+1)]}{(N+1)\sin^2(0.5\theta)}e^{-j0.5\theta(N-1)}. \tag{6.11}$$

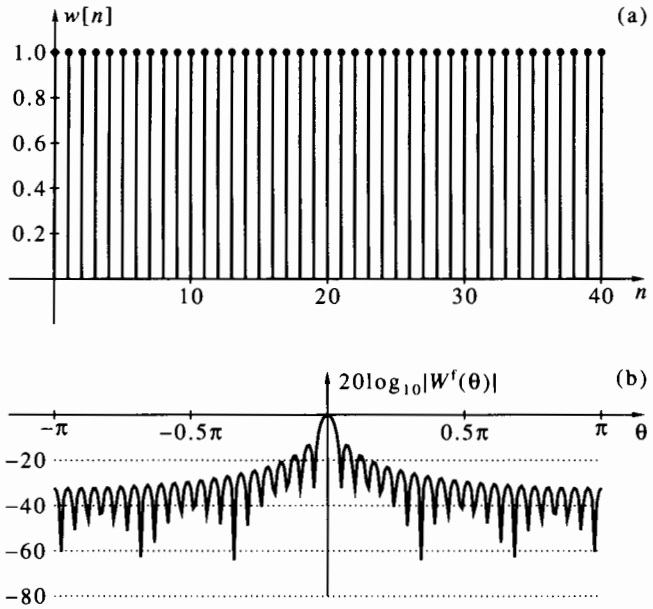

Figure 6.8 A rectangular window, $N = 41$: (a) time-domain plot; (b) frequency-domain magnitude plot.

This window is called the *Bartlett window* (after its discoverer) or a *triangular window* (owing to its shape). Figure 6.9 depicts the Bartlett window in the time and frequency domains, for $N = 41$. As is clear from its construction, the side-lobe level of the Bartlett window is -27 dB. The width of the main lobe is $8\pi/(N+1)$, which is nearly twice that of a rectangular window of the same length.

When N is even, we can obtain the Bartlett window by convolving a rectangular window of length $N/2$ with one of length $(N+2)/2$. The time-domain window function is again (6.10) and the kernel function is (see Problem 6.2)

$$\begin{aligned} W_{\mathrm{t}}^{\mathrm{f}}(\theta) &= \frac{2}{(N+1)} D(\theta, 0.5N) D(\theta, 0.5(N+2)) e^{-j0.5\theta(N-1)} \\ &= \frac{2\sin(0.25\theta N)\sin[0.25\theta(N+2)]}{(N+1)\sin^2(0.5\theta)} e^{-j0.5\theta(N-1)}. \end{aligned} \tag{6.12}$$

6.3.3 Hann Window

Whereas the Bartlett window achieves side-lobe level reduction by squaring, the Hann window achieves a similar effect by superposition. The kernel function of the Hann window is obtained by adding three Dirichlet kernels, shifted in frequency so as to yield partial cancellation of their side lobes. Figure 6.10 illustrates the construction of the Hann window. The two side kernels are shifted by $\pm 2\pi/(N-1)$ with respect to the one at the center. The center kernel has magnitude 0.5 and the ones at the sides have magnitudes 0.25 each. Figure 6.10 also shows the sum of the three kernels, in which attenuation of the side lobes is apparent.

The kernel function of the Hann window is given by

$$\begin{aligned} W_{\mathrm{hn}}^{\mathrm{f}}(\theta) &= \left[0.5D(\theta, N) + 0.25D\left(\theta - \frac{2\pi}{N-1}, N\right) + 0.25D\left(\theta + \frac{2\pi}{N-1}, N\right)\right] e^{-j0.5\theta(N-1)} \\ &= 0.5W_{\mathrm{r}}^{\mathrm{f}}(\theta) - 0.25W_{\mathrm{r}}^{\mathrm{f}}\left(\theta - \frac{2\pi}{N-1}\right) - 0.25W_{\mathrm{r}}^{\mathrm{f}}\left(\theta + \frac{2\pi}{N-1}\right). \end{aligned} \tag{6.13}$$

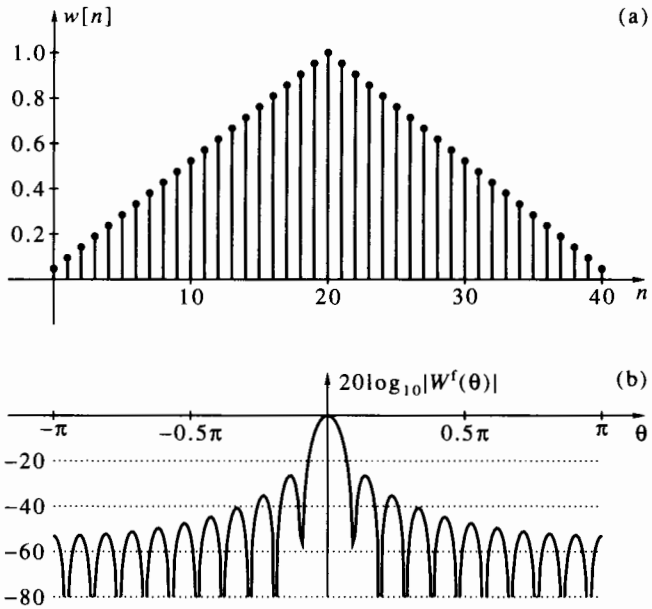

Figure 6.9 Bartlett window, $N = 41$: (a) time-domain plot; (b) frequency-domain magnitude plot.

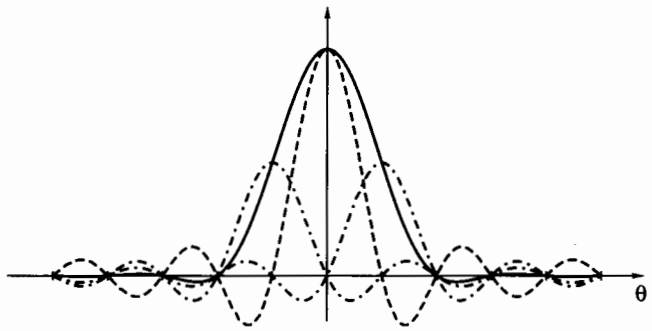

Figure 6.10 Construction of the Hann window from three Dirichlet kernels. Dashed line: center kernel; dot-dashed lines: shifted kernels; solid line: the sum.

By the modulation property of the Fourier transform we get that the window sequence corresponding to (6.13) is

$$\begin{aligned} w_{\text{hn}}[n] &= 0.5 - 0.25\exp\left(\frac{j2\pi n}{N-1}\right) - 0.25\exp\left(-\frac{j2\pi n}{N-1}\right) \\ &= 0.5\left[1 - \cos\left(\frac{2\pi n}{N-1}\right)\right], \quad 0 \le n \le N-1. \end{aligned} \tag{6.14}$$

The Hann window is also known in the literature as the "Hanning" window, for reasons explained in the next section. It is also called a cosine window. Figure 6.11 depicts the Hann window in the time and frequency domains, for $N = 41$. The side-lobe level of this window is -32 dB and the width of the main lobe is $8\pi/N$. The main lobe width is the same as that of the Bartlett window, but the side-lobe level is lower.

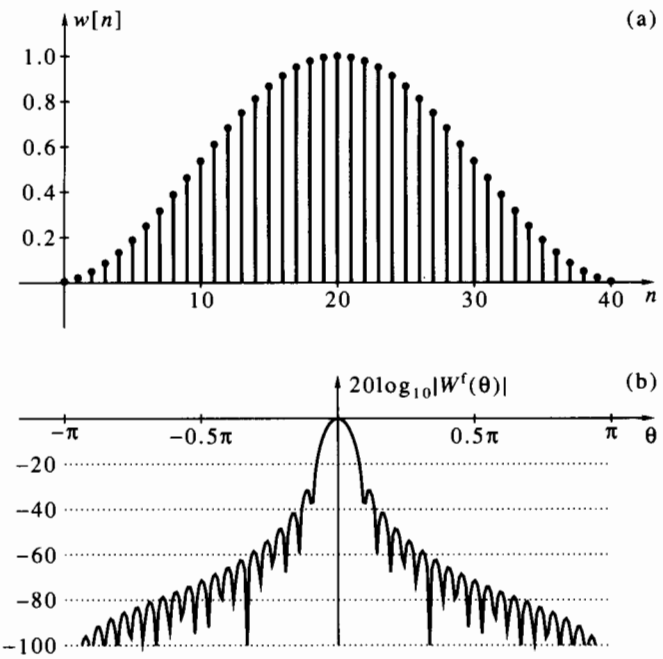

Figure 6.11 Hann window, $N = 41$: (a) time-domain plot; (b) frequency-domain magnitude plot.

The Hann window has a peculiar property: Its two end points are zero. When applied to a signal $y[n]$, it effectively deletes the points $y[0]$ and $y[N-1]$. This suggests increasing the window length by 2 with respect to the desired N and deleting the two end points. The modified Hann window thus obtained is

$$w_{\text{hn}}[n] = 0.5\left\{1 - \cos\left[\frac{2\pi(n+1)}{N+1}\right]\right\}, \quad 0 \le n \le N-1; \tag{6.15}$$

the kernel function changes accordingly.

6.3.4 Hamming Window

The Hamming window is obtained by a slight modification of the Hann window, which amounts to choosing different magnitudes for the three Dirichlet kernels. The window and kernel function are

$$w_{\text{hm}}[n] = 0.54 - 0.46\cos\left(\frac{2\pi n}{N-1}\right), \quad 0 \le n \le N-1, \tag{6.16}$$

$$W^{\text{f}}_{\text{hm}}(\theta) = 0.54W^{\text{f}}_{\text{r}}(\theta) - 0.23W^{\text{f}}_{\text{r}}\left(\theta - \frac{2\pi}{N-1}\right) - 0.23W^{\text{f}}_{\text{r}}\left(\theta + \frac{2\pi}{N-1}\right). \tag{6.17}$$

Figure 6.12 shows the three replicas and their sum for the Hamming window. As we see, the side lobes of the sum are lower than those of the Hann window. Hamming got the numbers 0.54 and 0.46 by trial and error, seeking to minimize the amplitude of the highest side lobe.

Figure 6.13 depicts the Hamming window in the time and frequency domains, for $N = 41$. The side-lobe level of this window is -43 dB; the main-lobe width is $8\pi/N$, the same as that of the Bartlett and Hann windows. A peculiar property of this window is that the highest side lobe is not the one nearest to the main lobe. Its end points are not zero, so its length need not be increased by 2, as in the case of the Hann window.

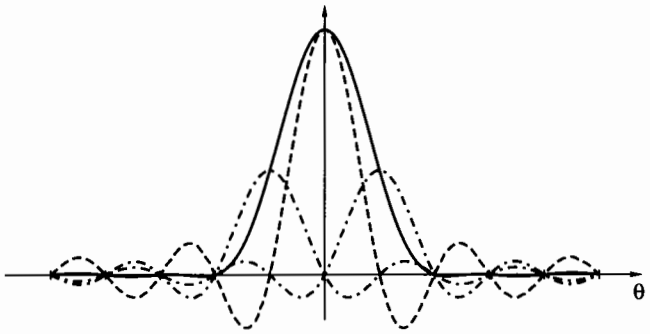

Figure 6.12 Construction of the Hamming window from three Dirichlet kernels. Dashed line: center kernel; dot-dashed lines: shifted kernels; solid line: the sum.

The Hamming window is also called a raised-cosine window. When Blackman and Tukey described the Hann and Hamming windows in their classical book of 1958, they nicknamed the former "Hanning," which though probably intended as a pun, was its official name for many years. Recently, however, the window has been properly renamed *Hann* after its inventor.

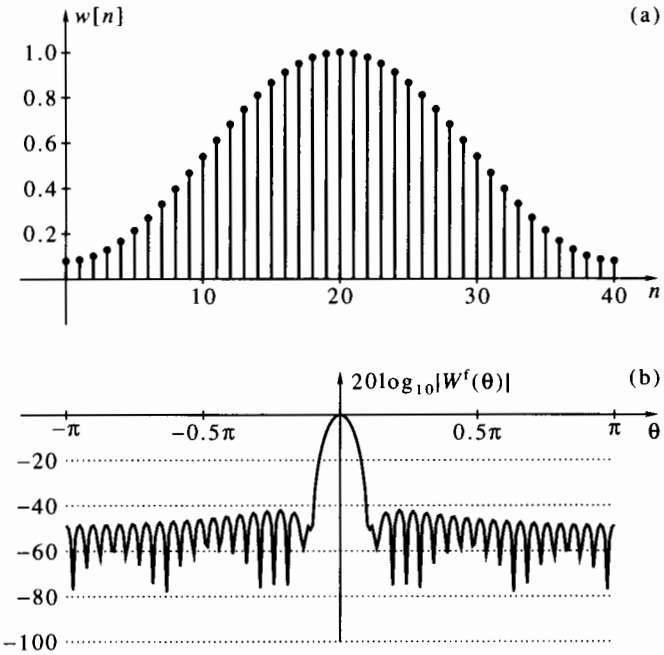

Figure 6.13 Hamming window, $N = 41$: (a) time-domain plot; (b) frequency-domain magnitude plot.

6.3.5 Blackman Window

The Hamming window has the lowest possible side-lobe level among all windows based on three Dirichlet kernels. The Blackman window uses five Dirichlet kernels, thus reducing the side-lobe level still further. The Blackman window and its kernel function

are

$$w_b[n] = 0.42 - 0.5\cos\left(\frac{2\pi n}{N-1}\right) + 0.08\cos\left(\frac{4\pi n}{N-1}\right), \quad 0 \le n \le N-1, \tag{6.18}$$

$$W_b^f(\theta) = 0.42W_r^f(\theta) - 0.25W_r^f\left(\theta + \frac{2\pi}{N-1}\right) - 0.25W_r^f\left(\theta - \frac{2\pi}{N-1}\right) + 0.04W_r^f\left(\theta + \frac{4\pi}{N-1}\right) + 0.04W_r^f\left(\theta - \frac{4\pi}{N-1}\right). \tag{6.19}$$

Figure 6.14 depicts the Blackman window in the time and frequency domains, for $N = 41$. The side-lobe level of the Blackman window is -57 dB and the width of the main lobe is $12\pi/N$. As in the case of the Hann window, the two end points of the Blackman window are zero, so in practice we can increase N by 2 and remove the two end points. The weights of the Blackman window are not optimized to minimize the side-lobe level. Such optimization was performed by Harris [1978] and is discussed in Problem 6.8.

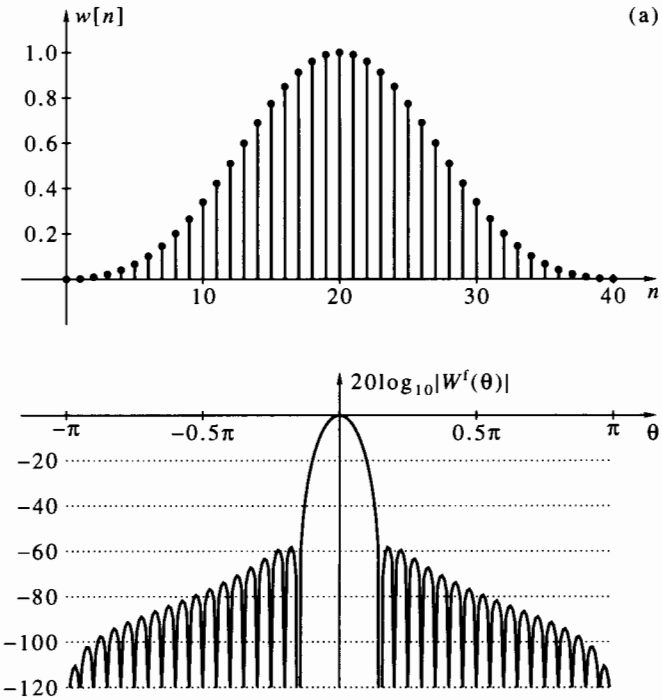

Figure 6.14 Blackman window, $N = 41$: (a) time-domain plot; (b) frequency-domain magnitude plot.

6.3.6 Kaiser Window

The windows described so far are considered as classical. They have been derived based on intuition and educated guesses. Modern windows are based on optimality criteria; they aim to be best in a certain respect, while meeting certain constraints. Different optimality criteria give rise to different windows. Of special importance in this category are the following optimality criteria:

1. *Dolph's criterion:* Minimize the width of the main lobe of the kernel, under the constraint that the window length be fixed and the side-lobe level not exceed a given maximum value. The window values depend on the length and the permitted side-lobe level.
2. *Kaiser's criterion:* Minimize the width of the main lobe of the kernel, under the constraint that the window length be fixed and the energy in the side lobes not exceed a given percentage of the total energy. The energy in the side lobes is defined as the integral of the square magnitude of the kernel function over the range $[-\pi, \pi]$, excluding the interval of the main lobe. The window values depend on the length and the permitted energy in the side lobes.

Of the windows based on these two criteria, the Kaiser window is much more popular than the Dolph window. Kaiser's criterion gives rise to a family of windows that has become, perhaps, the most widely used for modern digital signal processing [Kuo and Kaiser, 1966]. We now describe the Kaiser window.

The solution to Kaiser's optimization problem is described in terms of the *modified Bessel function* of order zero. This function is given by the infinite power series

$$I_0(x) = \sum_{k=0}^{\infty} \left(\frac{x^k}{2^k k!} \right)^2. \tag{6.20}$$

Using this function, the Kaiser window is given by

$$w_k[n] = \frac{I_0\left[\alpha\sqrt{1 - \left(\frac{|2n-N+1|}{N-1}\right)^2}\right]}{I_0[\alpha]}, \quad 0 \le n \le N-1. \tag{6.21}$$

The parameter α is used for controlling the main-lobe width and the side-lobe level. In general, higher α leads to a wider main lobe and lower side lobes. Figure 6.15 shows the dependence of the main-lobe width (as a multiple of $2\pi/N$) and the side-lobe level on α. The jagged appearance of the upper graph is due to the finite window length used for the computation. In theory (for window length approaching infinity), the graph should be smooth.

Figure 6.16 depicts the Kaiser window in the time and frequency domains, for $N = 41$ and $\alpha = 12$. In this case, the main-lobe width is $16\pi/N$ and the side-lobe level is -90 dB.

6.3.7 Dolph Window*

As we have said, Dolph's criterion (also called the Dolph–Chebyshev criterion) for window design is the minimization of the main-lobe width under the constraint that the window length be fixed and the side-lobe level not exceed a given maximum value [Dolph, 1946]. The mathematical theory underlying the Dolph window is beyond the scope of this book, so we describe the window, but not its derivation.

The kernel function of the Dolph window is

$$W_d^f(\theta) = \begin{cases} C\cos\{(N-1)\arccos[\beta\cos(0.5\theta)]\}, & |\beta\cos(0.5\theta)| \le 1, \\ C\cosh\{(N-1)\operatorname{arccosh}[\beta\cos(0.5\theta)]\}, & |\beta\cos(0.5\theta)| > 1, \end{cases} \tag{6.22}$$

where

$$\beta = \cosh[(N-1)^{-1}\operatorname{arccosh}(10^{-0.05\alpha})], \tag{6.23}$$

and α is the side-lobe level in decibels (a negative number). The parameter C is a normalization constant, chosen to make the middle coefficient of the window equal to

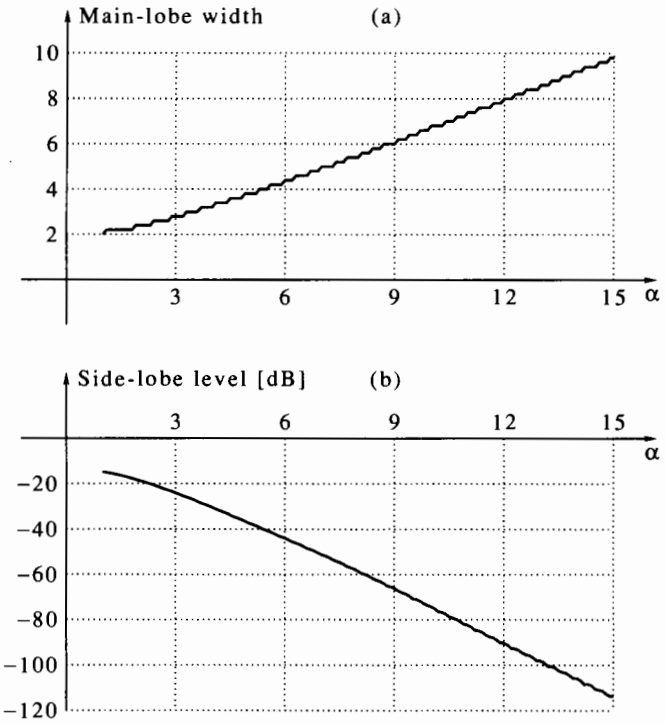

Figure 6.15 Properties of the Kaiser window as a function of the parameter α: (a) main-lobe width, as a multiple of $2\pi/N$; (b) side-lobe level.

1. The functions *cosh* and *arccosh* are

$$\cosh(x) = 0.5(e^x + e^{-x}), \quad \operatorname{arccosh}(x) = \log_e[x + (x^2 - 1)^{1/2}].$$

The window sequence $w_d[n]$ is obtained by sampling $W_d^f(\theta)$ at the N points (where N is the desired window length)

$$\theta[k] = \frac{2\pi k}{N}, \quad 0 \le k \le N - 1,$$

and computing the inverse DFT of the result. The procedure is straightforward when N is odd, but needs care when N is even. For even N, we want to get a window that is symmetric around the fractional point $0.5(N - 1)$. This requires shifting to the left by half a sample, which is equivalent to multiplying $W_d^f(\theta[k])$ by the sequence $\exp(j\pi k/N)$ before the inverse DFT. Problem 4.6 provides the theoretical justification for this procedure. We finally must swap the two halves of the inverse DFT to get the sequence $w_d[n]$ (for both even and odd N).

Figure 6.17 illustrates the Dolph window in the time and frequency domains for $N = 41$ and $\alpha = -60$ dB. As we see, the side-lobe level is constant for all side lobes. Because of this property, the Dolph window is called an *equiripple window.*

The Dolph window has a certain idiosyncrasy: The two end points, $w_d[0]$ and $w_d[N - 1]$, are sometimes larger in magnitude than points nearer to the center. This is not noticeable in Figure 6.17, but becomes more pronounced as N increases. Many engineers are reluctant to use a window that emphasizes the end points of the data segment (although there is no theoretical objection to this), so the Dolph window is not popular. Another reason for the rare use of this window is its high sensitivity to coefficient accuracy, which prevents its use in computers having short word lengths.

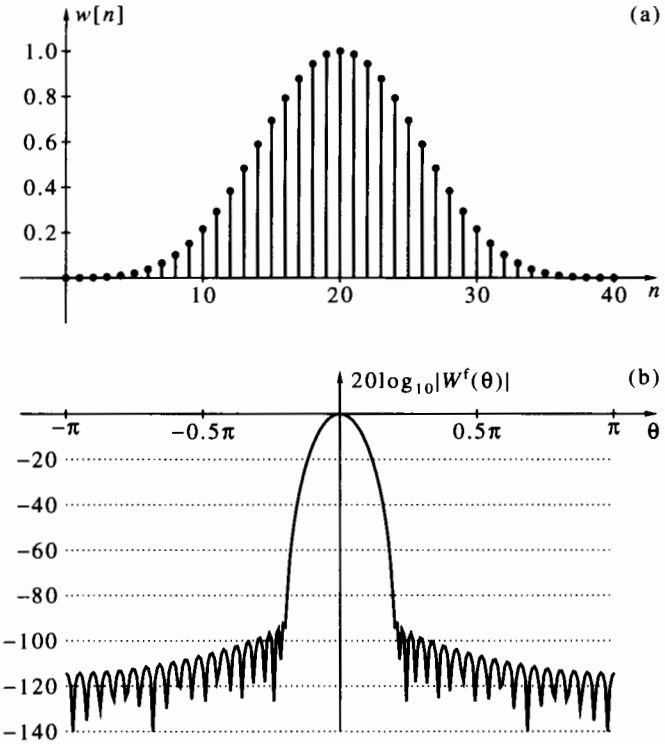

Figure 6.16 Kaiser window, $N = 41$, $\alpha = 12$: (a) time-domain plot; (b) frequency-domain magnitude plot.

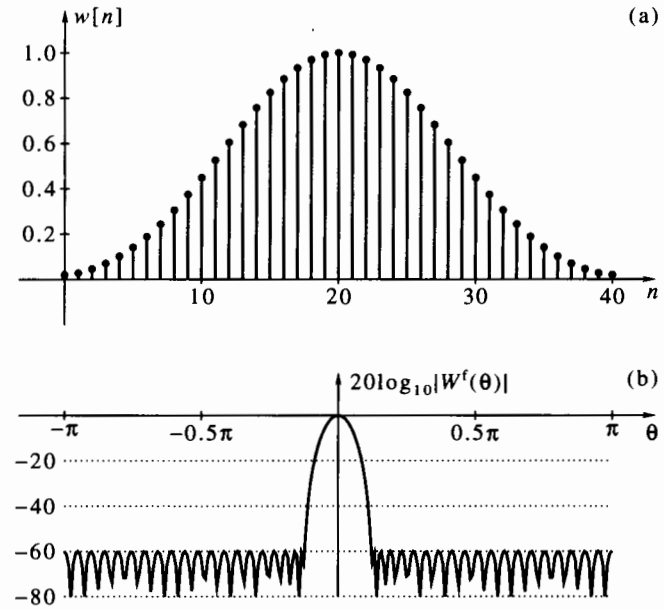

Figure 6.17 Dolph window, $N = 41$, $\alpha = -60$ dB: (a) time-domain plot; (b) frequency-domain magnitude plot.

6.3.8 MATLAB Implementation of Common Windows

The MATLAB Signal Processing Toolbox contains the functions `boxcar`, `bartlett`, `hanning`, `hamming`, `blackman`, `kaiser`, and `chebwin`, which generate the seven window sequences described in this section. Here we also include, in Program 6.1, MATLAB implementations of these windows. The procedure `window` accepts the length N and the window's name as input arguments. In the case of Kaiser or Dolph windows, it accepts the parameter α as a third argument. The output is the desired window sequence. The Dolph window is implemented in `dolph`, shown in Program 6.2.

The following differences exist between Program 6.1 and the aforementioned MATLAB routines:

1. Whereas the MATLAB output is a column vector, ours is a row vector.
2. The output of the MATLAB command `bartlett(N)` is a vector of N elements, the first and last of which are zero. When these two zeros are deleted, the remaining vector is identical to the one obtained by entering the command `window(N-2,'bart')`.
3. The output of the command `window(N,'hann')` is a vector of N elements, the first and last of which are zero. When these two zeros are deleted, the remaining vector is identical to the output of the MATLAB command `hanning(N)`.

6.4 Frequency Measurement

One of the most important applications of the DFT is the measurement of frequencies of periodic signals, in particular sinusoidal signals. Sinusoidal signals are prevalent in science and engineering and the need to measure the frequency of a sinusoidal signal arises in numerous applications. The Fourier transform is a natural tool for this purpose. Practical signals are measured over a finite time interval and the Fourier transform can be computed only on a discrete set of frequencies. The implications of these two restrictions on the theory and practice of frequency measurement are explored in this section.

It is convenient, from a pedagogical point of view, to deal with complex exponential signals first, and proceed to real-valued sinusoids later. In practice, real-valued sinusoids are much more common. However, in certain applications (such as radar and communication), complex signals appear naturally. Therefore, the treatment of complex exponential signals is useful not only as an introduction to the case of real signals, but also for its own merit.

6.4.1 Frequency Measurement for a Single Complex Exponential

The simplest case of a signal for which frequency measurement is a meaningful problem is a single complex exponential signal. Suppose we are given the signal

$$y(t) = Ae^{j(\omega_0 t + \phi_0)}, \tag{6.24}$$

and we wish to measure the frequency ω_0. We sample the signal at interval T and collect N consecutive data points, thus getting the discrete-time signal

$$y[n] = Ae^{j(\theta_0 n + \phi_0)}, \quad 0 \le n \le N-1, \tag{6.25}$$

where $\theta_0 = \omega_0 T$. We assume that $-\pi < \omega_0 T < \pi$, so measurement of θ_0 implies unambiguous measurement of ω_0.

The Fourier transform of the sampled signal is given by

$$Y^{\mathrm{f}}(\theta) = Ae^{j\phi_0}\sum_{n=0}^{N-1} e^{-j(\theta-\theta_0)n} = Ae^{-j[0.5(\theta-\theta_0)(N-1)-\phi_0]}D(\theta-\theta_0,N). \tag{6.26}$$

In particular, since $D(0,N) = N$, evaluation of the Fourier transform at the frequency $\theta = \theta_0$ gives

$$Y^{\mathrm{f}}(\theta_0) = NAe^{j\phi_0}, \quad \text{therefore} \quad |Y^{\mathrm{f}}(\theta_0)| = NA. \tag{6.27}$$

Furthermore, since $|D(\theta,N)| < N$ for all $\theta \neq 0$, the point $\theta = \theta_0$ is the unique global maximum of $|Y^{\mathrm{f}}(\theta)|$ on $-\pi < \theta < \pi$. We therefore conclude that θ_0 can be obtained by finding the point of global maximum of $|Y^{\mathrm{f}}(\theta)|$ in the frequency range $(-\pi, \pi)$. To reiterate:

> The magnitude of the Fourier transform of a finite segment of a complex exponential signal $y[n] = Ae^{j(\theta_0 n+\phi_0)}$ exhibits a unique global maximum at the frequency $\theta = \theta_0$, thus enabling the determination of θ_0 from $|Y^{\mathrm{f}}(\theta)|$.

In practice, it is not possible to find the global maximum of $|Y^{\mathrm{f}}(\theta)|$ exactly, since we cannot evaluate this function at an infinite number of frequency points. A first approximation can be obtained by computing the DFT of $y[n]$ and searching for the point of maximum of $Y^{\mathrm{d}}[k]$. The index k_0 for which $Y^{\mathrm{d}}[k]$ is maximized yields a corresponding frequency $\theta[k_0] = 2\pi k_0/N$ and this serves as the measured value of θ_0 (subtracting 2π if $k_0 \geq 0.5N$). Better approximations can be obtained, if necessary, by either zero padding the sequence $y[n]$ or using a chirp Fourier transform, as explained in Section 5.7. In summary, measurement of the frequency of a single complex exponential can be accomplished by simple DFT or by a DFT-based algorithm, depending on the desired accuracy.

6.4.2 Frequency Measurement for Two Complex Exponentials

Proceeding to a more difficult problem, we now consider a signal consisting of two complex exponentials, that is,

$$y(t) = A_1e^{j(\omega_1 t+\phi_1)} + A_2e^{j(\omega_2 t+\phi_2)}. \tag{6.28}$$

Our aim is to measure the frequencies ω_1 and ω_2. As before, we sample the signal at interval T and collect N measurements of the sampled signal to obtain

$$y[n] = A_1e^{j(\theta_1 n+\phi_1)} + A_2e^{j(\theta_2 n+\phi_2)}, \quad 0 \leq n \leq N-1, \tag{6.29}$$

where $\theta_i = \omega_i T$, $i = 1, 2$; we assume that $-\pi < \omega_i T < \pi$.

The Fourier transform of the sampled signal is given by

$$\begin{aligned} Y^{\mathrm{f}}(\theta) &= A_1e^{j\phi_1}\sum_{n=0}^{N-1} e^{-j(\theta-\theta_1)n} + A_2e^{j\phi_2}\sum_{n=0}^{N-1} e^{-j(\theta-\theta_2)n} \\ &= A_1e^{-j[0.5(\theta-\theta_1)(N-1)-\phi_1]}D(\theta-\theta_1,N) \\ &\quad + A_2e^{-j[0.5(\theta-\theta_2)(N-1)-\phi_2]}D(\theta-\theta_2,N). \end{aligned} \tag{6.30}$$

In particular, evaluation of the Fourier transform at the frequency $\theta = \theta_1$ gives

$$Y^{\mathrm{f}}(\theta_1) = NA_1e^{j\phi_1} + A_2e^{-j[0.5(\theta_1-\theta_2)(N-1)-\phi_2]}D(\theta_1-\theta_2,N). \tag{6.31}$$

We observe the following:

1. If $A_2 = 0$, that is, if only one complex sinusoid is present, then $|Y^f(\theta_1)| = NA_1$, as before. Therefore we can find θ_1 from the maximum point of $|Y^f(\theta)|$.
2. If $A_2 \neq 0$, but
$$|A_2 D(\theta_1 - \theta_2, N)| \ll NA_1, \tag{6.32}$$
we still have a good chance of finding a local maximum of $|Y^f(\theta)|$ near θ_1, since the behavior of $|Y^f(\theta)|$ near θ_1 will not be much perturbed by the second term in (6.31). However, this local maximum is not necessarily a global maximum any more, because the second term in (6.31) affects the global behavior of $|Y^f(\theta)|$.
3. Condition (6.32) will hold if $|\theta_2 - \theta_1| \geq 2\pi/N$, and if A_2 is not much larger than A_1. Otherwise, (6.32) may fail, and we may be unable to find a local maximum of $|Y^f(\theta)|$ near θ_1.
4. The discussion is symmetric with respect to the two sinusoidal components. We can find another local maximum of $|Y^f(\theta)|$ near θ_2, provided the interference from the Dirichlet kernel centered at θ_1 is sufficiently small in the vicinity of θ_2.

Example 6.1 Figure 6.18 shows the behavior of $|Y^f(\theta)|$ in the range of θ of interest (in dB below the maximum) for several choices of θ_1, θ_2, A_1, and A_2 (in all cases $N = 64$). In parts a, b, and c the amplitudes of the two complex exponentials are equal. In part a, the difference $\theta_2 - \theta_1$ is $2\pi/N$. As we see, the two peaks are well separated, indicating the presence of two complex exponentials. However, the peaks appear to repel each other, causing their locations to deviate slightly from the true θ_1 and θ_2. This phenomenon is called *bias*; it typically occurs when the frequencies are close to each other. In part b, the frequency difference is $1.5\pi/N$. Now the two peaks start to merge with each other, but are still distinct; bias is again apparent. In part c, the frequency difference is π/N. In this case there is only one peak, so the two complex exponentials cannot be distinguished from each other.

In parts d, e, and f the frequencies are the same as in parts a, b, and c, respectively, but $A_2 = 0.25A_1$. There are still two distinguishable peaks when $\theta_2 - \theta_1 = 2\pi/N$ (part d); however, the weaker component is considerably biased. When the frequency difference decreases, the weaker component becomes invisible. □

Condition (6.32) may be hard to satisfy since the side lobes of the Dirichlet kernel are relatively high; the highest side lobe is only 13.5 dB lower than the main lobe. For example, if $|\theta_2 - \theta_1| = 3\pi/N$ and A_2 is larger than A_1 by more than 13.5 dB, the Dirichlet kernel centered at θ_2 will interfere with the one centered at θ_1 and may prohibit measurement of θ_1. Because of the slow rate of decay of the side lobes of the Dirichlet kernel, however, the problem does not completely disappear, even if the distance between the two frequencies is large with respect to $2\pi/N$.

Windowing alleviates the problems we have described. Suppose we multiply $y[n]$ by a window $w[n]$ prior to computing the Fourier transform and denote
$$x[n] = y[n]w[n].$$
Then (6.30) changes to
$$\begin{aligned} X^f(\theta) &= A_1 e^{j\phi_1} \sum_{n=0}^{N-1} w[n] e^{-j(\theta-\theta_1)n} + A_2 e^{j\phi_2} \sum_{n=0}^{N-1} w[n] e^{-j(\theta-\theta_2)n} \\ &= A_1 e^{j\phi_1} W^f(\theta - \theta_1, N) + A_2 e^{j\phi_2} W^f(\theta - \theta_2, N), \end{aligned} \tag{6.33}$$

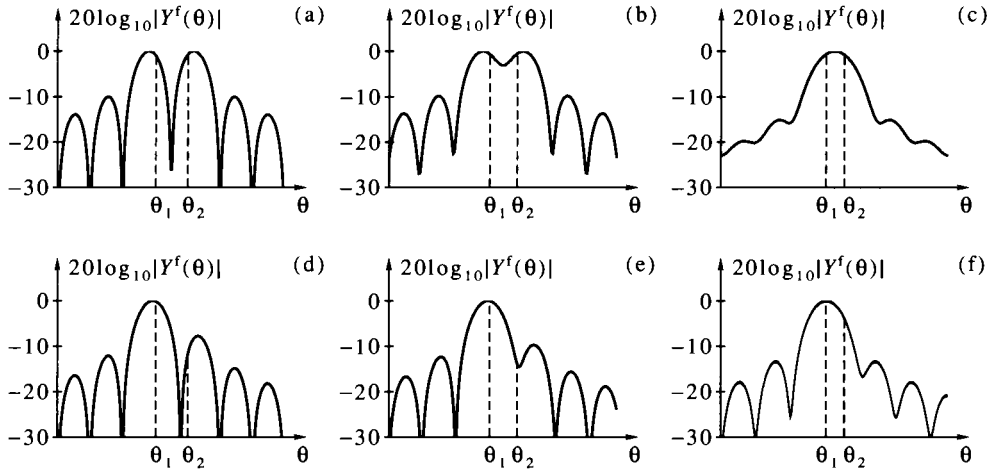

Figure 6.18 The Fourier transform of two complex exponentials in Example 6.1: (a) $A_2 = A_1$, $\theta_2 - \theta_1 = 2\pi/N$; (b) $A_2 = A_1$, $\theta_2 - \theta_1 = 1.5\pi/N$; (c) $A_2 = A_1$, $\theta_2 - \theta_1 = \pi/N$; (d) $A_2 = 0.25A_1$, $\theta_2 - \theta_1 = 2\pi/N$; (e) $A_2 = 0.25A_1$, $\theta_2 - \theta_1 = 1.5\pi/N$; (f) $A_2 = 0.25A_1$, $\theta_2 - \theta_1 = \pi/N$.

where $W^f(\theta, N)$ is the kernel function of the window. In particular,

$$X^f(\theta_1) = A_1 e^{j\phi_1} W^f(0, N) + A_2 e^{j\phi_2} W^f(\theta_1 - \theta_2, N). \tag{6.34}$$

We have $W^f(0, N) = \sum_{n=0}^{N-1} w[n]$, and this sum is approximately proportional to N. Suppose that

$$|A_2 W^f(\theta_1 - \theta_2, N)| \ll A_1 \sum_{n=0}^{N-1} w[n]; \tag{6.35}$$

then we have a good chance of finding a local maximum of $|X^f(\theta)|$ near θ_1. Condition (6.35) holds if $|\theta_2 - \theta_1|$ is greater than the width of the main lobe of the kernel function, and if $20\log_{10}(A_1/A_2)$ is larger than the side-lobe level. We can increase the chances of meeting the latter condition by choosing a window with an extremely low side-lobe level. For example, if we use a Kaiser window with side-lobe level of -80 dB, we will be able to handle sinusoidal components whose amplitudes differ by up to four orders of magnitude. However, this comes at the price of making the frequency separation condition more difficult to meet, due to the widening of the main lobe. We may be able to compensate for this by increasing the number of samples N, depending on the application.

Example 6.2 Figure 6.19 shows the behavior of the windowed DFT $|X^f(\theta)|$ for two complex exponentials, using the Hann window. The amplitudes A_1, A_2 are the same as in Example 6.1. The frequency differences are $8\pi/N$, $6\pi/N$, and $4\pi/N$. As we see, the two peaks are well separated when the frequency difference is $8\pi/N$, and are still distinguishable when it is $6\pi/N$. The bias is considerably smaller than in the case of unwindowed DFT. However, the two complex exponentials become indistinguishable when the frequency difference is $4\pi/N$, whereas with unwindowed DFT they would be well separated (as we recall from Example 6.1). □

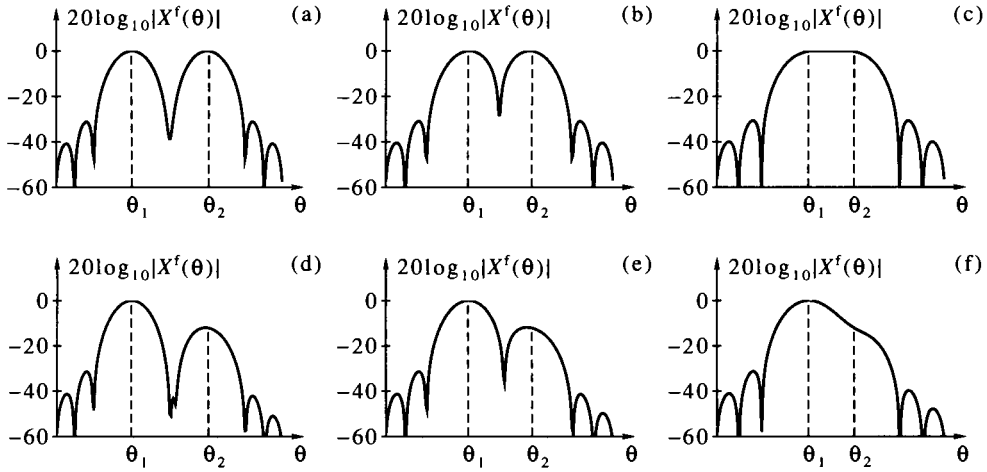

Figure 6.19 The windowed Fourier transform of two complex exponentials in Example 6.2: (a) $A_2 = A_1$, $\theta_2 - \theta_1 = 8\pi/N$; (b) $A_2 = A_1$, $\theta_2 - \theta_1 = 6\pi/N$; (c) $A_2 = A_1$, $\theta_2 - \theta_1 = 4\pi/N$; (d) $A_2 = 0.25A_1$, $\theta_2 - \theta_1 = 8\pi/N$; (e) $A_2 = 0.25A_1$, $\theta_2 - \theta_1 = 6\pi/N$; (f) $A_2 = 0.25A_1$, $\theta_2 - \theta_1 = 4\pi/N$.

6.4.3 Frequency Measurement for Real Sinusoids

We now generalize the frequency measurement problem from 2 to M signal components. We also change from complex exponentials to real-valued sinusoidal signals. The given signal is

$$y(t) = \sum_{k=1}^{M} A_k \cos(\omega_k t + \phi_k). \tag{6.36}$$

Our aim is to measure the frequencies $\{\omega_k,\ 1 \le k \le M\}$. Sometimes the amplitudes A_k and the phases ϕ_k are also of interest, but we do not consider them here. Since we are now dealing with real sinusoids, we need to consider only positive frequencies.

We sample the signal (6.36) at interval T and collect N measurements to get

$$y[n] = \sum_{k=1}^{M} A_k \cos(\theta_k n + \phi_k), \quad 0 \le n \le N-1, \tag{6.37}$$

where $\theta_k = \omega_k T$, $1 \le k \le M$; we assume that $-\pi < \omega_k T < \pi$.

To compute the Fourier transform of the signal $y[n]$, we express it in the form

$$y[n] = \sum_{k=1}^{M} [0.5A_k e^{j\phi_k} e^{j\theta_k n} + 0.5A_k e^{-j\phi_k} e^{-j\theta_k n}]. \tag{6.38}$$

Then,

$$\begin{aligned} Y^f(\theta) &= \sum_{k=1}^{M} 0.5A_k e^{j\phi_k} \left[\sum_{n=0}^{N-1} e^{-j(\theta-\theta_k)n}\right] + \sum_{k=1}^{M} 0.5A_k e^{-j\phi_k} \left[\sum_{n=0}^{N-1} e^{-j(\theta+\theta_k)n}\right] \\ &= \sum_{k=1}^{M} 0.5A_k e^{-j[0.5(\theta-\theta_k)(N-1)-\phi_k]} D(\theta - \theta_k, N) \\ &\quad + \sum_{k=1}^{M} 0.5A_k e^{-j[0.5(\theta+\theta_k)(N-1)+\phi_k]} D(\theta + \theta_k, N). \end{aligned} \tag{6.39}$$

Let us examine the behavior of $Y^f(\theta)$ at one of the sinusoid frequencies themselves,

say at $\theta = \theta_m$:

$$Y^{\mathrm{f}}(\theta_m) = 0.5NA_m e^{j\phi_m} + \sum_{\substack{k=1 \\ k\neq m}}^{M} 0.5A_k e^{-j[0.5(\theta_m-\theta_k)(N-1)-\phi_k]} D(\theta_m - \theta_k, N)$$
$$+ \sum_{k=1}^{M} 0.5A_k e^{-j[0.5(\theta_m+\theta_k)(N-1)+\phi_k]} D(\theta_m + \theta_k, N). \tag{6.40}$$

Suppose that all the following conditions hold:

1. All frequencies $\{\theta_k,\ k \neq m\}$ are far from θ_m by $2\pi/N$ at least.
2. The frequency θ_m is not smaller than π/N and not higher than $\pi(1 - 1/N)$.
3. All $\{A_k, k \neq m\}$ are either smaller or not much larger than A_m.

Then the first term in (6.40) will dominate the sum, so $|Y^{\mathrm{f}}(\theta)|$ can be expected to have a local maximum near $\theta = \theta_m$. If we assume that the conditions hold for *all* θ_m, then $|Y^{\mathrm{f}}(\theta)|$ can be expected to exhibit M local maxima in the range $\theta \in (0, \pi)$ near the true values of the θ_m. Of course, the total number of local maxima will usually be larger than M. However, the M maxima of interest correspond to the *main lobes* of the M Dirichlet kernels, so they can be expected to be the largest. We can state the principle of measuring frequencies of real sinusoids by the Fourier transform as follows: Find all local maxima of $|Y^{\mathrm{f}}(\theta)|$ in the range $\theta \in (0, \pi)$ and take the frequencies of the M largest maxima as the measured values of $\{\theta_k\}$.

As in the case of two complex exponentials, the side lobes of the Dirichlet kernel are likely to interfere and prevent this procedure from working. We therefore modify it by applying a window prior to the Fourier transform, that is, we form

$$x[n] = y[n]w[n].$$

Then (6.39) changes to

$$\begin{aligned} X^{\mathrm{f}}(\theta) &= \sum_{n=0}^{N-1} y[n]w[n]e^{-j\theta n} \\ &= \sum_{k=1}^{M} 0.5A_k e^{j\phi_k} \left[\sum_{n=0}^{N-1} w[n]e^{-j(\theta-\theta_k)n}\right] + \sum_{k=1}^{M} 0.5A_k e^{-j\phi_k} \left[\sum_{n=0}^{N-1} w[n]e^{-j(\theta+\theta_k)n}\right] \\ &= \sum_{k=1}^{M} 0.5A_k e^{j\phi_k} W^{\mathrm{f}}(\theta - \theta_k, N) + \sum_{k=1}^{M} 0.5A_k e^{-j\phi_k} W^{\mathrm{f}}(\theta + \theta_k, N). \end{aligned} \tag{6.41}$$

In particular,

$$X^{\mathrm{f}}(\theta_m) = 0.5W^{\mathrm{f}}(0, N)A_m e^{j\phi_m} + \sum_{k\neq m} 0.5A_k e^{j\phi_k} W^{\mathrm{f}}(\theta_m - \theta_k, N)$$
$$+ \sum_{k=1}^{M} 0.5A_k e^{-j\phi_k} W^{\mathrm{f}}(\theta_m + \theta_k, N). \tag{6.42}$$

As we see from (6.42), $|X^{\mathrm{f}}(\theta)|$ is likely to have its M highest local maxima near the $\{\theta_k\}$ if the following conditions are satisfied for each θ_m:

1. All frequencies $\{\theta_k,\ k \neq m\}$ are far from θ_m at least by half the width of the main lobe of $W^{\mathrm{f}}(\theta, N)$.
2. The frequency θ_m is not smaller than half the width of the main lobe of $W^{\mathrm{f}}(\theta, N)$ and not larger than π minus half the width of the main lobe of $W^{\mathrm{f}}(\theta, N)$.
3. The quantities $\{20 \log_{10} A_k\}$ differ from each other by no more than the side-lobe level of the kernel function.

In summary, we state the principle of frequency measurement by Fourier transform as follows:

> Choose a window whose side-lobe level reflects the largest expected ratio between the strongest and weakest sinusoidal components. Multiply the sampled signal by the window and compute $|X^{\mathrm{f}}(\theta)|$. Find all local maxima of $|X^{\mathrm{f}}(\theta)|$ in the range $\theta \in (0, \pi)$. Obtain $\{\theta_k\}$ as the frequencies of the M largest maxima.

6.4.4 Practice of Frequency Measurement

Practical frequency measurement using DFT consists, as a minimum, of the following three steps:

1. Multiplication of the sampled sequence by a window.
2. Computation of the DFT, usually through FFT.
3. Search for the local maxima of the absolute value of the DFT and selection of the maxima of interest.

Additional steps are necessary if measurement of the amplitudes and phases is required as well, but we shall not discuss them here.

As we have explained, the choice of a window requires knowledge of the nature of the signal. Specifically, we need to know the maximum ratio between different A_k, or at least an estimate of the maximum ratio. It is also useful to know the minimum possible separation between frequencies of different components, the distance of the lowest frequency from zero, and the distance of the highest frequency from π. These enable us to tell whether the required resolution can be achieved for the given signal. Without such prior knowledge, the best we can do is to try the procedure on the given signal, but there is no guarantee that it will succeed.

When using basic (windowed) DFT, the accuracy of the measured frequencies is limited by the number of samples. In certain applications, the number of samples may be large, but we may not be able to afford to use them all in a single DFT operation because the number of operations is prohibitively large. In other applications, the number of samples is limited and we must make do with the available amount of data.

As was said before, we have at least two means of interpolating between the DFT points: zero padding and the chirp Fourier transform. The former is recommended when the total number of operations, after zero padding, is not too large. The latter is an attractive alternative when zero padding is too costly in computations. It is often advantageous to repeat the measurement procedure twice in the following manner. In the first pass we compute the DFT at the basic resolution provided by the number of samples N and find the local maxima of interest. In the second pass we compute the DFT at a higher resolution around the local maxima found in the first pass, using the chirp Fourier transform algorithm, to improve the accuracy of the local maxima.

A convenient procedure for finding the M largest local maxima of the DFT is as follows. Suppose we are given the DFT sequence $X^{\mathrm{d}}[k]$. Define

$$\Delta[k] = |X^{\mathrm{d}}[k]| - |X^{\mathrm{d}}[k-1]|, \quad 1 \le k \le N-1. \tag{6.43}$$

Find all occurrences of k_l such that $\Delta[k_l] \ge 0$ and $\Delta[k_l + 1] < 0$. Each k_l is a point of local maximum (at the resolution of the given sequence). Now take the subsequence $|X^{\mathrm{d}}[k_l]|$ and sort it in a decreasing order of magnitude. The indices of the first M points of the sorted sequence, multiplied by $2\pi/N$, are the frequencies of local maxima.

The procedure `maxdft` in Program 6.3 implements frequency measurement in MATLAB. The program contains two main parts. In the first part, the M largest local maxima of the DFT (where M is chosen by the user) are found to an accuracy $\pm\pi/N$. In the second part, the routine `chirpf` (described in Section 5.7) is invoked for each of the local maxima, to find the maximum point to an accuracy $\pm 2\pi/N^2$. The procedure `locmax` in Program 6.4 implements the local maxima search. We note that the input vector must be real (the program needs modification in order to work properly for complex vectors) and that the DFT is optionally windowed.

Example 6.3 Consider the discrete-time signal

$$y[n] = \sin(2\pi \cdot 0.1992n) + 0.005\sin(2\pi \cdot 0.25n), \quad 0 \le n \le 127. \tag{6.44}$$

The frequencies were selected such that the stronger component is in the middle between the points $k = 25$ and $k = 26$ of the DFT, and the weaker component is at $k = 32$ exactly. The amplitude difference is 46 dB in this case.

Figure 6.20 shows the magnitude of the DFT with three windows: rectangular, Bartlett, and Hann. Figure 6.21 continues to show three more windows: Hamming, Blackman, and Kaiser with $\alpha = 10$. As we see, the rectangular and Bartlett windows mask the weaker component and, except for a certain irregularity in the side lobes, we cannot discern anything. The Hann window does show the weaker component, because at a distance of 6.5 units of $2\pi/N$ its side lobes are already low enough. The Hamming window, perhaps unexpectedly, has poorer performance than the Hann window in this case, because of the relative flatness of its side lobes (look again at Figure 6.13!). The Blackman and Kaiser windows both perform well; the weaker component is clearly visible and easily distinguishable from the stronger one. In the Kaiser window, the two components are better separated than in the Blackman window. □

6.5 Frequency Measurement of Signals in Noise*

We now generalize the discussion in Section 6.4 to sinusoidal signals measured with additive white noise. Noise is present to one degree or another in almost all real-life applications. Noise is often broad band and becomes white (or nearly so) after prefiltering and sampling. Proper understanding of the effect of noise on frequency measurement is therefore crucial to practical spectral analysis. The basic method of frequency measurement in the presence of noise is the same as when there is no noise: multiplication of the data vector by a window, computation of the magnitude of the DFT, search for maximum (or several local maxima), followed by an optional fine search, by either zero padding or the chirp Fourier transform. Noise affects this procedure in two respects:

1. It masks the peaks in the magnitude of the DFT (in addition to masking caused by side lobes, which we have already encountered), thus making them more difficult to identify. The problem is then to find the peaks belonging to the sinusoidal signals in the DFT when the DFT contains many noise peaks. This is an example of a general problem known as *signal detection.*
2. It causes the point of maximum to deviate from the true frequency, thus introducing errors to the measured frequencies. Since noise always introduces errors, it is common to refer to frequency measurement in noise as *frequency estimation.* The terms *estimation* and *estimates* imply randomness of the measured parameter(s) due to randomness in the signal.

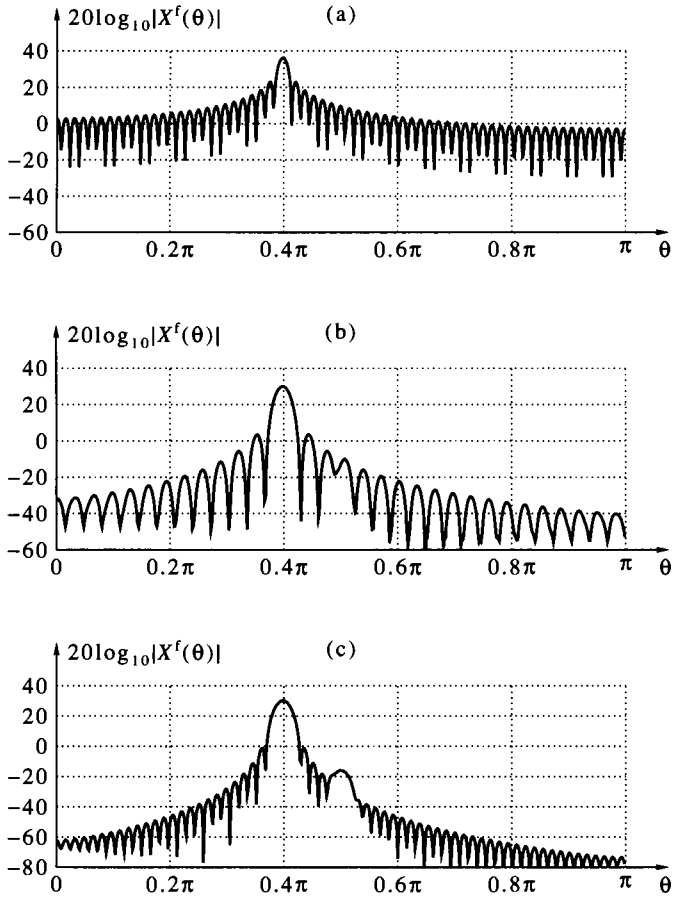

Figure 6.20 Frequency measurement of two sinusoids by windowed DFT: (a) rectangular window; (b) Bartlett window; (c) Hann window.

Our goal in this section is to analyze these two effects quantitatively. Some of the mathematical derivations are based only on the material in Chapter 2; these will be carried out in detail. Other results rely on tools beyond the scope of this book; for these we shall give only the final formulas.

6.5.1 Signal Detection

To simplify the derivations, we examine first the case of a single complex exponential signal. A real sinusoid has two complex exponential components, which interfere with each other and complicate the analysis, so we defer its discussion until later. Since we assume that the signal is complex valued, we also assume that the noise is complex valued for compatibility. Our signal model is thus

$$y[n] = s[n] + v[n] = Ae^{j\theta_0 n} + v[n], \quad 0 \le n \le N-1, \tag{6.45}$$

where $v[n]$ is discrete-time complex white noise,[2] with zero mean and variance γ_v.

The windowed signal is

$$x[n] = y[n]w[n] = Ae^{j\theta_0 n}w[n] + v[n]w[n].$$

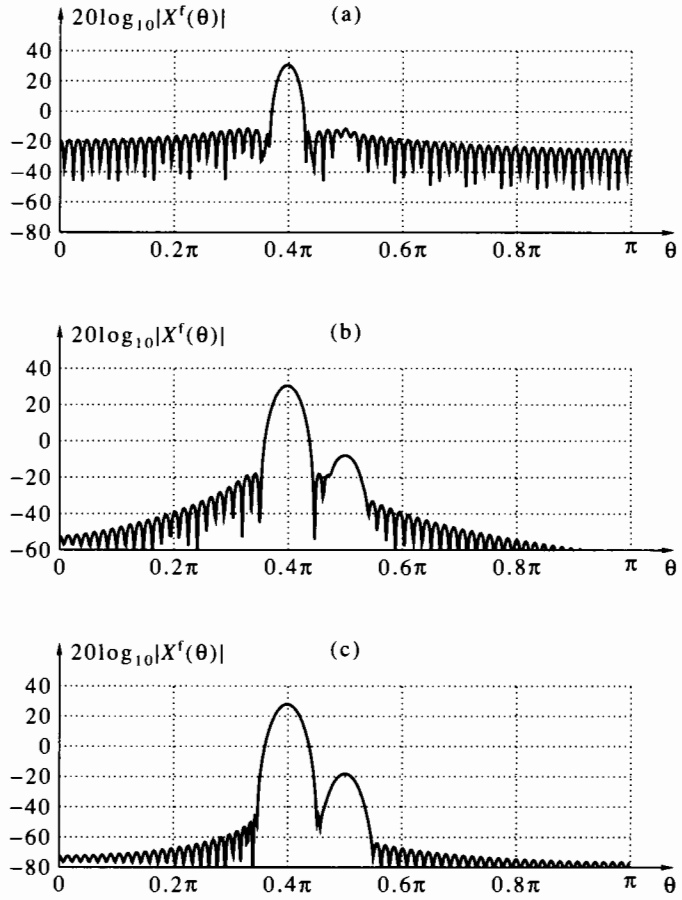

Figure 6.21 Frequency measurement of two sinusoids by windowed DFT: (a) Hamming window; (b) Blackman window; (c) Kaiser window, $\alpha = 10$.

The Fourier transform of the windowed signal is given by

$$X^{\mathrm{f}}(\theta) = A \sum_{n=0}^{N-1} w[n] e^{j(\theta_0 - \theta)n} + \sum_{n=0}^{N-1} w[n] v[n] e^{-j\theta n}. \tag{6.46}$$

Since the elements of the window are always real and nonnegative, the value of θ at which the transform of $s[n]w[n]$ is maximized is $\theta = \theta_0$. The corresponding value of $X^{\mathrm{f}}(\theta_0)$ is

$$X^{\mathrm{f}}(\theta_0) = A \sum_{n=0}^{N-1} w[n] + \sum_{n=0}^{N-1} w[n] v[n] e^{-j\theta_0 n}. \tag{6.47}$$

Let us define

$$\mathrm{CG} = \frac{1}{N} \sum_{n=0}^{N-1} w[n]. \tag{6.48}$$

The parameter CG is called the *coherent gain* of the window. Using this parameter, we can express (6.47) as

$$X^{\mathrm{f}}(\theta_0) = NA \cdot \mathrm{CG} + \sum_{n=0}^{N-1} w[n] v[n] e^{-j\theta_0 n}. \tag{6.49}$$

The coherent gain depends on the window type, but is nearly independent of the window length N. Therefore, the gain of the complex exponential at the maximum point of the Fourier transform is approximately linear in N.

As we saw in the preceding section, frequency measurement is based on the magnitude of the Fourier transform. Here it will be more convenient to work with the square magnitude, so let us derive an expression for $|X^{\mathrm{f}}(\theta_0)|^2$. We have

$$|X^{\mathrm{f}}(\theta_0)|^2 = (NA \cdot \mathrm{CG})^2 + 2NA \cdot \mathrm{CG} \cdot \Re\left\{\sum_{n=0}^{N-1} w[n]v[n]e^{-j\theta_0 n}\right\} + \sum_{n=0}^{N-1}\sum_{m=0}^{N-1} w[n]v[n]w[m]\bar{v}[m]e^{-j\theta_0(n-m)}. \tag{6.50}$$

The square magnitude of the Fourier transform is a random variable; its mean is given by

$$\begin{aligned} E(|X^{\mathrm{f}}(\theta_0)|^2) &= (NA \cdot \mathrm{CG})^2 + 2NA \cdot \mathrm{CG} \cdot \Re\left\{\sum_{n=0}^{N-1} w[n]E(v[n])e^{-j\theta_0 n}\right\} \\ &\quad + \sum_{n=0}^{N-1}\sum_{m=0}^{N-1} w[n]w[m]E(v[n]\bar{v}[m])e^{-j\theta_0(n-m)} \\ &= (NA \cdot \mathrm{CG})^2 + \gamma_v \sum_{n=0}^{N-1}\sum_{m=0}^{N-1} w[n]w[m]\delta[n-m]e^{-j\theta_0(n-m)} \\ &= (NA \cdot \mathrm{CG})^2 + \gamma_v \sum_{n=0}^{N-1} w^2[n]. \end{aligned} \tag{6.51}$$

The first term on the right side of (6.51) comes from the complex exponential signal, whereas the second comes from the noise. We define the *output signal-to-noise ratio* (OSNR) of the windowed Fourier transform as the ratio between the two, that is,

$$\mathrm{SNR_o} = \frac{(NA \cdot \mathrm{CG})^2}{\gamma_v \sum_{n=0}^{N-1} w^2[n]}. \tag{6.52}$$

We also define the *input signal-to-noise ratio* (ISNR) as

$$\mathrm{SNR_i} = \frac{A^2}{\gamma_v}. \tag{6.53}$$

The *processing gain* of the windowed Fourier transform is, by definition, the ratio between the OSNR and N times the ISNR. We get from (6.52), (6.53)

$$\mathrm{PG} = \frac{\mathrm{SNR_o}}{N \cdot \mathrm{SNR_i}} = \frac{N^2(\mathrm{CG})^2}{N \sum_{n=0}^{N-1} w^2[n]} = \frac{\left(\sum_{n=0}^{N-1} w[n]\right)^2}{N \sum_{n=0}^{N-1} w^2[n]}. \tag{6.54}$$

Similarly to the coherent gain, the processing gain depends on the window type, but it is nearly independent of the window length N. Therefore, $\mathrm{SNR_o}$ is approximately linear in N.

Practice shows that when the OSNR is about 14 dB or greater (or 25 in absolute value), the presence of the complex exponential can be detected reliably,[3] and its frequency estimated to a good accuracy. Using the definition of processing gain, we get the following rule of thumb:

If the amplitude of the complex exponential A, the noise variance γ_v, and the number of samples N satisfy

$$\frac{NA^2 \cdot \mathrm{PG}}{\gamma_v} \geq 25, \tag{6.55}$$

there is high probability that the maximum magnitude of the Fourier transform will be near the true frequency; it is then safe to use the windowed Fourier transform for frequency estimation.

The discrete-time signal $y[n]$ is often obtained by antialias filtering and sampling of a continuous-time signal accompanied by broad-band noise. Recall from Section 3.7 that, in this case, the variance of the discrete-time noise is given by $\gamma_v = N_0/T$, where N_0 is the power density of the continuous-time noise, in units of power per hertz, and T is the sampling interval. Then the input SNR is given by

$$\mathrm{SNR_i} = \frac{A^2 T}{N_0}. \tag{6.56}$$

Correspondingly, the rule of thumb (6.55) can be expressed as

$$\frac{DA^2 \cdot \mathrm{PG}}{N_0} \geq 25, \tag{6.57}$$

where $D = NT$ is the measurement interval of the continuous-time signal.

In our analysis so far, we have assumed that the windowed Fourier transform is computed at all frequencies θ in order to find the maximum point. In practice this does not happen, of course, since only a finite number of frequency points can be computed. In the simplest case we compute the windowed DFT of the signal at a resolution of $2\pi/N$. If zero padding or a chirp Fourier transform is used, the resolution is better. Let us denote by $\Delta\theta$ the distance between two adjacent frequencies at which $|X^{\mathrm{f}}(\theta)|$ is computed (this is the frequency resolution of the algorithm being used). Then, if θ_0 is not an integer multiple of $\Delta\theta$, the use of the coherent gain [as defined in (6.48)] and the processing gain [as defined in (6.54)] are not justified any more. Instead, both parameters will depend on the deviation of θ_0 from the nearest integer multiple of $\Delta\theta$. In the worst case, θ_0 is exactly in the middle between two integer multiples of $\Delta\theta$. Accordingly, we define the *worst-case coherent gain* as

$$\mathrm{CG}(\Delta\theta) = \frac{1}{N}\left|\sum_{n=0}^{N-1} w[n]e^{j0.5\Delta\theta n}\right|, \tag{6.58}$$

and the *worst-case processing gain* as

$$\mathrm{PG}(\Delta\theta) = \frac{\left|\sum_{n=0}^{N-1} w[n]e^{j0.5\Delta\theta n}\right|^2}{N\sum_{n=0}^{N-1} w^2[n]}. \tag{6.59}$$

The rule of thumb (6.55) [or (6.57)] should be modified in this case by substituting $\mathrm{PG}(\Delta\theta)$ for PG. We emphasize again that $\Delta\theta$ depends on the algorithm: For simple DFT it is $2\pi/N$, for zero-padded DFT it is $2\pi/L$ (where L is the length after zero padding), and for the chirp Fourier transform it is on the order of $2\pi/N^2$. In the case of the chirp Fourier transform, the loss due to $\Delta\theta$ is usually negligible and $\mathrm{PG}(0)$ can be used for most practical purposes.

Table 6.2 summarizes the various parameters of the windows described in Section 6.3. SLL is the side-lobe level, as defined in Section 6.3. $\mathrm{CG}(0)$ and $\mathrm{CG}(2\pi/N)$ are the best- and worst-case coherent gains, where the worst case corresponds to simple DFT. $\mathrm{PG}(0)$ and $\mathrm{PG}(2\pi/N)$ are the corresponding processing gains. The parameter J_{w}

will be discussed later; for now it should be ignored. The values shown in the table are for large N. For short windows, the values differ slightly.

As we see from Table 6.2, all gains are actually losses, since they are all negative in dB. We must remember, however, that these values are normalized by N and that there is always an additional gain of N, as seen in (6.49) and (6.55). Windows with lower side-lobe levels have smaller best-case gains in general, but the difference between the best case and the worst case is also smaller. The rectangular window is the only one whose best-case gains are 0 dB, but its worst-case values are inferior. The worst-case processing gain of all windows is between -3 and -4 dB. The procedure `cpgains` in Program 6.5 implements the computation of the coherent gain and processing gain, as well as the parameter J_w defined in Section 6.5.2.

Window	SLL	CG(0)	$\mathrm{CG}\left(\frac{2\pi}{N}\right)$	PG(0)	$\mathrm{PG}\left(\frac{2\pi}{N}\right)$	J_w
rectangular	−13.5	0	−3.92	0	−3.92	1.00
Bartlett	−27	−6.01	−7.84	−1.25	−3.07	1.59
Hann	−32	−6.03	−7.45	−1.77	−3.19	2.35
Hamming	−43	−5.36	−7.11	−1.35	−3.10	1.65
Blackman	−57	−7.54	−8.64	−2.38	−3.47	3.15
Kaiser, $\alpha = 4$	−30	−4.39	−6.45	−0.96	−3.02	1.41
Kaiser, $\alpha = 8$	−58	−7.22	−8.40	−2.22	−3.40	2.82
Kaiser, $\alpha = 12$	−90	−8.93	−9.75	−3.03	−3.85	4.70
Dolph, $\alpha = -40$	−40	−4.60	−6.67	−1.15	−3.21	1.79
Dolph, $\alpha = -60$	−60	−6.40	−7.82	−1.81	−3.24	2.16
Dolph, $\alpha = -80$	−80	−7.66	−8.74	−2.41	−3.50	3.13

Table 6.2 Windows and their parameters (all in dB except for J_w).

6.5.2 Frequency Estimation

Let us now discuss the accuracy of the frequency estimated from the maximum point of the Fourier transform when noise is present. To isolate the effect of noise from that of sampling at a finite number of frequencies, we assume that $|X^f(\theta)|$ is computed at arbitrarily high resolution, for example, by the chirp Fourier transform. Let us denote by $\hat{\theta}_0$ the frequency where $|X^f(\theta)|$ is maximum. Then $\hat{\theta}_0$ is close, but not identical, to the true frequency θ_0. The difference $\hat{\theta}_0 - \theta_0$ is a random variable, depending on the noise, the amplitude of the complex exponential, and the number of samples. The derivation of the mean and variance of this random variable relies on mathematical tools beyond the scope of this book, so we give only the final results. These results are valid when the inequality (6.55) [or (6.57)] is satisfied.

For a given window $w[n]$, define

$$J_w = \frac{N^3 \sum_{n=0}^{N-1} [n - 0.5(N-1)]^2 w^2[n]}{12 \left(\sum_{n=0}^{N-1} [n - 0.5(N-1)]^2 w[n] \right)^2}. \tag{6.60}$$

This parameter is nearly independent of N for large N, depending only on the window type. Using this parameter, we have

$$E(\hat{\theta}_0 - \theta_0)^2 \approx \frac{6\gamma_v J_w}{A^2 N^3} = \frac{6J_w}{\mathrm{SNR_i} \cdot N^3}. \tag{6.61}$$

In addition,

$$E(\hat{\theta}_0 - \theta_0) \approx 0. \tag{6.62}$$

The parameter J_w is between 1 and 4 for most practical windows. Its smallest value, 1, is obtained for a rectangular window, that is, for an unwindowed Fourier transform. This case represents the best accuracy (i.e., the smallest mean-square error) that can be obtained for given SNR and number of samples.

When the discrete-time signal $y[n]$ is obtained by antialias filtering and sampling of a continuous-time signal accompanied by broad-band noise, the input SNR is given by (6.56). In this case we are interested in the error in ω, rather than the error in θ. Substituting $\omega = \theta/T$ and $D = NT$, we get from (6.61)

$$E(\hat{\omega}_0 - \omega_0)^2 \approx \frac{6N_0J_w}{A^2D^3}. \tag{6.63}$$

The mean-square error in the frequency f is given by

$$E(\hat{f}_0 - f_0)^2 \approx \frac{6N_0J_w}{(2\pi)^2A^2D^3}. \tag{6.64}$$

Substitution of (6.57) in (6.63) leads to the inequality

$$[E(\hat{f}_0 - f_0)^2]^{1/2} \leq \sqrt{\frac{6J_w \cdot \mathrm{PG}}{100\pi^2}} \cdot \frac{1}{D}. \tag{6.65}$$

The left side of (6.65) represents the root-mean-square (RMS) frequency error in hertz. As we recall from Section 4.1, the quantity $1/D$ (the reciprocal of the measurement interval) represents the basic resolution of the DFT in hertz; see page 99. The numerical factor multiplying $1/D$ in (6.65) is smaller than 1 for all common windows. For example, with PG = 0.5 and $J_w = 4$, this factor is about 0.11. We are thus led to the conclusion that *the RMS frequency error due to noise is smaller than the basic DFT resolution.* Furthermore, if the left side of (6.57) is much larger than 25, this factor will be much smaller than 1. It therefore pays to compute the maximum point of $|X^f(\theta)|$ at a resolution much better than the basic DFT resolution. Zero padding is not practical for this purpose, since it would require a high padding ratio. On the other hand, the chirp Fourier transform is useful. An alternative to the chirp Fourier transform for this purpose is discussed in Problem 6.20.

6.5.3 Detection and Frequency Estimation for Real Sinusoids

Having treated the case of a single complex exponential, let us now proceed to see the effect of noise on a single real sinusoid. The measured signal is now

$$\begin{aligned} y[n] &= s[n] + v[n] = A\cos(\theta_0 n + \phi_0) + v[n] \\ &= 0.5Ae^{j(\theta_0 n+\phi_0)} + 0.5Ae^{-j(\theta_0 n+\phi_0)} + v[n]. \end{aligned} \tag{6.66}$$

Note that the noise $v[n]$ is now real valued. The Fourier transform of the windowed signal $x[n]$ at $\theta = \theta_0$ is

$$X^f(\theta_0) = 0.5NAe^{j\phi_0} \cdot \mathrm{CG} + 0.5A\sum_{n=0}^{N-1} w[n]e^{-j(2\theta_0 n+\phi_0)} + \sum_{n=0}^{N-1} w[n]v[n]e^{-j\theta_0 n}. \tag{6.67}$$

Let us assume that the number of samples N is such that $2\theta_0$ and $2(\pi - \theta_0)$ are both larger than the main-lobe width of $W^f(\theta)$, the kernel function of the window. As we saw in Section 6.4, this guarantees that only a side lobe of the negative spectral component interfere with the positive one. In other words, the second term on the

right side of (6.67) is negligible, resulting in

$$X^{\mathrm{f}}(\theta_0) \approx 0.5NAe^{j\phi_0} \cdot \mathrm{CG} + \sum_{n=0}^{N-1} w[n]v[n]e^{-j\theta_0 n}. \tag{6.68}$$

Comparing this expression with (6.49), we see that the main change is the factor 0.5, resulting from splitting the real sinusoid into two spectral lines, one positive and one negative. The constant phase term $e^{j\phi_0}$ is immaterial, since it disappears when taking the absolute value. Therefore, all results derived for a complex exponential, and the conclusions drawn from them hold for the real case, the only change being the replacement of A by $0.5A$. In particular, formula (6.52) changes to

$$\mathrm{SNR_o} = \frac{(NA \cdot \mathrm{CG})^2}{4\gamma_v \sum_{n=0}^{N-1} w^2[n]}. \tag{6.69}$$

The input SNR of a real sinusoid is defined as

$$\mathrm{SNR_i} = \frac{A^2}{2\gamma_v}. \tag{6.70}$$

Note that, although $(0.5A)^2 = 0.25A^2$, there is a division only by 2 in the definition, since the average power of a sinusoid is half its square amplitude. The processing gain is now defined as

$$\mathrm{PG} = \frac{\mathrm{SNR_o}}{0.5N \cdot \mathrm{SNR_i}}, \tag{6.71}$$

so its expression in terms of the window coefficients (6.54) remains unchanged. The rules of thumb (6.55), (6.57) change to

$$\frac{NA^2 \cdot \mathrm{PG}}{\gamma_v} \geq 100, \quad \frac{DA^2 \cdot \mathrm{PG}}{N_0} \geq 100. \tag{6.72}$$

The frequency error expressions change to

$$E(\hat{\theta}_0 - \theta_0)^2 \approx \frac{24\gamma_v J_{\mathrm{w}}}{A^2 N^3}, \quad E(\hat{\theta}_0 - \theta_0) \approx 0, \tag{6.73a}$$

$$E(\hat{f}_0 - f_0)^2 \approx \frac{24N_0 J_{\mathrm{w}}}{(2\pi)^2 A^2 D^3}, \quad E(\hat{f}_0 - f_0) \approx 0. \tag{6.73b}$$

We emphasize again that these approximations are not as good as the ones in the case of a complex exponential, due to the interference term from the negative spectral line, which we have neglected.

Finally, we consider the frequency estimation problem for the M-sinusoids model (6.37), modified to include additive white noise $v[n]$, that is,

$$y[n] = \sum_{k=1}^{M} A_k \cos(\theta_k n + \phi_k) + v[n], \quad 0 \leq n \leq N-1. \tag{6.74}$$

Assume that the following conditions are satisfied:

1. The frequencies and the amplitudes are such that, for each k, the kth complex exponential $0.5A_k e^{j(\theta_k n + \phi_k)}$ is only weakly disturbed by the other complex exponentials.
2. The inequalities (6.72) hold for all A_k.

In this case, the approximate expressions (6.73) apply to each k individually, that is,

$$E(\hat{\theta}_k - \theta_k)^2 \approx \frac{24\gamma_v J_{\mathrm{w}}}{A_k^2 N^3}, \quad E(\hat{\theta}_k - \theta_k) \approx 0, \tag{6.75a}$$

$$E(\hat{f}_k - f_k)^2 \approx \frac{24N_0 J_{\mathrm{w}}}{(2\pi)^2 A_k^2 D^3}, \quad E(\hat{f}_k - f_k) \approx 0. \tag{6.75b}$$

Great care should be taken, however, in using these approximations, since they tend to be rather crude, especially when the number of sinusoids is large. Unfortunately, accurate analysis of the M-sinusoid problem is difficult and few general results are known. Therefore, it is good practice to perform computer simulations and field tests, and to avoid relying on the approximate formulas exclusively.

Example 6.4 Consider again the discrete-time signal in Example 6.3,

$$y[n] = \sin(2\pi \cdot 0.1992n) + 0.005 \sin(2\pi \cdot 0.25n), \quad 0 \le n \le 127. \tag{6.76}$$

Here we wish to examine the windowed Fourier transform of this signal when measured with white noise. Since the Kaiser window with $\alpha = 10$ was proved to be the best among all windows tested in Example 6.3, we use only this window here. First, we choose the noise standard deviation such that (6.55) will be satisfied with equality for the weaker of the two sinusoids. We have

$$N = 128, \quad A_2 = 0.005, \quad \text{PG} = 0.538,$$

which gives $\gamma_v^{1/2} = 0.00415$. Figure 6.22(a) shows a typical DFT magnitude for these parameters. Comparing this figure with Figure 6.21(c), we see that the DFT floor has been raised and looks noisy. The peak corresponding to the first (strong) sinusoid has not suffered any discernible degradation. The peak corresponding to the second (weak) sinusoid is still the second largest, but is only a few dB above some noise peaks.

Figure 6.22(b) shows what happens when the noise standard deviation is doubled. Now inequality (6.55) is not satisfied any more. As we see, the peak corresponding to the weak sinusoid is no longer the second in magnitude. Thus, the noise masks this sinusoid and renders it undetectable with high probability.

Figure 6.22(c) shows the effect of raising the noise level such that (6.55) holds with equality for the strong sinusoid. The corresponding value of $\gamma_v^{1/2}$ is 0.83. The weak sinusoid is now completely lost in the noise. The strong sinusoid still gives the largest peak, but some noise peaks appear only a few dB below it. Increasing the noise level still further would mask the strong sinusoid as well. □

Example 6.5 We continue Example 6.4 and examine the accuracy of the frequency estimates. The parameter J_w of a Kaiser window with $\alpha = 10$ is 3.79. The RMS of the frequency error $\hat{\theta}_2 - \theta_2$ is, according to (6.73), approximately 5.47×10^{-3}. To confirm this value, we perform a sequence of random simulations. (Simulations of this kind are called *Monte-Carlo simulations*, in honor of Europe's unofficial gambling capital.) Each simulation adds a different noise realization to the signal $y[n]$ (using the MATLAB function `randn`), and then the frequencies $\hat{\theta}_1$, $\hat{\theta}_2$ are found from the largest local maxima of the windowed Fourier transform. After L such simulations have been performed, we compute the *empirical mean error*

$$\langle \hat{\theta}_2 - \theta_2 \rangle = \frac{1}{L} \sum_{l=1}^{L} (\hat{\theta}_{2,l} - \theta_2), \tag{6.77}$$

and the *empirical RMS error*

$$\langle (\hat{\theta}_2 - \theta_2)^2 \rangle^{1/2} = \left[\frac{1}{L} \sum_{l=1}^{L} (\hat{\theta}_{2,l} - \theta_2)^2 \right]^{1/2}, \tag{6.78}$$

where $\{\hat{\theta}_{2,l},\ 1 \le l \le L\}$ are the results of the individual random runs. We do not do the same for $\hat{\theta}_1$, since its accuracy, according to (6.73), is much higher than the accuracy

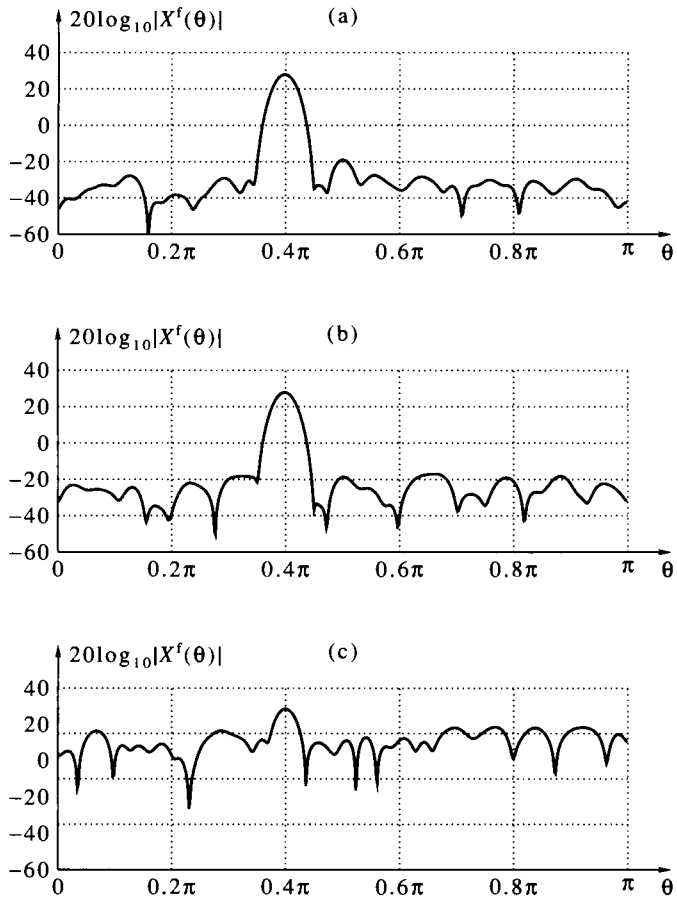

Figure 6.22 Frequency measurement of two sinusoids in noise by Kaiser-windowed DFT: (a) $\gamma_v^{1/2} = 0.00415$; (b) $\gamma_v^{1/2} = 0.0083$; (c) $\gamma_v^{1/2} = 0.83$.

of the chirp Fourier transform algorithm in this case. In this test we use $L = 1000$. The results are:

$$\langle \hat{\theta}_2 - \theta_2 \rangle = -8.64 \times 10^{-4}, \quad \langle (\hat{\theta}_2 - \theta_2)^2 \rangle^{1/2} = 5.27 \times 10^{-3}.$$

As we see, the empirical RMS is close to the theoretical RMS, which is 5.47×10^{-3}, and the empirical mean is smaller than the empirical RMS by almost an order of magnitude. Therefore, the computer simulations are in good agreement with the theoretical formulas in this case.

Next we repeat the same experiment with $\gamma_v^{1/2} = 0.83$. As we saw in Example 6.4, only the frequency θ_1 is of interest in this case. Formula (6.73) predicts an RMS error of 5.47×10^{-3} for $\hat{\theta}_1$ in this case. The empirical mean and RMS errors obtained from the simulations are

$$\langle \hat{\theta}_1 - \theta_1 \rangle = -8.78 \times 10^{-5}, \quad \langle (\hat{\theta}_1 - \theta_1)^2 \rangle^{1/2} = 5.61 \times 10^{-3}.$$

Again, there is good agreement between the empirical and the theoretical RMS error. Moreover, the empirical mean is smaller than the empirical RMS by almost two orders of magnitude. □

6.6 Summary and Complements

6.6.1 Summary

This chapter was devoted to practical aspects of spectral analysis, in particular short-time spectral analysis. Short-time spectral analysis is needed (1) when the data sequence is naturally short or (2) when it is long, but splitting it into short segments and analyzing each segment separately makes more sense physically.

The main tool in short-time spectral analysis is windowing. Windowing is the combined operation of selecting a fixed-length segment of the data and shaping the signal in the segment by multiplication, as expressed by (6.8). The equivalent operation in the frequency domain is convolution with the window's kernel function (its Fourier transform), as expressed by (6.9).

The two main parameters by which we judge the suitability of a window for a given task are the width of the main lobe of its kernel function and the side-lobe level relative to the main lobe. The main lobe acts to smear the Fourier transform of the signal (through frequency-domain convolution); therefore it should be as narrow as possible. The side lobes lead to interference between signal components at different frequencies, so they should be as low as possible. These two requirements contradict each other; therefore, the selection of a window for a particular application involves a trade-off between the two parameters.

The rectangular window has the narrowest main lobe (for a given length of the window), but the highest side lobes. Because of its high side lobes, it is rarely used. Common windows, arranged in a decreasing order of side-lobe level, are Bartlett (6.10), Hann (6.14), Hamming (6.16), and Blackman (6.18). Two window types that enable tuning of the side-lobe level via an additional parameter are the Kaiser window (6.21) and the Dolph window (6.22). The former approximately minimizes the side-lobe energy for a given width of the main lobe. The latter is an equiripple window; its side lobes have a flat, tunable level. Of the two, the Kaiser window is more commonly used.

We have demonstrated the use of spectral analysis for sinusoidal frequency measurement, first without noise and then in the presence of noise. Sinusoidal frequency measurement accuracy depends on the length of the data interval, the separation between the frequencies of the various sinusoids, the relative amplitudes of the sinusoids, and the window used. If noise is present, accuracy also depends on the signal to noise ratio. Special forms of the DFT, such as zero-padded DFT or chirp Fourier transform, can be used for increasing the accuracy.

6.6.2 Complements

1. [p. 164] The notes of the eight chords are shown in the following table; see Section 14.3 for an explanation of the relationships between notes and their frequencies.

Bar No.	1	2	3	4	5	6	7	8
note 1	C	A	E	C	F#	E	B	E
note 2	E	C	G	E	A#	G	D#	G#
note 3	A	F#	B	A	C#	B	F	B
note 4	—	—	—	—	E	—	A	—

2. [p. 186] A discrete-time complex white noise is $v[n] = v_r[n] + jv_i[n]$, where
   (a) Each of $v_r[n]$, $v_i[n]$ is white noise and they have zero means and equal variances.

(b) $v_r[n]$, $v_i[n]$ are uncorrelated, that is, $E(v_r[n]v_i[m]) = 0$ for all n, m.

The covariance sequence of $v[n]$ is defined by

$$\kappa_v[m] = E(v[n+m]\bar{v}[n]) = \gamma_v\delta[m] = (\gamma_{v_r} + \gamma_{v_i})\delta[m].$$

3. [p. 188] It is common to measure the reliability of signal detection by two criteria:
   (a) The probability that a nonexistent signal will be falsely detected. This is called the *false alarm probability* and is denoted by P_{fa}.
   (b) The probability that an existing signal will not be detected. This is called the *miss probability* and is denoted by P_{miss}. Its complement, $P_{det} = 1 - P_{miss}$, is called the *detection probability*.

   These two criteria always conflict with each other; decreasing the probability of false alarm increases the probability of miss and vice versa. Quantitative analysis of these probabilities for the problem of detecting sinusoids in noise is beyond the level of this book; see, for example, Van Trees [1968].

6.7 MATLAB Programs

Program 6.1 Generation of the seven common window types.

```
function  w = window(N,name,alpha);
% Synopsis: w = window(N,name,alpha).
% Generates a window.
% input parameters:
% N: the window length
% name: the window name inside quotes, as follows:
%       'rect' for rectangular
%       'bart' for Bartlett
%       'hann' for Hann
%       'hamm' for Hamming
%       'blac' for Blackman
%       'kais' for Kaiser
%       'dolp' for Dolph
% alpha: the Kaiser window parameter for name = 'kais',
%        the side-lobe level (in dB) for name = 'dolp',
%        not used otherwise.
% Output:
% w: the window (a row vector)

n = 0:N-1;
if (name(1:4) == 'rect'),
   w = ones(1,N);
elseif (name(1:4) == 'bart'),
   w = 1-(1/(N+1))*abs(2*n-N+1);
elseif (name(1:4) == 'hann'),
   w = 0.5*(1-cos((2*pi/(N-1))*n));
elseif (name(1:4) == 'hamm'),
   w = 0.54-0.46*cos((2*pi/(N-1))*n);
elseif (name(1:4) == 'blac'),
   w = 0.42-0.5*cos((2*pi/(N-1))*n) + 0.08*cos((4*pi/(N-1))*n);
elseif (name(1:4) == 'kais'),
   w = 1/(besseli(0,alpha))* ...
       besseli(0,alpha*sqrt(1-((1/(N-1))*abs(2*n-N+1)).^2));
elseif (name(1:4) == 'dolp'),
   w = dolph(N,alpha);
else,
   error('Unrecognized window name in WINDOW')
end
```

Program 6.2 Implementation of Dolph window.

```
function w = dolph(N,sll);
% Synopsis: w = dolph(N,sll).
% Computes a Dolph-Chebyshev window.
% Input parameters:
% N: the window length
% sll: the side-lobe level, in dB (negative number)
% Output:
% w: the window.

b = 10^(-sll/20); b = cosh(log(b+sqrt(b^2-1))/(N-1));
c = b*cos((pi/N)*(0:N-1)); ind1 = find(abs(c) <= 1);
ind2 = find(abs(c) > 1); w = zeros(1,N);
w(ind1) = cos((N-1)*acos(c(ind1)));
w(ind2) = cosh((N-1)*log(c(ind2)+sqrt(c(ind2).^2-1))); w = real(w);
if (rem(N,2) == 0), w = w.*exp((j*pi/N)*(0:N-1)); end
w = fftshift(real(ifft(w))); w = (1/w(floor(N+1)/2))*w;
```

Program 6.3 Computation of the M largest maxima of the DFT of a real vector.

```
function [theta,val] = maxdft(x,M,name,alpha);
% Synopsis: [theta,val] = maxdft(x,M,name,alpha).
% Finds the M largest maxima of the DFT of the real vector x.
% Input parameters:
% x: the input vector
% M: number of local maxima to be found
% name: an optional window for x; one of the names in window.m
% alpha: needed if name = 'kaiser'.
% Output parameters:
% theta: vector of thetas at the local maxima
% val: vector of corresponding values of DFT(x).

N = length(x); x = reshape(x,1,N);
if (nargin == 3), x = x.*window(N,name);
elseif (nargin == 4), x = x.*window(N,name,alpha); end
X = abs(fft(x)); X = X(1:floor((N+1)/2));
[y,ind] = locmax(X); ind = ind(1:M);
theta = (2*pi/N)*(ind-1); val = zeros(1,M);
for m = 1:M,
   X = chirpf(x,theta(m)-(2*pi/N),(2*pi/N^2),2*N+1);
   [y,ind] = max(abs(X));
   theta(m) = theta(m)-(2*pi/N) + (ind-1)*2*pi/N^2;
   val(m) = y;
end
```

Program 6.4 Search for the local maxima of a vector and their indices.

```
function [y,ind] = locmax(x);
% Synopsis: [y,ind] = locmax(x).
% Finds all local maxima in a vector and their locations,
% sorted in decreasing maxima values.
% Input:
% x: the input vector.
% Output parameters:
% y: the vector of local maxima values
% ind: the corresponding vector of indices of the input vector x.

n = length(x); x = reshape(x,1,n);
xd = x(2:n)-x(1:n-1);
i = find(xd(1:n-2) > 0.0 & xd(2:n-1) < 0.0) + 1;
if (x(1) > x(2)), i = [1,i]; end
if (x(n) > x(n-1)), i = [i,n]; end
[y,ind] = sort(x(i)); ind = fliplr(ind);
ind = i(ind); y = x(ind);
```

Program 6.5 The coherent gain and processing gain of a window.

```
function [cg,pg,jw] = cpgains(w,dtheta);
% Synopsis: [cg,pg,jw] = cpgains(w,dtheta).
% Computes the coherent gain and the processing gain of
% a given window as a function of the frequency deviation.
% Also computes the parameter Jw (see text).
% Input parameters:
% w: the window sequence (of length N)
% dtheta: the frequency deviation:
%   0 gives the best-case gains
%   2*pi/N gives the worst case for N-point DFT
%   2*pi/M gives the worst case for M-point zero-padded DFT
% Output parameters:
% cg: the coherent gain, in dB
% pg: the processing gain, in dB
% jw: the parameter Jw.

N = length(w); w = reshape(w,1,N);
cg = (1/N)*abs(sum(w.*exp(j*0.5*dtheta*(0:N-1))));
pg = N*cg^2/sum(w.*w);
cg = 20*log10(cg); pg = 10*log10(pg);
n = (0:N-1)-0.5*(N-1);
jw = (N^3/12)*(sum((w.*n).^2)/(sum(w.*(n.^2)))^2);
```

6.8 Problems

6.1 We have seen that the largest side lobe of the rectangular window has attenuation -13.5 dB with respect to the main lobe. What is approximately the attenuation of the smallest side lobe? Note that the result depends on N.

6.2 Let N be even. Show that the convolution of a rectangular window of length $N/2$ and a rectangular window of length $(N+2)/2$ gives the Bartlett window (6.10) for even-length N, up to a constant scale factor. Then show that the kernel function of this window is (6.12).

6.3 A window $w[n]$ is generated by convolving a rectangular window of length N with a Bartlett window of length $2N-1$. What are the length, main lobe width, and side-lobe level of $w[n]$?

6.4 Recall that the Bartlett window was obtained by convolving a rectangular window $w_r[n]$ with itself. Suppose that we generate a window $w[n]$ by K-fold convolution of $w_r[n]$ with itself. Describe the properties of $w[n]$.

6.5 Let $x[n]$ be a discrete-time signal on $0 \le n \le N-1$ and $X^d[k]$ its DFT. Define

$$Y_1^d[k] = 0.5X^d[k] - 0.25X^d[(k-1) \bmod N] - 0.25X^d[(k+1) \bmod N]. \quad (6.79)$$

(a) Show that $Y_1^d[k]$ represents a windowing operation in the time domain. How is this window related to the Hann window? Hint: Use (4.32).

(b) Define

$$Y_2^d[k] = 0.42X^d[k] - 0.25X^d[(k-1) \bmod N] - 0.25X^d[(k+1) \bmod N] + 0.04X^d[(k-2) \bmod N] + 0.04X^d[(k+2) \bmod N]. \quad (6.80)$$

Show that $Y_2^d[k]$ represents a windowing operation in the time domain. To which of the windows you have learned is this window related?

(c) Discuss possible implementation advantages of the windows in parts a and b.

(d) Can the idea proposed in this problem be applied to other windows you know?

6.6 Consider the window

$$w[n] = \sin^4\left(\frac{\pi n}{N-1}\right).$$

Explore the properties of this window: main-lobe width and side-lobe level. How is this window related to the Hann window?

6.7 We are given 128 samples of the signal

$$x[n] = \sin\left(2\pi\frac{6.3}{128}n\right) + 0.001\sin\left(2\pi\frac{56}{128}n\right).$$

(a) Explain why a rectangular window is not adequate for detecting the second component. Hint: Problem 6.1 is relevant here.

(b) Of the Hann and Hamming windows, guess which one is better in this case for detecting the second component, then test your guess on a computer.

6.8 Change the coefficients $0.42, 0.5, 0.08$ in the Blackman window to 0.42323, 0.49755, 0.07922, respectively. Using MATLAB, find the side-lobe level of this window. This is called the *Blackman–Harris window*; see Harris [1978].

6.9 When using windowing *and* zero padding for a length-N sequence $x[n]$, which of the following two is the proper procedure: (1) zero-pad first to length M, then use a window of length M or (2) use a length-N window on $x[n]$, then zero-pad to length M? Give reasons. If you are in doubt, experiment with MATLAB.

6.10 A communication system receives 2000 samples of the signal $\cos(\theta_0 n + \phi_0)$. For certain reasons, however, the signal is chopped in the following manner: After every 80 signal points there are 20 erased points, that is, points containing zeros. In MATLAB, such a signal is generated by the command

```
x = cos(theta0*(0:1999)+phi0).* ...
    kron(ones(1,20),[ones(1,80), zeros(1,20)]);
```

It is required to measure the frequency θ_0, using windowed DFT. Consider the following windowing options:

(a) A single window of length 2000 applied to the entire data sequence.

(b) 20 windows of length 80 each, applied to the individual chops.

(c) Both a and b.

Explore these three options with MATLAB. Decide which is the proper one to use and give reasons.

6.11 The signal $x(t) = \cos(2\pi t)$ is sampled at interval $T = 0.01$ second, and $N = 100$ samples are collected. The frequency of the signal is then measured from a windowed DFT (without zero padding). Of the windows you have studied, which ones will show that the signal has nonzero frequency and which will mislead us to think that the frequency is zero? For the Kaiser window, the answer depends on the parameter α. Answer first without trying it out; then verify your answers with MATLAB. Finally, repeat the experiment with $N = 200$.

6.12 This problem introduces another measure of performance of windows, called *roll-off rate*. The roll-off rate measures how fast the side lobes of the window decay as $|\theta|$ increases. It can be shown that, for large values of N, the peaks of the magnitude response $|W^{\mathrm{f}}(\theta)|$ of a window are approximately tangent to a curve of the form

$$A(\theta) = \frac{K}{|\theta|^r},$$

where r is an integer. The larger r, the faster the decay of the side lobes. The quantity $6r$ is called the roll-off rate, and is measured in dB/octave. The reason for this definition is that

$$20\log_{10} A(\theta) = 20\log_{10} K - 20r\log_{10}|\theta|;$$

therefore, $20\log_{10} A(\theta)$ decreases by $6r$ dB each time $|\theta|$ is doubled. A frequency is said to be an *octave* higher than another frequency if it is two times higher. This term comes from music, where the interval between two notes bearing the same name is an integer number of octaves; that is, the frequency ratio is an integer power of 2.

A convenient way of visualizing the roll-off rate is to plot $|W^{\mathrm{f}}(\theta)|$ and $A(\theta)$ on a log-log scale. Then $A(\theta)$ appears as a straight line tangent to the peaks of $|W^{\mathrm{f}}(\theta)|$. In the following exercises, use windows of length $N = 128$, zero-pad to 1024 for plotting, and use the MATLAB command `loglog` for plotting.

(a) For a rectangular window, $K = 2$ and $r = 1$. Therefore, the roll-off rate is 6 dB/octave. Convince yourself that this is so by plotting $|W^{\mathrm{f}}(\theta)|$ and $A(\theta)$.

(b) Guess the roll-off rate of the Bartlett window, then confirm your guess by examining the plots.

(c) The roll-off rate of the Hann window is 18 dB/octave. Convince yourself that this is so by plotting $|W^{\mathrm{f}}(\theta)|$ and $A(\theta)$. Find K by experimenting.

(d) Guess the roll-off rate of the Hamming window, then confirm your guess by examining its plot.

(e) Find the roll-off rates of the Blackman and Kaiser windows by experimenting with their plots.

(f) What is the roll-off rate of the Dolph window? Answer without relying on a computer.

6.13 Discuss whether windowing is needed for frequency measurement of a *single* real sinusoid.

6.14* Recall the procedure of signal interpolation by zero padding in the frequency domain, studied in Section 4.5 and implemented in Problem 4.41. Suggest how to use windowing for improving the appearance of the interpolated signal, modify the program you have written for Problem 4.41, experiment, and report the results.

6.15* A student who has studied Section 6.5 proposed the following idea: "We know that, since $J_{\mathrm{w}} > 1$ for all windows except the rectangular, the RMS errors of the frequency estimates $\hat{\theta}_m$ are larger for any window than for a rectangular window. We cannot work with a rectangular window exclusively, because of the problem of cross-interference. However, we can have the best of both worlds if we use different windows for the two estimation steps. During the coarse phase we will use a window with low side-lobe level, so that we can reliably identify the spectral peaks. Then, when performing the fine step of frequency estimation, we will revert to a rectangular window to get the best possible accuracy."

What is wrong with this idea?

6.16* A student who has studied Sections 6.4 and 6.5 proposed the following idea: "We know that any window other than the rectangular widens the main lobe by a factor of 2 at least. Therefore, when using a windowed DFT for sinusoid detection, we do not need all N frequency points, but we can do equally well with the $N/2$ even-index points. Since the main lobe of the window is at least $\pm 4\pi/N$, the sinusoid will show in at least one of the even-index DFT points. The even-index points can be implemented with *one* DFT of length $N/2$, by slightly modifying the radix-2 frequency-decimated FFT algorithm. This will save about half the number of operations, without degrading our ability to detect the sinusoidal signal."

(a) Explain the proposed modification of the radix-2 frequency-decimated FFT algorithm.

(b) What is wrong with this idea? Hint: Examine the worst-case processing gain of the windows you have studied under the new conditions.

6.17* An NRZ signal $x(t)$ (recall its description in Example 3.13) having bit interval $T = 5 \times 10^{-4}$ second [see (3.60) for the definition of T] is modulated by a complex carrier $e^{j2\pi f_0 t}$ to give the complex signal

$$y(t) = x(t)e^{j2\pi f_0 t}.$$

The carrier frequency f_0 is known to be in the range $|f_0| < 1000\,\text{Hz}$. It is required to estimate f_0 from a finite measured interval of the modulated BPSK signal.

(a) Explain why the measurement of f_0 from the magnitude of the DFT of $y(t)$ (with or without a window) is likely to be problematic. If you are in doubt, experiment with MATLAB.

(b) It is proposed to estimate f_0 by the following procedure:

i. Sample the signal and square the result; that is, compute

$$z[n] = y^2(nT).$$

ii. Estimate f_0 from the magnitude of the DFT of $z[n]$.

Explain why this method works and determine the sampling interval necessary for unambiguous measurement of f_0.

(c) Suppose that discrete-time white noise $v[n]$ is added to $y(nT)$. Let γ_v be the variance of the white noise. Assume that a rectangular window is used. Compute the quantity SNR_0 which is, by definition, the ratio of the signal power to the noise power at the frequency of maximum of the magnitude of the DFT. Assume that the input SNR is relatively high, so the term $v^2[n]$ can be neglected in the expansion of $z[n]$.

(d) Draw conclusions from part c about the loss in SNR due to the squaring operation used in part b.

The method presented in this problem is useful for carrier frequency estimation of BPSK signals but is limited to cases of relatively high input SNR, for the reasons you have discovered in parts c and d.

6.18* We wish to detect the presence of a single complex exponential in white noise using windowed DFT, as discussed in Section 6.5.1. Suppose we know a priori that the frequency θ_0 of the complex exponential is less in magnitude than a certain θ_c, where $0 < \theta_c < \pi$. It is proposed to filter the signal $y[n]$ by an ideal low-pass filter having frequency response

$$H^f(\theta) = \text{rect}\left(\frac{\theta}{2\theta_c}\right)$$

and use the filtered signal for the windowed DFT. The variance of the noise will be reduced by this filter, whereas the complex exponential will not be affected. Therefore, it is argued that the processing gain of the windowed DFT will be increased.

(a) Show that the noise term in $E(|X^f(\theta_0)|^2)$ is given by

$$\frac{\gamma_v}{2\pi}\int_{-\theta_c-\theta_0}^{\theta_c-\theta_0} |W^f(\theta)|^2 d\theta,$$

where $W^f(\theta)$ is the kernel function of the window. Hint: The second term on the third line of (6.51) changes to

$$\sum_{n=0}^{N-1}\sum_{m=0}^{N-1} w[n]w[m]\kappa[n-m]e^{-j\theta_0(n-m)}, \qquad (*)$$

where $\kappa[n-m]$ is the covariance sequence of the discrete-time noise at the output of the low-pass filter. Use (2.135) to express $\kappa[n-m]$ as

$$\kappa[n-m] = \frac{\gamma_v}{2\pi}\int_{-\theta_c}^{\theta_c} e^{j\theta(n-m)} d\theta.$$

Substitute this expression in $(*)$ and interchange the order of the integration and the two summations.

(b) Show that $\theta_c = \pi$ gives the result in (6.51).

(c) Explain why, if the window is good and $|\theta_0|$ is not close to θ_c, the proposed idea helps little in increasing the processing gain of the windowed DFT.

6.19* This problem discusses the effect of *sampling jitter* on the spectrum of a sampled signal. Sampling jitter is the name given to random perturbations of the sampling instants. Ideal uniform sampling is defined by the sequence of sampling instants $t[n] = nT$. Sampling jitter is characterized by a sequence of sampling instants $t[n] = nT + \varepsilon[n]$, where $\varepsilon[n]$ is a sequence of random variables resulting from the nonideal operation of the sampling hardware. Here we illustrate the effect of sampling jitter on a sinusoidal signal $x(t) = \cos(\omega_0 t)$.

(a) Choose $\omega_0 = 0.5\pi$; generate 512 samples of $x(t)$, sampled uniformly at $T = 1$. Compute the windowed DFT of the sampled signal, using the Blackman window, and plot the magnitude response in dB.

(b) Generate a sequence of 512 uniformly distributed random variables in the range $\pm\varepsilon_{\max}$, where $\varepsilon_{\max} = 0.1$, and use it for generating the sequence of sampling instants $t[n]$. Now generate the sampled sequence $x(t[n])$, compute its windowed DFT using the Blackman window, and superimpose the magnitude response on the one in part a. Repeat this experiment for $\varepsilon_{\max} = 0.01, 0.001$. Report what you have observed.

(c) Analyze the jitter effect in mathematical terms. Hint: Assume that $|\omega_0 \varepsilon[n]| \ll 1$ and expand $x(nT + \varepsilon[n])$ in first-order Taylor series around nT.

6.20* The following procedure for estimating the local peaks of a windowed Fourier transform is a useful alternative to the chirp Fourier transform algorithm:

(a) Find the index k_0 of the windowed DFT (without zero padding) that corresponds to the local peak of interest.

(b) Compute $20\log_{10}|X^{\mathrm{f}}(\theta)|$ at the three points

$$\theta_{\mathrm{a}} = \frac{2\pi(k_0 - 1)}{N}, \quad \theta_{\mathrm{b}} = \frac{2\pi k_0}{N}, \quad \theta_{\mathrm{c}} = \frac{2\pi(k_0 + 1)}{N}.$$

(c) Find a parabola that passes through the three points

$$\{(\theta_{\mathrm{x}}, 20\log_{10}|X^{\mathrm{f}}(\theta_{\mathrm{x}})|),\ \mathrm{x} = \mathrm{a}, \mathrm{b}, \mathrm{c}\}$$

(there is always a unique such parabola).

(d) Take $\hat{\theta}$ as the maximum point of the parabola.

The advantage of this procedure is in that it requires only a few operations. Its disadvantage is in that it is likely to give less accurate results than the chirp Fourier transform.

(a) Derive a closed-form formula for $\hat{\theta}$ in terms of the three points (abscissas and ordinates). Hint: Derive the coefficients of the parabola first, then the maximum point as a function of the coefficients.

(b) Write a MATLAB procedure to implement the computation.

(c) Change the MATLAB program `maxdft` such that it will use this procedure instead of `chirpf`.

(d) Repeat Example 6.4 with this procedure.