CHAPTER

DISCRETE FOURIER TRANSFORM AND SIGNAL SPECTRUM

4

CHAPTER OUTLINE

4.1 DISCRETE FOURIER TRANSFORM

In time domain, representation of digital signals describes the signal amplitude vs. the sampling time instant or the sample number. However, in some applications, the signal frequency content is very useful than the digital signal samples. The representation of the digital signal in terms of its frequency component in a frequency domain, that is, the signal spectrum, needs to be developed. As an example shown in Fig. 4.1, the top plot displays 32 signal samples (vertical bars with circles) from sampling a continuous 1000-Hz sinusoid (dashed line) at a sampling rate of 8000 Hz in time domain representation; the bottom plot shows the magnitude of signal spectrum in frequency domain representation, where we can clearly observe that the amplitude peak is located at the frequency of 1000 Hz in the calculated spectrum. Hence, the spectral plot better displays frequency information of a digital signal.

The algorithm transforming the time domain signal samples to the frequency domain components is known as the *discrete Fourier transform*, or DFT. The DFT also establishes a relationship between the time domain representation and the frequency domain representation. Therefore, we can apply the DFT to perform frequency analysis of a time domain sequence. In addition, the DFT is widely used in many other areas, including spectral analysis, acoustics, imaging/video, audio, instrumentation, and communications systems.

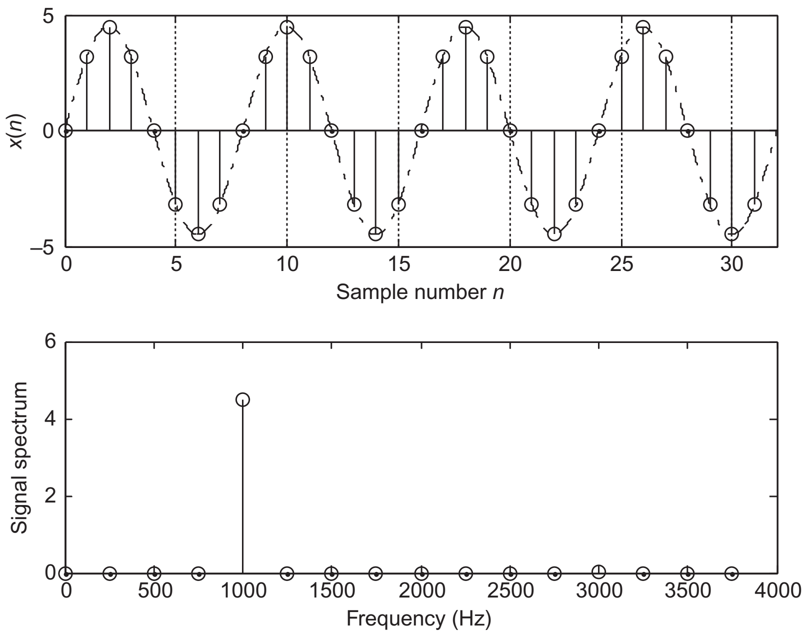

FIG. 4.1

Example of the digital signal and its amplitude spectrum.

To be able to develop the DFT and understand how to use it, we first study the spectrum of periodic digital signals using the Fourier series. (Detailed discussion on Fourier series is provided in Appendix B).

4.1.1 FOURIER SERIES COEFFICIENTS OF PERIODIC DIGITAL SIGNALS

Let us look at a process in which we want to estimate the spectrum of a periodic digital signal $x(n)$ sampled at a rate of f_s Hz with the fundamental period $T_0 = NT$, as shown in Fig. 4.2, where there are N samples within the duration of the fundamental period and $T = 1/f_s$ is the sampling period. For the time being, we assume that the periodic digital signal is band limited to have all harmonic frequencies less than the folding frequency $f_s/2$ so that aliasing does not occur.

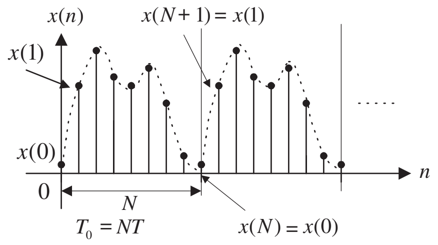

FIG. 4.2

Periodic digital signal.

According to Fourier series analysis (Appendix B), the coefficients of the Fourier series expansion of the periodic signal $x(t)$ in a complex form is

$$c_k = \frac{1}{T_0}\int_{T_0} x(t)e^{-jk\omega_0 t}dt, \text{ for } -\infty < k < \infty, \tag{4.1}$$

where k is the number of harmonics corresponding to the harmonic frequency of kf_0, and $\omega_0 = 2\pi/T_0$ and $f_0 = 1/T_0$ are the fundamental frequency in radians per second and the fundamental frequency in Hz, respectively. To apply Eq. (4.1), we substitute $T_0 = NT$, $\omega_0 = 2\pi/T_0$ and approximate the integration over one period using a summation by substituting $dt = T$ and $t = nT$. We obtain

$$c_k = \frac{1}{N}\sum_{n=0}^{N-1} x(n)e^{-j2\pi kn/N}, \text{ for } -\infty < k < \infty. \tag{4.2}$$

We refer c_k as discrete-time Fourier series coefficients. Since the coefficients c_k are obtained from the Fourier series expansion in the complex form, the resultant spectrum c_k will have two sides. There is an important feature of Eq. (4.2) in which the Fourier series coefficient c_k is periodic of N. We can verify this as follows:

$$c_{k+N} = \frac{1}{N}\sum_{n=0}^{N-1} x(n)e^{-j2\pi(k+N)n/N} = \frac{1}{N}\sum_{n=0}^{N-1} x(n)e^{-j2\pi kn/N}e^{-j2\pi n}. \tag{4.3}$$

Since $e^{-j2\pi n} = \cos(2\pi n) - j\sin(2\pi n) = 1$, it follows that

$$c_{k+N} = c_k. \tag{4.4}$$

Therefore, the two-sided line amplitude spectrum $|c_k|$ is periodic, as shown in Fig. 4.3.

We note the following points:

(a) As displayed in Fig. 4.3, only the line spectral portion between the frequencies $-f_s/2$ and $f_s/2$ (folding frequency) represents frequency information of the periodic signal.

(b) Note that the spectral portion from $f_s/2$ to f_s is a copy of the spectrum in the negative frequency range from $-f_s/2$ to 0 Hz due to the spectrum being periodic for every Nf_0 Hz. Again, the amplitude spectral components indexed from $f_s/2$ to f_s can be folded at the folding frequency $f_s/2$ to match the

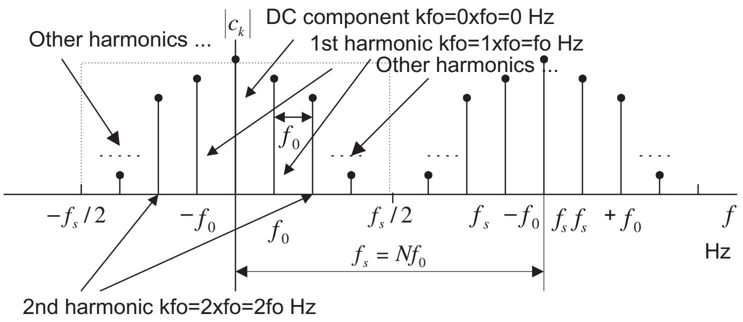

FIG. 4.3

Amplitude spectrum of the periodic digital signal.

amplitude spectral components indexed from 0 to $f_s/2$ in terms of $f_s - f$ Hz, where f is in the range from $f_s/2$ to f_s. For convenience, we compute the spectrum over the range from 0 to f_s Hz with nonnegative indices, that is,

$$c_k = \frac{1}{N}\sum_{n=0}^{N-1} x(n)e^{-j2\pi kn/N},\ k = 0, 1, \ldots, N-1. \tag{4.5}$$

We can apply Eq. (4.4) to find the negative indexed spectral values if they are required.

(c) For the kth harmonic, the frequency is

$$f = kf_0 \text{ Hz}. \tag{4.6}$$

The frequency spacing between the consecutive spectral lines, called the frequency resolution, is f_0 Hz.

EXAMPLE 4.1

The periodic signal

$$x(t) = \sin(2\pi t)$$

is sampled using the sampling rate $f_s = 4$ Hz.

(a) Compute the spectrum c_k using the samples in one period.
(b) Plot the two-sided amplitude spectrum $|c_k|$ over the range from -2 to 2 Hz.

Solution:

(a) From the analog signal, we can determine the fundamental frequency $\omega_0 = 2\pi$ rad/s and $f_0 = \frac{\omega_0}{2\pi} = \frac{2\pi}{2\pi} = 1$ Hz, and the fundamental period $T_0 = 1$ s.

Since using the sampling interval $T = 1/f_s = 0.25$ s, we get the sampled signal as

$$x(n) = x(nT) = \sin(2\pi nT) = \sin(0.5\pi n)$$

and plot the first eight samples as shown in Fig. 4.4.

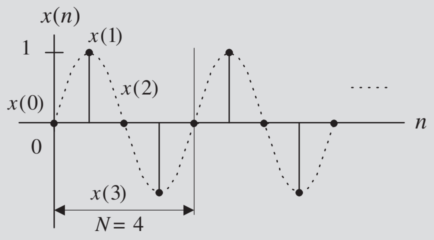

FIG. 4.4

Periodic digital signal.

Choosing the duration of one period, $N = 4$, we have the sample values as follows $x(0) = 0$; $x(1) = 1$; $x(2) = 0$; and $x(3) = -1$.

Using Eq. (4.5),

$$c_0 = \frac{1}{4}\sum_{n=0}^{3} x(n)e^{-j2\pi\times 0n/4} = \frac{1}{4}(x(0)+x(1)+x(2)+x(3))$$

$$= \frac{1}{4}(0+1+0-1) \qquad = 0$$

$$c_1 = \frac{1}{4}\sum_{k=0}^{3} x(n)e^{-j2\pi\times 1n/4} = \frac{1}{4}\left(x(0)+x(1)e^{-j\pi/2}+x(2)e^{-j\pi}+x(3)e^{-j3\pi/2}\right)$$

$$= \frac{1}{4}(x(0)-jx(1)-x(2)+jx(3) = 0-j(1)-0+j(-1)) = -0.5j$$

Similarly, we get

$$c_2 = \frac{1}{4}\sum_{n=0}^{3} x(n)e^{-j2\pi\times 2n/4} = \frac{1}{4}\left(x(0)+x(1)e^{-j2\pi/2}+x(2)e^{-j2\pi}+x(3)e^{-j6\pi/2}\right)$$

$$= \frac{1}{4}(x(0)-x(1)+x(2)-x(3)) = \frac{1}{4}(0-1+0-(-1)) = 0$$

$$c_3 = \frac{1}{4}\sum_{n=0}^{3} x(n)e^{-j2\pi\times 4n/4} = j0.5$$

Using periodicity, it follows that

$$c_{-1} = c_3 = j0.5 \text{ and } c_{-2} = c_2 = 0$$

(b) The amplitude spectrum for the digital signal is sketched in Fig. 4.5.

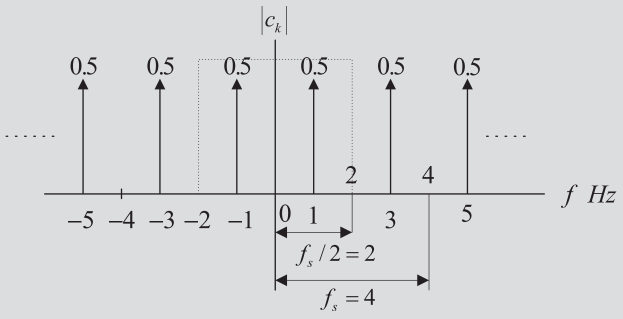

FIG. 4.5

Two-sided spectrum for the periodic digital signal in Example 4.1.

As we know, the spectrum in the range of –2 to 2 Hz presents the information of the sinusoid with a frequency of 1 Hz and a peak value of $2|c_1| = 1$, which is converted from two sided to one sided by doubling the spectral value. Note that we do not double the direct-current (DC) component, that is, c_0.

4.1.2 DISCRETE FOURIER TRANSFORM FORMULAS

Now, let us concentrate on the development of the DFT. Fig. 4.6 shows one way to obtain the DFT formula.

First, we assume that the process acquires data samples from digitizing the interested continuous signal for a duration of T_0 seconds. Next, we assume that a periodic signal $x(n)$ is obtained by copying the acquired N data samples with the duration of T_0 to itself repetitively. Note that we assume continuity between the N data sample frames. This is not true in practice. We will tackle this problem in Section 4.3. Finally, we determine the Fourier series coefficients using one-period N data samples and Eq. (4.5). Then we multiply the Fourier series coefficients by a factor of N to obtain

$$X(k) = Nc_k = \sum_{n=0}^{N-1} x(n)e^{-j2\pi kn/N}, \quad \text{for} \quad k = 0, 1, \ldots, N-1, \tag{4.7}$$

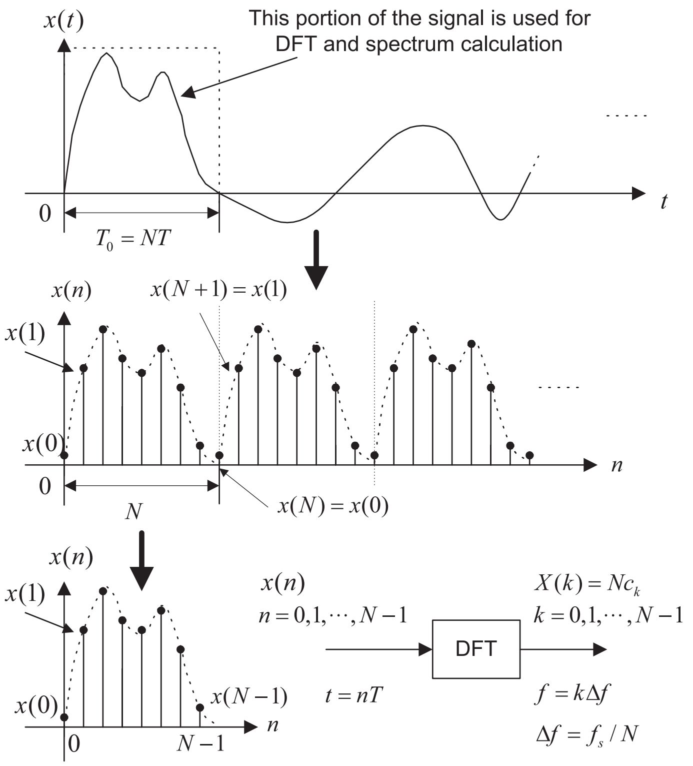

FIG. 4.6

Development of DFT formula.

where $X(k)$ constitutes the DFT coefficients. Note that the factor of N is a constant and does not affect the relative magnitudes of the DFT coefficients $X(k)$. As shown in the last plot, applying DFT with N data samples of $x(n)$ sampled at a sampling rate of f_s (sampling period is $T=1/f_s$) produces N complex DFT coefficients $X(k)$. The index n is the time index representing the sample number of the digital sequence, whereas k is the frequency index indicating each calculated DFT coefficient, and can be further mapped to the corresponding signal frequency in terms of Hz.

Now let us conclude the DFT definition. Given a sequence $x(n), 0 \le n \le N-1$, its DFT is defined as

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi kn/N} = \sum_{n=0}^{N-1} x(n) W_N^{kn}, \text{ for } k = 0, 1, \cdots, N-1. \tag{4.8}$$

Eq. (4.8) can be expanded as

$$X(k) = x(0)W_N^{k0} + x(1)W_N^{k1} + x(2)W_N^{k2} + \cdots + x(N-1)W_N^{k(N-1)}, \text{ for } k = 0, 1, \cdots, N-1, \tag{4.9}$$

where the factor W_N (called the twiddle factor in some textbooks) is defined as

$$W_N = e^{-j2\pi/N} = \cos\left(\frac{2\pi}{N}\right) - j\sin\left(\frac{2\pi}{N}\right). \tag{4.10}$$

To determine the inverse of DFT, we can multiply both sides of Eq. (4.8) by $e^{j2\pi km/N}$ and sum from $k=0$ to $k=N-1$, that is

$$\sum_{k=0}^{N-1} X(k) e^{j\frac{2\pi km}{N}} = \sum_{k=0}^{N-1}\sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi kn}{N}} e^{j\frac{2\pi km}{N}} = \sum_{n=0}^{N-1} x(n) \sum_{k=0}^{N-1} e^{j\frac{2\pi k(m-n)}{N}}. \tag{4.11}$$

Note that

$$\sum_{k=0}^{N-1} e^{j\frac{2\pi k(m-n)}{N}} = \begin{cases} N & m=n \\ 0 & m \ne n \end{cases}. \tag{4.12}$$

With considering the following fact:

$$\sum_{k=0}^{N-1} e^{j\frac{2\pi k(m-n)}{N}} = \sum_{k=0}^{N-1} \left(e^{j\frac{2\pi(m-n)}{N}}\right)^k = \frac{1-e^{j2\pi(m-n)}}{1-e^{j\frac{2\pi(m-n)}{N}}} = \frac{1-1}{1-e^{j\frac{2\pi(m-n)}{N}}} = 0.$$

Thus, Eq. (4.11) becomes

$$\sum_{k=0}^{N-1} X(k) e^{j\frac{2\pi km}{N}} = \sum_{n=0}^{N-1} x(n) \sum_{k=0}^{N-1} e^{j\frac{2\pi k(m-n)}{N}}. \tag{4.13}$$

Using Eq. (4.13), we see that

$$\text{for } m=0, \sum_{k=0}^{N-1} X(k)e^{j\frac{2\pi k0}{N}} = \sum_{n=0}^{N-1} x(n)\sum_{k=0}^{N-1} e^{j\frac{2\pi k(0-n)}{N}} = x(0)\sum_{k=0}^{N-1} 1 = Nx(0)$$

$$\text{for } m=1, \sum_{k=0}^{N-1} X(k)e^{j\frac{2\pi k1}{N}} = \sum_{n=0}^{N-1} x(n)\sum_{k=0}^{N-1} e^{j\frac{2\pi k(1-n)}{N}} = x(1)\sum_{k=0}^{N-1} 1 = Nx(1)$$

…

$$\text{for } m=N-1, \sum_{k=0}^{N-1} X(k)e^{j\frac{2\pi k(N-1)}{N}} = \sum_{n=0}^{N-1} x(n)\sum_{k=0}^{N-1} e^{j\frac{2\pi k(N-1-n)}{N}} = x(N-1)\sum_{k=0}^{N-1} 1 = Nx(N-1)$$

As a summary, we conclude

$$x(m) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{j\frac{2\pi km}{N}}. \tag{4.14}$$

The inverse of DFT is given by

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{j2\pi kn/N} = \frac{1}{N}\sum_{k=0}^{N-1} X(k)W_N^{-kn}, \quad \text{for } n=0,1,\cdots,N-1. \tag{4.15}$$

Proof can also be found in Ahmed and Natarajan (1983); Proakis and Manolakis (1996); Oppenheim et al. (1998); and Stearns and Hush (1990).

Similar to Eq. (4.8), the expansion of Eq. (4.15) leads to

$$x(n) = \frac{1}{N}\left(X(0)W_N^{-0n} + X(1)W_N^{-1n} + X(2)W_N^{-2n} + \cdots + X(N-1)W_N^{-(N-1)n}\right), \quad \text{for } n=0,1,\cdots,N-1. \tag{4.16}$$

As shown in Fig. 4.6, in time domain we use the sample number or time index n for indexing the digital sample sequence $x(n)$. However, in frequency domain, we use index k for indexing N calculated DFT coefficients $X(k)$. We also refer k as the frequency bin number in Eqs. (4.8) and (4.9).

We can use MATLAB functions **fft()** and **ifft()** to compute the DFT coefficients and the inverse DFT with the syntax given in Table 4.1.

The following examples serve to illustrate the application of DFT and the inverse of DFT.

Table 4.1 MATLAB FFT Functions

X = fft(x)	**% Calculate DFT coefficients**

x =ifft(X)
x = input vector
X =DFT coefficient vector
	% Inverse of DFT

EXAMPLE 4.2

Given a sequence $x(n)$ for $0 \le n \le 3$, where $x(0)=1$, $x(1)=2$, $x(2)=3$, and $x(3)=4$, evaluate its DFT $X(k)$.

Solution:

Since $N=4$ and $W_4 = e^{-j\frac{\pi}{2}}$, using Eq. (4.8) we have a simplified formula

$$X(k) = \sum_{n=0}^{3} x(n) W_4^{kn} = \sum_{n=0}^{3} x(n) e^{-j\frac{\pi k n}{2}}.$$

Thus, for $k=0$

$$\begin{aligned} X(0) = \sum_{n=0}^{3} x(n) e^{-j0} &= x(0)e^{-j0} + x(1)e^{-j0} + x(2)e^{-j0} + x(3)e^{-j0} \\ &= x(0) + x(1) + x(2) + x(3) \\ &= 1 + 2 + 3 + 4 = 10 \end{aligned}$$

for $k=1$

$$\begin{aligned} X(1) = \sum_{n=0}^{3} x(n) e^{-j\frac{\pi n}{2}} &= x(0)e^{-j0} + x(1)e^{-j\frac{\pi}{2}} + x(2)e^{-j\pi} + x(3)e^{-j\frac{3\pi}{2}} \\ &= x(0) - jx(1) - x(2) + jx(3) \\ &= 1 - j2 - 3 + j4 = -2 + j2 \end{aligned}$$

for $k=2$

$$\begin{aligned} X(2) = \sum_{n=0}^{3} x(n) e^{-j\pi n} &= x(0)e^{-j0} + x(1)e^{-j\pi} + x(2)e^{-j2\pi} + x(3)e^{-j3\pi} \\ &= x(0) - x(1) + x(2) - x(3) \\ &= 1 - 2 + 3 - 4 = -2 \end{aligned}$$

and for $k=3$

$$\begin{aligned} X(3) = \sum_{n=0}^{3} x(n) e^{-j\frac{3\pi n}{2}} &= x(0)e^{-j0} + x(1)e^{-j\frac{3\pi}{2}} + x(2)e^{-j3\pi} + x(3)e^{-j\frac{9\pi}{2}} \\ &= x(0) + jx(1) - x(2) - jx(3) \\ &= 1 + j2 - 3 - j4 = -2 - j2 \end{aligned}$$

Let us verify the result using the MATLAB function **fft()**:

```
>> X = fft([1234])
X = 10.0000 - 2.0000 + 2.0000i - 2.0000 - 2.0000 - 2.0000i
```

EXAMPLE 4.3

Using the DFT coefficients $X(k)$ for $0 \le k \le 3$ computed in Example 4.2, evaluate its inverse DFT to determine the time domain sequence $x(n)$.

Solution:

Since $N = 4$ and $W_4^{-1} = e^{j\frac{\pi}{2}}$, using Eq. (4.15) we achieve a simplified formula

$$x(n) = \frac{1}{4}\sum_{k=0}^{3} X(k) W_4^{-nk} = \frac{1}{4}\sum_{k=0}^{3} X(k) e^{j\frac{\pi k n}{2}}.$$

Then for $n = 0$

$$x(0) = \frac{1}{4}\sum_{k=0}^{3} X(k) e^{j0} = \frac{1}{4}\left(X(0)e^{j0} + X(1)e^{j0} + X(2)e^{j0} + X(3)e^{j0}\right)$$
$$= \frac{1}{4}(10 + (-2 + j2) - 2 + (-2 - j2)) = 1$$

for $n = 1$

$$x(1) = \frac{1}{4}\sum_{k=0}^{3} X(k) e^{j\frac{k\pi}{2}} = \frac{1}{4}\left(X(0)e^{j0} + X(1)e^{j\frac{\pi}{2}} + X(2)e^{j\pi} + X(3)e^{j\frac{3\pi}{2}}\right)$$
$$= \frac{1}{4}(X(0) + jX(1) - X(2) - jX(3))$$
$$= \frac{1}{4}(10 + j(-2 + j2) - (-2) - j(-2 - j2)) = 2$$

for $n = 2$

$$x(2) = \frac{1}{4}\sum_{k=0}^{3} X(k) e^{jk\pi} = \frac{1}{4}\left(X(0)e^{j0} + X(1)e^{j\pi} + X(2)e^{j2\pi} + X(3)e^{j3\pi}\right)$$
$$= \frac{1}{4}(X(0) - X(1) + X(2) - X(3))$$
$$= \frac{1}{4}(10 - (-2 + j2) + (-2) - (-2 - j2)) = 3$$

and for $n = 3$

$$x(3) = \frac{1}{4}\sum_{k=0}^{3} X(k) e^{j\frac{k\pi 3}{2}} = \frac{1}{4}\left(X(0)e^{j0} + X(1)e^{j\frac{3\pi}{2}} + X(2)e^{j3\pi} + X(3)e^{j\frac{9\pi}{2}}\right)$$
$$= \frac{1}{4}(X(0) - jX(1) - X(2) + jX(3))$$
$$= \frac{1}{4}(10 - j(-2 + j2) - (-2) + j(-2 - j2)) = 4$$

This example actually verifies the inverse of DFT. Applying the MATLAB function **ifft()** achieves:

```
>> x = ifft([10 -2+2j -2 -2-2j])
x = 1 2 3 4
```

Now we explore the relationship between the frequency bin k and its associated frequency. Omitting the proof, the calculated N DFT coefficients $X(k)$ represent the frequency components ranging from 0 Hz (or rad/s) to f_s Hz (or ω_s rad/s), hence we can map the frequency bin k to its corresponding frequency as follows:

$$\omega = \frac{k\omega_s}{N} \text{ (rad/s)}, \tag{4.17}$$

or in terms of Hz,

$$f = \frac{kf_s}{N} \text{ (Hz)}, \tag{4.18}$$

where $\omega_s = 2\pi f_s$.

We can define the frequency resolution as the frequency step between two consecutive DFT coefficients to measure how fine the frequency domain presentation is and achieve

$$\Delta\omega = \frac{\omega_s}{N} \text{ (rad/s)}, \tag{4.19}$$

or in terms of Hz, it follows that

$$\Delta f = \frac{f_s}{N} \text{ (Hz)}. \tag{4.20}$$

Let us study the following example.

EXAMPLE 4.4

In Example 4.2, given a sequence $x(n)$ for $0 \le n \le 3$, where $x(0) = 1$, $x(1) = 2$, $x(2) = 3$, and $x(3) = 4$, we have computed four DFT coefficients $X(k)$ for $0 \le k \le 3$ as $X(0) = 10$, $X(1) = -2 + j2$, $X(2) = -2$, and $X(3) = -2 - j2$. If the sampling rate is 10 Hz,

(a) Determine the sampling period, time index, and sampling time instant for a digital sample $x(3)$ in time domain.

(b) Determine the frequency resolution, frequency bin, and mapped frequencies for each of the DFT coefficients $X(1)$ and $X(3)$ in frequency domain.

Solution:

(a) In time domain, we have the sampling period calculated as

$$T = \frac{1}{f_s} = \frac{1}{10} = 0.1 \text{ s}.$$

For data $x(3)$, the time index is $n = 3$ and the sampling time instant is determined by

$$t = nT = 3 \times 0.1 = 0.3 \text{ s}.$$

(b) In frequency domain, since the total number of DFT coefficients is four, the frequency resolution is determined by

$$\Delta f = \frac{f_s}{N} = \frac{10}{4} = 2.5 \text{ Hz}.$$

Continued

EXAMPLE 4.4—CONT'D

The frequency bin for $X(1)$ should be $k=1$ and its corresponding frequency is determined by

$$f=\frac{kf_s}{N}=\frac{1\times 10}{4}=2.5\,\text{Hz}.$$

Similarly, for $X(3)$ and $k=3$,

$$f=\frac{kf_s}{N}=\frac{3\times 10}{4}=7.5\,\text{Hz}.$$

Note that from Eq. (4.4), $k=3$ is equivalent to $k-N=3-4=-1$, and $f=7.5\,\text{Hz}$ is also equivalent to the frequency $f=(-1\times 10)/4=-2.5\,\text{Hz}$, which corresponds to the negative side spectrum. The amplitude spectrum at 7.5 Hz after folding should match the one at $f_s-f=10.0-7.5=2.5\,\text{Hz}$. We will apply these developed notations in the following section for amplitude and power spectral estimation.

4.2 AMPLITUDE SPECTRUM AND POWER SPECTRUM

One of the DFT applications is transformation of a finite-length digital signal $x(n)$ into the spectrum in frequency domain. Fig. 4.7 demonstrates such an application, where A_k and P_k are the computed amplitude spectrum and the power spectrum, respectively, using the DFT coefficients $X(k)$.

First, we achieve the digital sequence $x(n)$ by sampling the analog signal $x(t)$ and truncating the sampled signal with a data window of a length $T_0=NT$, where T is the sampling period and N is the number of data points. The time for data window is

$$T_0=NT. \tag{4.21}$$

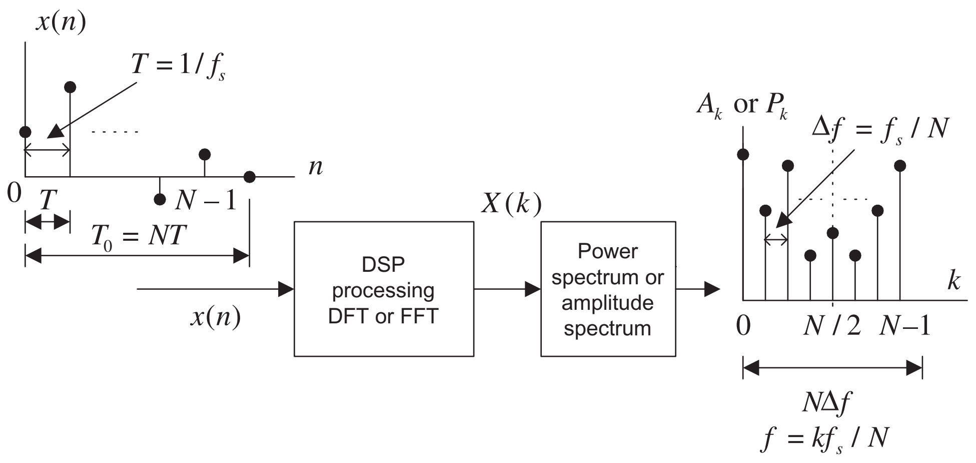

FIG. 4.7

Applications of DFT/FFT.

For the truncated sequence $x(n)$ with a range of $n = 0, 1, 2, \cdots, N-1$, we get

$$x(0), x(1), x(2), \cdots, x(N-1). \tag{4.22}$$

Next, we apply the DFT to the obtained sequence, $x(n)$, to get the N DFT coefficients

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk}, \text{ for } k = 0, 1, 2, \cdots, N-1. \tag{4.23}$$

Since each calculated DFT coefficient is a complex number, it is not convenient to plot it vs. its frequency index. Hence, after evaluating Eq. (4.23), the magnitude and phase of each DFT coefficient (we refer them as the amplitude spectrum and phase spectrum, respectively) can be determined and plotted vs. its frequency index. We define the amplitude spectrum as

$$A_k = \frac{1}{N}|X(k)| = \frac{1}{N}\sqrt{(\text{Real}[X(k)])^2 + (\text{Imag}[X(k)])^2}, \ k = 0, 1, 2, \cdots, N-1. \tag{4.24}$$

We can modify the amplitude spectrum to a one-sided amplitude spectrum by doubling the amplitudes in Eq. (4.24), keeping the original DC term at $k = 0$. Thus we have

$$\overline{A}_k = \begin{cases} \frac{1}{N}|X(0)|, & k = 0 \\ \frac{2}{N}|X(k)|, & k = 1, \cdots, N/2 \end{cases}. \tag{4.25}$$

We can also map the frequency bin k to its corresponding frequency as

$$f = \frac{kf_s}{N}. \tag{4.26}$$

Correspondingly, the phase spectrum is given by

$$\varphi_k = \tan^{-1}\left(\frac{\text{Imag}[X(k)]}{\text{Real}[X(k)]}\right), k = 0, 1, 2, \cdots, N-1. \tag{4.27}$$

Besides the amplitude spectrum, the power spectrum is also used. The DFT power spectrum is defined as

$$P_k = \frac{1}{N^2}|X(k)|^2 = \frac{1}{N^2}\left\{(\text{Real}[X(k)])^2 + (\text{Imag}[X(k)])^2\right\}, \ k = 0, 1, 2, \cdots, N-1. \tag{4.28}$$

Similarly, for a one-sided power spectrum, we get

One-sided power spectrum

$$\overline{P}_k = \begin{cases} \frac{1}{N^2}|X(0)|^2 & k = 0 \\ \frac{2}{N^2}|X(k)|^2 & k = 1, \cdots, N/2 \end{cases} \tag{4.29}$$

$$\text{and } f = \frac{kf_s}{N}. \tag{4.30}$$

Again, note that the frequency resolution, which denotes the frequency spacing between DFT coefficients in frequency domain, is defined as

$$\Delta f = \frac{f_s}{N} \text{ (Hz)}. \tag{4.31}$$

It follows that better frequency resolution can be achieved using a longer data sequence.

EXAMPLE 4.5

Consider the sequence in Fig. 4.8

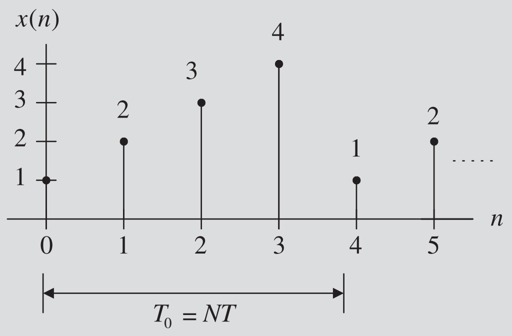

FIG. 4.8

Sampled values in Example 4.5.

Assuming that $f_s = 100$ Hz, compute the amplitude spectrum, phase spectrum, and power spectrum.

Solution:

Since $N = 4$, and using the DFT shown in Example 4.2, we find the DFT coefficients to be

$$\begin{aligned} &X(0) = 10 \\ &X(1) = -2 + j2 \\ &X(2) = -2 \\ &X(3) = -2 - j2. \end{aligned}$$

The amplitude spectrum, phase spectrum, and power density spectrum are computed as follows:

$$\text{for } k = 0, f = k \cdot f_s/N = 0 \times 100/4 = 0\,\text{Hz},$$

$$A_0 = \frac{1}{4}|X(0)| = 2.5, \varphi_0 = \tan^{-1}\left(\frac{\text{Imag}[X(0)]}{\text{Real}([X(0)]}\right) = 0^0, P_0 = \frac{1}{4^2}|X(0)|^2 = 6.25$$

$$\text{for } k = 1, f = 1 \times 100/4 = 25\,\text{Hz},$$

$$A_1 = \frac{1}{4}|X(1)| = 0.7071, \varphi_1 = \tan^{-1}\left(\frac{\text{Imag}[X(1)]}{\text{Real}[X(1)]}\right) = 135^0, P_1 = \frac{1}{4^2}|X(1)|^2 = 0.5000$$

$$\text{for } k = 2, f = 2 \times 100/4 = 50\,\text{Hz},$$

$$A_2 = \frac{1}{4}|X(2)| = 0.5, \varphi_2 = \tan^{-1}\left(\frac{\text{Imag}[X(2)]}{\text{Real}[X(2)]}\right) = 180^0, P_2 = \frac{1}{4^2}|X(2)|^2 = 0.2500$$

Similarly,

$$\text{for } k = 3, f = 3 \times 100/4 = 75\,\text{Hz},$$

$$A_3 = \frac{1}{4}|X(3)| = 0.7071, \varphi_3 = \tan^{-1}\left(\frac{\text{Imag}[X(3)]}{\text{Real}[X(3)]}\right) = -135^0, P_3 = \frac{1}{4^2}|X(3)|^2 = 0.5000.$$

Thus, the sketches for the amplitude spectrum, phase spectrum, and power spectrum are given in Fig. 4.9.

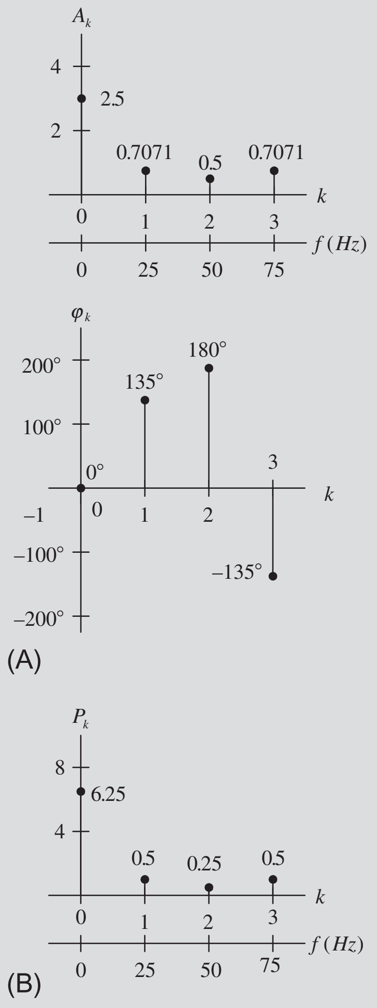

FIG. 4.9

(A) Amplitude spectrum and phase spectrum in Example 4.5. (B) Power density spectrum in Example 4.5.

Note that the folding frequency in this example is 50 Hz and the amplitude and power spectrum values at 75 Hz are each image counterparts (corresponding negative-indexed frequency components), respectively. Thus, values at 0, 25, and 50 Hz are corresponding to the positive-indexed frequency components.

We can easily find the one-sided amplitude spectrum and one-sided power spectrum as

$$\overline{A}_0 = 2.5, \overline{A}_1 = 1.4141, \overline{A}_2 = 1 \text{ and}$$
$$\overline{P}_0 = 6.25, \overline{P}_1 = 2, \overline{P}_2 = 1.$$

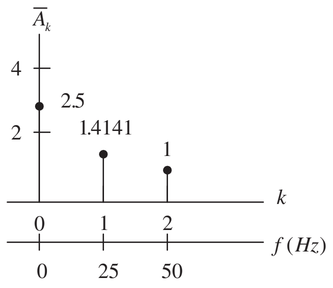

FIG. 4.10

One-sided amplitude spectrum in Example 4.5.

We plot the one-sided amplitude spectrum for comparison as shown in Fig. 4.10.

Note that in the one-sided amplitude spectrum, the negative-indexed frequency components are added back to the corresponding positive-indexed frequency components; thus each amplitude value other than DC term is doubled. It represents the frequency components up to the folding frequency.

EXAMPLE 4.6

Consider a digital sequence sampled at the rate of 10 kHz. If we use a size of 1024 data points and apply the 1024-point DFT to compute the spectrum,

(a) Determine the frequency resolution.

(b) Determine the highest frequency in the spectrum.

Solution:

(a)

$$\Delta f = \frac{f_s}{N} = \frac{10,000}{1024} = 9.776\,\text{Hz}.$$

(b) The highest frequency is the folding frequency, given by

$$f_{\max} = \frac{N}{2}\Delta f = \frac{f_s}{2}$$
$$= 512 \times 9.776 = 5000\,\text{Hz}.$$

As shown in Fig. 4.7, the DFT coefficients may be computed via a *fast Fourier transform* (FFT) algorithm (see Section 4.5). The FFT is a very efficient algorithm for computing DFT coefficients. In this book, we only focus on the FFT algorithm which requires the time domain sequence $x(n)$ with a length of data points equal to a power of 2; that is, 2^m samples, where m is a positive integer. For example, the number of samples in $x(n)$ can be $N = 2, 4, 8, 16$, etc.

In the case of using the FFT algorithm to compute DFT coefficients, where the length of the available data is not equal to a power of 2 (required by the FFT), we can pad the data sequence with zeros to create a new sequence with a larger number of samples, $\overline{N} = 2^m > N$. The modified data sequence for applying FFT, therefore, is

$$\bar{x}(n) = \begin{cases} x(n) & 0 \le n \le N-1 \\ 0 & N \le n \le \overline{N}-1 \end{cases}. \tag{4.32}$$

It is very important to note that the signal spectra obtained via zero padding the data sequence in Eq. (4.32) does not add any new information and does not contain more accurate signal spectral presentation. In this situation, the frequency spacing is reduced due to more DFT points, and the achieved spectrum is a interpolated version with "better display." We illustrate the zero-padding effect via the following example instead of theoretical analysis. A theoretical discussion of zero padding in FFT can be found in Proakis and Manolakis (1996).

Fig. 4.11A shows the 12 data samples from an analog signal containing frequencies of 10 and 25 Hz at a sampling rate of 100 Hz, and the amplitude spectrum obtained by applying the DFT. Fig. 4.11B displays the signal samples with padding of four zeros to the original data to make up a data sequence of 16 samples, along with the amplitude spectrum calculated by FFT. The data sequence padded with 20 zeros and its calculated amplitude spectrum using FFT are shown in Fig. 4.11C. It is evident that increasing the data length via zero padding to compute the signal spectrum does not add basic information and does not change the spectral shape but gives the "interpolated spectrum" with the reduced frequency spacing. We can get a better view of the two spectral peaks described in this case.

The only way to obtain the detailed signal spectrum with a fine frequency resolution is to apply more available data samples, that is, a longer sequence of data. Here, we choose to pad the least number of zeros to satisfy the minimum FFT computational requirement. Let us look at another example.

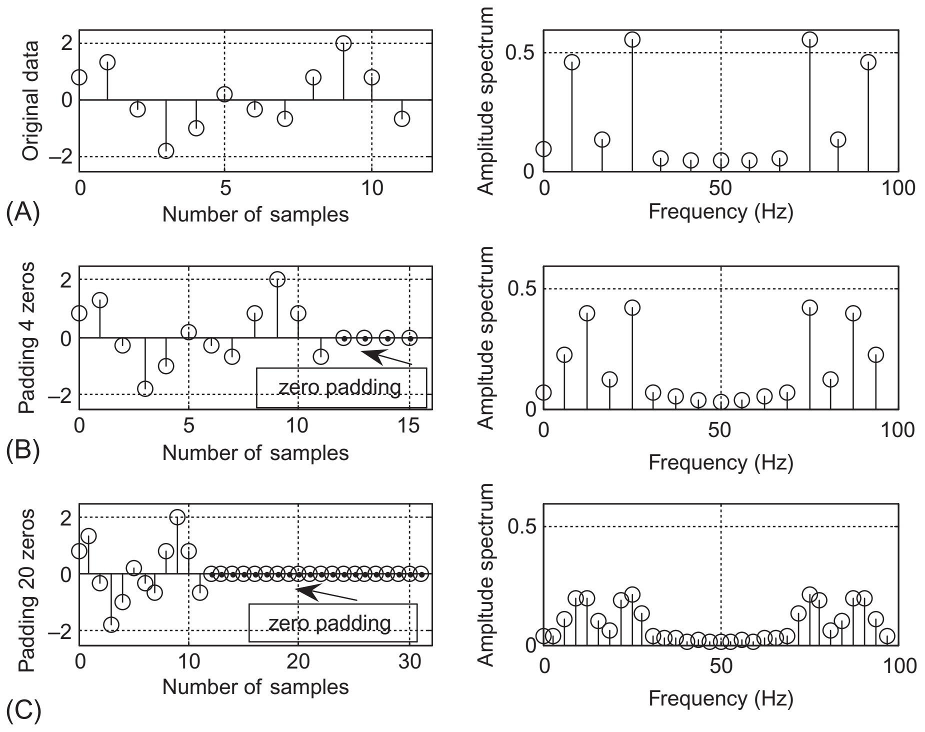

FIG. 4.11

Zero-padding effect by using FFT.

EXAMPLE 4.7

We use the DFT to compute the amplitude spectrum of a sampled data sequence with a sampling rate $f_s = 10\,\text{kHz}$. Given that it requires the frequency resolution to be less than 0.5 Hz, determine the number of data points by using the FFT algorithm, assuming that the data samples are available.

Solution:

$$\Delta f = 0.5\,\text{Hz}$$
$$N = \frac{f_s}{\Delta f} = \frac{10{,}000}{0.5} = 20{,}000$$

Since we use the FFT to compute the spectrum, the number of the data points must be a power of 2, that is,

$$N = 2^{15} = 32{,}768.$$

and the resulting frequency resolution can be recalculated as

$$\Delta f = \frac{f_s}{N} = \frac{10{,}000}{32{,}768} = 0.31\,\text{Hz}.$$

Next, we study a MATLAB example.

EXAMPLE 4.8

Given a sinusoid

$$x(n) = 2 \times \sin\left(2000\pi \frac{n}{8000}\right)$$

obtained by sampling the analog signal

$$x(t) = 2 \times \sin(2000\pi t)$$

with a sampling rate of $f_s = 8000\text{Hz}$,

(a) Use the MATLAB DFT to compute the signal spectrum with the frequency resolution to be equal to or less than 8 Hz.

(b) Use the MATALB FFT and zero padding to compute the signal spectrum, assuming that the data samples are available in (a).

Solution:

(a) The number of data points is found to be

$$N = \frac{f_s}{\Delta f} = \frac{8000}{8} = 1000\,.$$

There is no zero padding needed if we use the DFT formula. The detailed implementation is given in Program 4.1. The first and second plots in Fig. 4.12 show the two-sided amplitude and power spectra, respectively, using the DFT, where each frequency counterpart at 7000 Hz appears. The third and fourth plots are the one-sided amplitude and power spectra, where the true frequency contents are displayed from 0 Hz to the Nyquist frequency of 4 kHz (folding frequency).

(b) If the FFT is used, the number of data points must be a power of 2. Hence we choose

$$N = 2^{10} = 1024.$$

Assuming that there are only 1000 data samples available in (a), we need to pad 24 zeros to the original 1000 data samples before applying the FFT algorithm, as required.

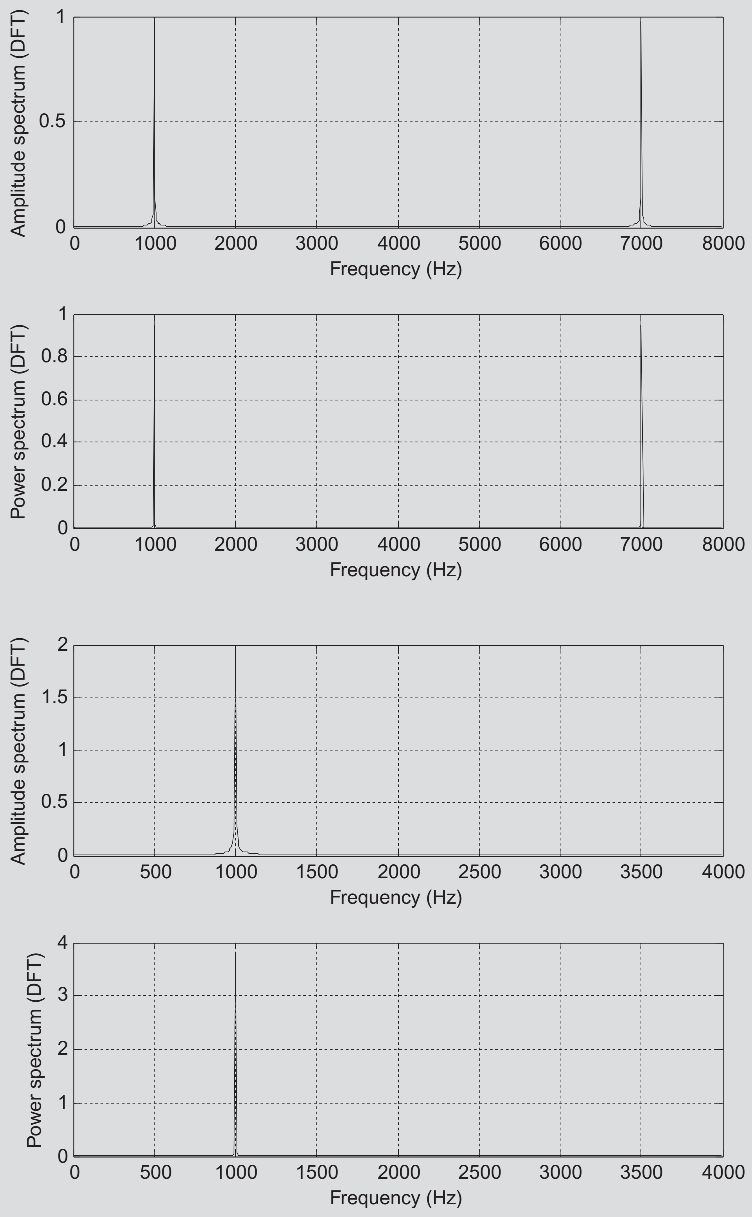

FIG. 4.12

Amplitude spectrum and power spectrum using DFT for Example 4.8.

Thus the calculated frequency resolution is $\Delta f = f_s/N = 8000/1024 = 7.8125$ Hz. Note that this is an interpolated frequency resolution by using zero padding. The zero padding actually interpolates a signal spectrum and carries no additional frequency information. Fig. 4.13 shows the spectral plots using FFT. The detailed implementation is given in Program 4.1.

Continued

EXAMPLE 4.8—CONT'D

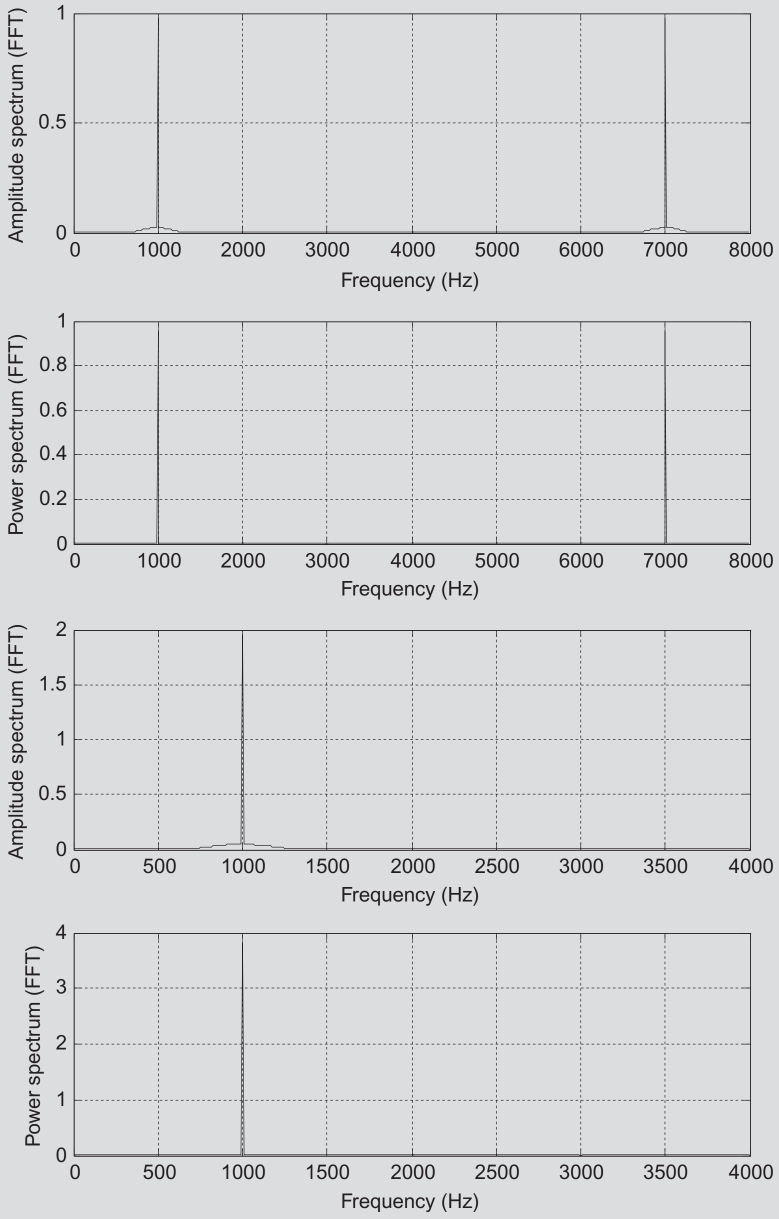

FIG. 4.13

Amplitude spectrum and power spectrum using FFT for Example 4.8.

Program 4.1 MATLAB program for Example 4.8

```
% Example 4.8
close all;clear all
% Generate the sine wave sequence
fs=8000;       %Sampling rate
N=1000;       % Number of data points
x=2*sin(2000*pi*[0:1:N-1]/fs);
% Apply the DFT algorithm
figure(1)
xf=abs(fft(x))/N;        %Compute the amplitude spectrum
P=xf.*xf;          %Compute power spectrum
f=[0:1:N-1]*fs/N;        %Map the frequency bin to frequency (Hz)
subplot(2,1,1); plot(f,xf );grid
xlabel('Frequency (Hz)'); ylabel('Amplitude spectrum (DFT)');
subplot(2,1,2);plot(f,P);grid
xlabel('Frequency (Hz)'); ylabel('Power spectrum (DFT)');
figure(2)
% Convert it to one side spectrum
xf(2:N)=2*xf(2:N);     % Get the single-side spectrum
P=xf.*xf;       % Calculate the power spectrum
f=[0:1:N/2]*fs/N     % Frequencies up to the folding frequency
subplot(2,1,1); plot(f,xf(1:N/2+1));grid
xlabel('Frequency (Hz)'); ylabel('Amplitude spectrum (DFT)');
subplot(2,1,2);plot(f,P(1:N/2+1));grid
xlabel('Frequency (Hz)'); ylabel('Power spectrum (DFT)');
figure (3)
% Zero padding to the length of 1024
x=[x,zeros(1,24)];
N=length(x);
xf=abs(fft(x))/N;     %Compute amplitude spectrum with zero padding
P=xf.*xf;        %Compute power spectrum
f=[0:1:N-1]*fs/N;     %Map frequency bin to frequency (Hz)
subplot(2,1,1); plot(f,xf );grid
xlabel('Frequency (Hz)'); ylabel('Amplitude spectrum (FFT)');
subplot(2,1,2);plot(f,P);grid
xlabel('Frequency (Hz)'); ylabel('Power spectrum (FFT)');
figure(4)
% Convert it to one side spectrum
xf(2:N)=2*xf(2:N);
P=xf.*xf;
f=[0:1:N/2]*fs/N;
subplot(2,1,1); plot(f,xf(1:N/2+1));grid
xlabel('Frequency (Hz)'); ylabel('Amplitude spectrum (FFT)');
subplot(2,1,2);plot(f,P(1:N/2+1));grid
xlabel('Frequency (Hz)'); ylabel('Power spectrum (FFT)');
```

4.3 SPECTRAL ESTIMATION USING WINDOW FUNCTIONS

When we apply DFT to the sampled data in the previous section, we theoretically imply the following assumptions: first, the sampled data are periodic to themselves (repeat themselves), and second, the sampled data is continuous to themselves and band limited to the folding frequency. The second assumption is often violated, thus the discontinuity produces undesired harmonic frequencies. Consider a pure 1-Hz sine wave with 32 samples shown in Fig. 4.14.

As shown in the figure, if we use a window size of $N=16$ samples, which is multiple of the two waveform cycles, the second window repeats with continuity. However, when the window size is chosen to be 18 samples, which is not multiple of the waveform cycles (2.25 cycles), the second window repeats the first window with discontinuity. It is this discontinuity that produces harmonic frequencies that are not present in the original signal. Fig. 4.15 shows the spectral plots for both cases using the DFT/FFT directly.

The first spectral plot contains a single frequency component, as we expected, while the second spectrum has the expected frequency component plus many harmonics, which do not exist in the original signal. We called such an effect *spectral leakage*. The amount of spectral leakage shown in the second plot is due to amplitude discontinuity in time domain. The bigger the discontinuity, the more the leakage. To reduce the effect of spectral leakage, a window function can be used whose amplitude tapers smoothly and gradually toward zero at both ends. Applying the window function $w(n)$ to a data sequence $x(n)$ to obtain the windowed sequence $x_w(n)$ is better illustrated in Fig. 4.16 using Eq. (4.33):

$$x_w(n) = x(n)w(n), \text{ for } n = 0, 1, \cdots, N-1. \tag{4.33}$$

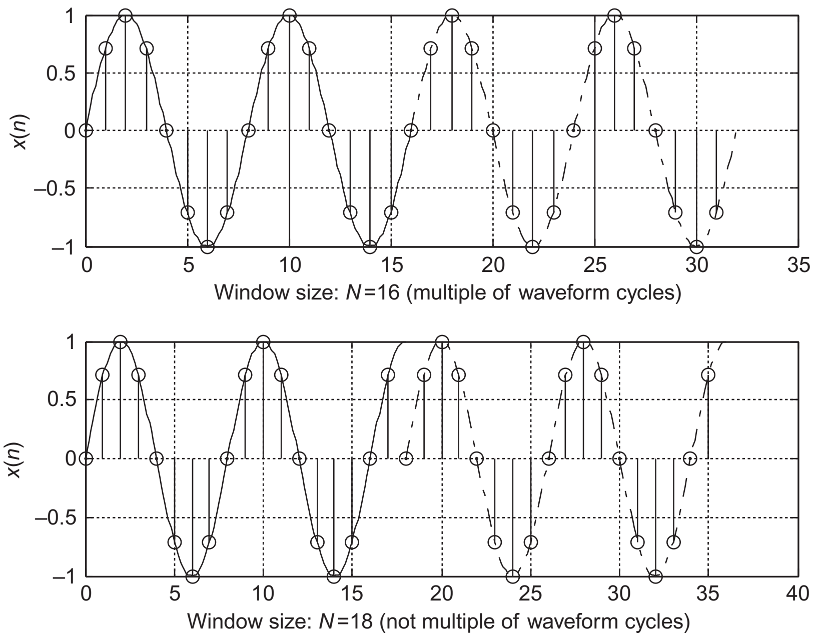

FIG. 4.14

Sampling a 1-Hz sine wave using (top) 16 samples per cycle and (bottom) 18 samples per cycle (the sampling interval: $T=1/8$ s).

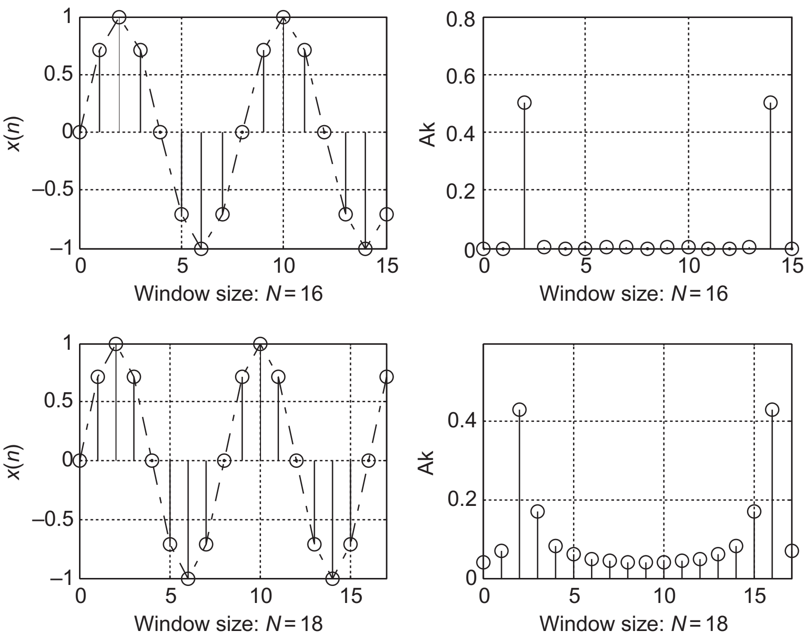

FIG. 4.15

Signal samples and spectra without spectral leakage and with spectral leakage.

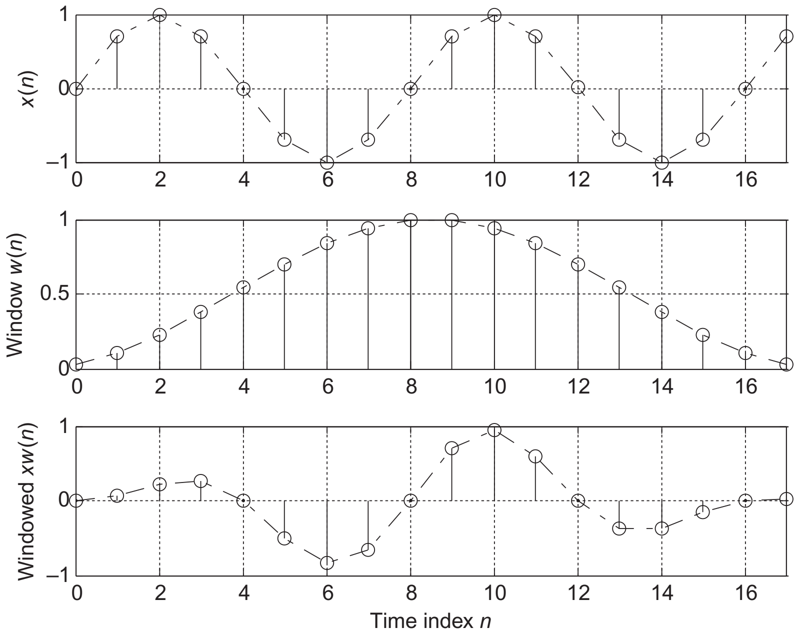

FIG. 4.16

Illustration of the window operation.

The top plot is the data sequence $x(n)$, and the middle plot is the window function $w(n)$. The bottom plot in Fig. 4.16 shows that the windowed sequence $x_w(n)$ is tapped down by a window function to zero at both ends such that the discontinuity is dramatically reduced.

EXAMPLE 4.9

In Fig. 4.16, given

- $x(2) = 1$ and $w(2) = 0.2265$;
- $x(5) = -0.7071$ and $w(5) = 0.7008$,

calculate the windowed sequence data points $x_w(2)$ and $x_w(5)$.

Solution:

Applying the window function operation leads to

$$x_w(2) = x(2) \times w(2) = 1 \times 0.2265 = 0.2265 \text{ and}$$
$$x_w(5) = x(5) \times w(5) = -0.7071 \times 0.7008 = -0.4956,$$

which agree with the values shown in the bottom plot in Fig. 4.16.

Using the windowed function shown in Example 4.9, the spectral plot is reproduced. As a result, the spectral leakage is greatly reduced, as shown in Fig. 4.17.

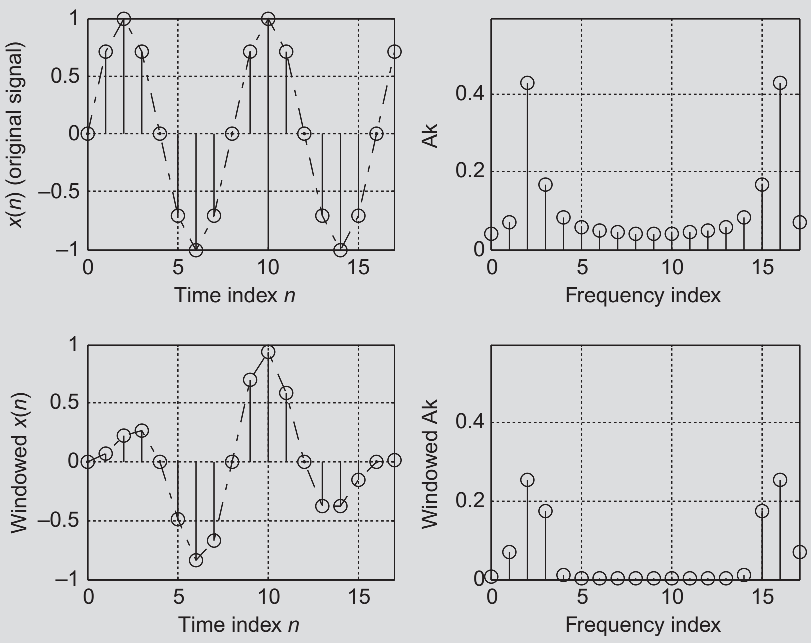

FIG. 4.17

Comparison of spectra calculated without using a window function and using a window function to reduce spectral leakage.

The common window functions are listed as follows:

The rectangular window (no window function):

$$w_R(n) = 1, 0 \le n \le N-1. \tag{4.34}$$

The triangular window:

$$w_{tri}(n) = 1 - \frac{|2n - N + 1|}{N-1}, 0 \le n \le N-1. \tag{4.35}$$

The Hamming window:

$$w_{hm}(n) = 0.54 - 0.46\cos\left(\frac{2\pi n}{N-1}\right), 0 \le n \le N-1. \tag{4.36}$$

The Hanning window:

$$w_{hn}(n) = 0.5 - 0.5\cos\left(\frac{2\pi n}{N-1}\right), 0 \le n \le N-1. \tag{4.37}$$

Plots for each window function for a size of 20 samples are shown in Fig. 4.18.

The following example details each step for computing the spectral information using the window functions.

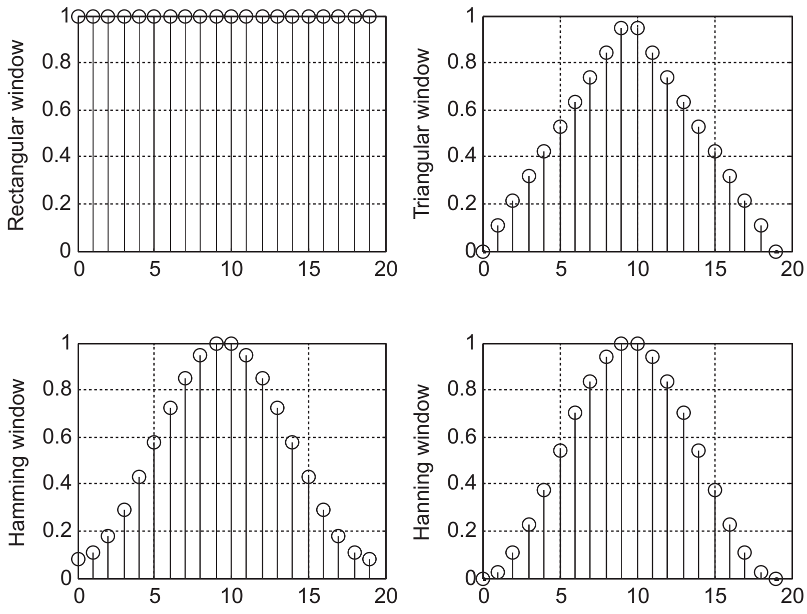

FIG. 4.18

Plots of window sequences.

EXAMPLE 4.10

Considering the sequence $x(0)=1$, $x(1)=2$, $x(2)=3$, $x(3)=4$, and given $f_s=100\,\text{Hz}$, $T=0.01$ s, compute the amplitude spectrum, phase spectrum, and power spectrum

(a) Using the triangle window function.

(b) Using the Hamming window function.

Solution:

(a) Since $N=4$, from the triangular window function, we have

$$w_{\text{tri}}(0)=1-\frac{|2\times 0-4+1|}{4-1}=0$$

$$w_{\text{tri}}(1)=1-\frac{|2\times 1-4+1|}{4-1}=0.6667.$$

Similarly, $w_{\text{tri}}(2)=0.6667, w_{\text{tri}}(3)=0$. Next, the windowed sequence is computed as

$$x_w(0)=x(0)\times w_{\text{tri}}(0)=1\times 0=0$$

$$x_w(1)=x(1)\times w_{\text{tri}}(1)=2\times 0.6667=1.3334$$

$$x_w(2)=x(2)\times w_{\text{tri}}(2)=3\times 0.6667=2$$

$$x_w(3)=x(3)\times w_{\text{tri}}(3)=4\times 0=0.$$

Apply DFT Eq. (4.8) to $x_w(n)$ for $k=0$, 1, 2, 3, respectively,

$$X(k)=x_w(0)W_4^{k\times 0}+x_w(1)W_4^{k\times 1}+x_w(2)W_4^{k\times 2}+x_w(3)W_4^{k\times 3}.$$

We have the following results:

$$X(0)=3.3334$$

$$X(1)=-2-j1.3334$$

$$X(2)=0.6666$$

$$X(3)=-2+j1.3334$$

$$\Delta f=\frac{1}{NT}=\frac{1}{4\cdot 0.01}=25\,\text{Hz}.$$

Applying Eqs. (4.24), (4.27), and (4.28) leads to

$$A_0=\frac{1}{4}|X(0)|=0.8334,\ \varphi_0=\tan^{-1}\left(\frac{0}{3.3334}\right)=0^0,\ P_0=\frac{1}{4^2}|X(0)|^2=0.6954$$

$$A_1=\frac{1}{4}|X(1)|=0.6009,\ \varphi_1=\tan^{-1}\left(\frac{-1.3334}{-2}\right)=-146.31^0,\ P_1=\frac{1}{4^2}|X(1)|^2=0.3611$$

$$A_2=\frac{1}{4}|X(2)|=0.1667,\ \varphi_2=\tan^{-1}\left(\frac{0}{0.6666}\right)=0^0,\ P_1=\frac{1}{4^2}|X(2)|^2=0.0278$$

Similarly,

$$A_3 = \frac{1}{4}|X(3)| = 0.6009, \varphi_3 = \tan^{-1}\left(\frac{1.3334}{-2}\right) = 146.31^0, P_3 = \frac{1}{4^2}|X(3)|^2 = 0.3611.$$

(b) Since $N=4$, from the Hamming window function, we have

$$w_{hm}(0) = 0.54 - 0.46\cos\left(\frac{2\pi \times 0}{4-1}\right) = 0.08$$

$$w_{hm}(1) = 0.54 - 0.46\cos\left(\frac{2\pi \times 1}{4-1}\right) = 0.77.$$

Similarly, $w_{hm}(2) = 0.77$, and $w_{hm}(3) = 0.08$. Next, the windowed sequence is computed as

$$x_w(0) = x(0) \times w_{hm}(0) = 1 \times 0.08 = 0.08$$

$$x_w(1) = x(1) \times w_{hm}(1) = 2 \times 0.77 = 1.54$$

$$x_w(2) = x(2) \times w_{hm}(2) = 3 \times 0.77 = 2.31$$

$$x_w(0) = x(3) \times w_{hm}(3) = 4 \times 0.08 = 0.32.$$

Apply DFT Eq. (4.8) to $x_w(n)$ for $k=0$, 1, 2, 3, respectively,

$$X(k) = x_w(0)W_4^{k\times 0} + x_w(1)W_4^{k\times 1} + x_w(2)W_4^{k\times 2} + x_w(3)W_4^{k\times 3}.$$

We yield the following:

$$X(0) = 4.25$$

$$X(1) = -2.23 - j1.22$$

$$X(2) = 0.53$$

$$X(3) = -2.23 + j1.22$$

$$\Delta f = \frac{1}{NT} = \frac{1}{4 \cdot 0.01} = 25\,\text{Hz}.$$

Applying Eqs. (4.24), (4.27), and (4.28), we achieve

$$A_0 = \frac{1}{4}|X(0)| = 1.0625, \varphi_0 = \tan^{-1}\left(\frac{0}{4.25}\right) = 0^0, P_0 = \frac{1}{4^2}|X(0)|^2 = 1.1289$$

$$A_1 = \frac{1}{4}|X(1)| = 0.6355, \varphi_1 = \tan^{-1}\left(\frac{-1.22}{-2.23}\right) = -151.32^0, P_1 = \frac{1}{4^2}|X(1)|^2 = 0.4308$$

$$A_2 = \frac{1}{4}|X(2)| = 0.1325, \varphi_2 = \tan^{-1}\left(\frac{0}{0.53}\right) = 0^0, P_2 = \frac{1}{4^2}|X(2)|^2 = 0.0176.$$

Similarly,

$$A_3 = \frac{1}{4}|X(3)| = 0.6355, \varphi_3 = \tan^{-1}\left(\frac{1.22}{-2.23}\right) = 151.32^0, P_3 = \frac{1}{4^2}|X(3)|^2 = 0.4308.$$

EXAMPLE 4.11

Given the sinusoid

$$x(n) = 2 \times \sin\left(2000\pi \frac{n}{8000}\right)$$

obtained by using a sampling rate of $f_s = 8000\,\text{Hz}$, use the DFT to compute the spectrum with the following specifications:

(a) Compute the spectrum of a triangular window function with a window size = 50.
(b) Compute the spectrum of a Hamming window function with a window size = 100.
(c) Compute the spectrum of a Hanning window function with a window size = 150 and one-sided spectrum.

The MATLAB program is listed in Program 4.2, and results are plotted in Figs. 4.19–4.21. As compared with the no-windowed (rectangular window) case, all three windows are able to effectively reduce the spectral leakage, as shown in the figures.

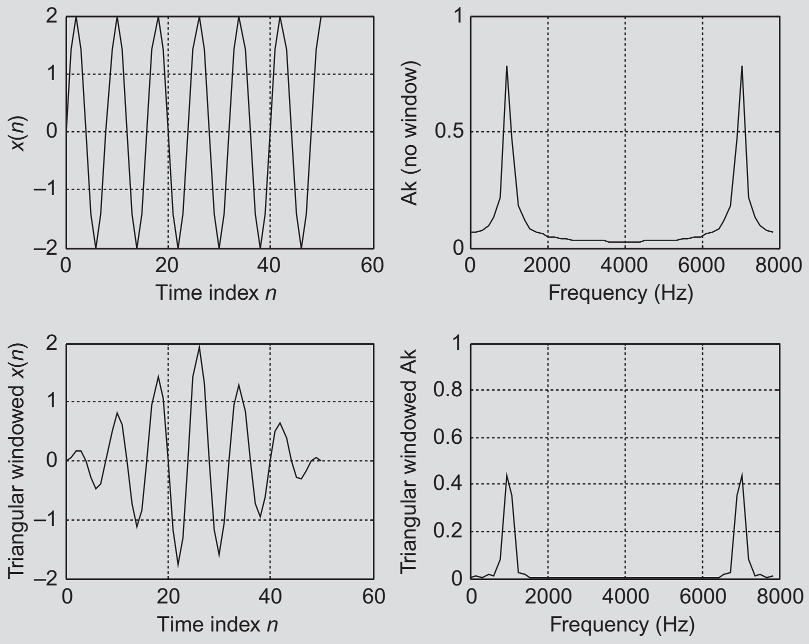

FIG. 4.19

Comparison of a spectrum without using a window function and a spectrum using a triangular window of size of 50 samples in Example 4.11.

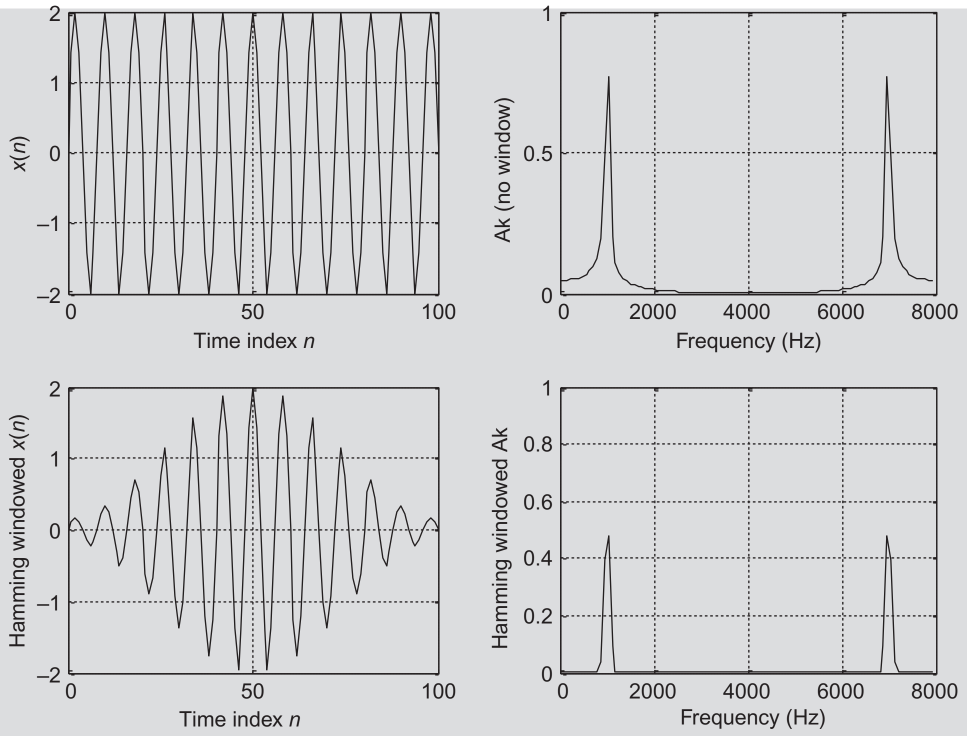

FIG. 4.20

Comparison of a spectrum without using a window function and a spectrum using a Hamming window of size of 100 samples in Example 4.11.

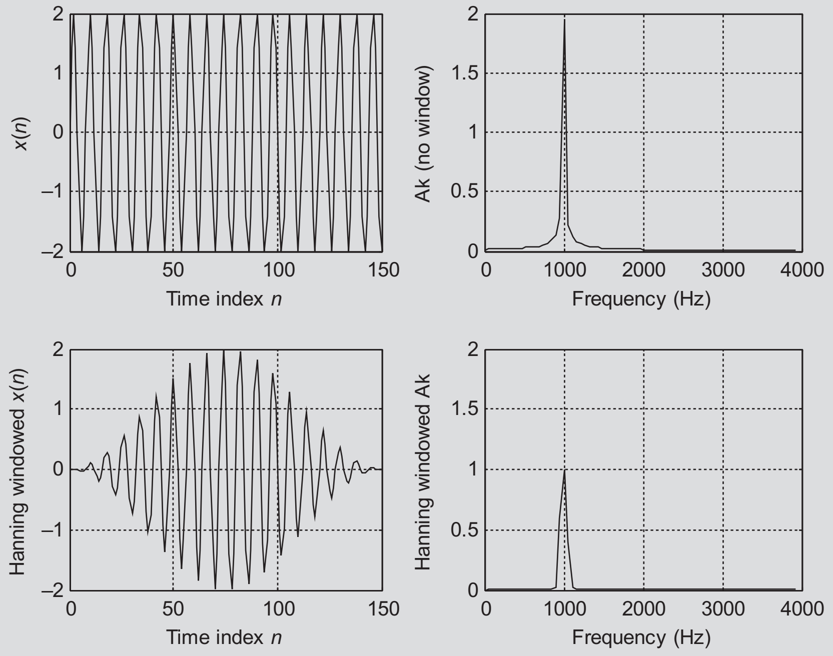

FIG. 4.21

Comparison of a one-sided spectrum without using the window function and a one-sided spectrum using a Hanning window of size of 150 samples in Example 4.11.

Program 4.2 MATLAB program for Example 4.11

```
%Example 4.11
close all;clear all
% Generate the sine wave sequence
fs=8000; T=1/fs;    % Sampling rate and sampling period
x=2*sin(2000*pi*[0:1:50]*T); %Generate 51 2000-Hz samples.
% Apply the FFT algorithm
N=length(x);
index_t=[0:1:N-1];
f=[0:1:N-1]*8000/N;      %Map frequency bin to frequency (Hz)
xf=abs(fft(x))/N;        %Calculate amplitude spectrum
figure(1)
%Using Bartlett window
x_b=x.*bartlett(N)';      %Apply triangular window function
xf_b=abs(fft(x_b))/N;    %Calculate amplitude spectrum
subplot(2,2,1);plot(index_t,x);grid
xlabel('Time index n'); ylabel('x(n)');
subplot(2,2,3); plot(index_t,x_b);grid
xlabel('Time index n'); ylabel('Triangular windowed x(n)');
subplot(2,2,2);plot(f,xf );grid;axis([0 8000 0 1]);
xlabel('Frequency (Hz)'); ylabel('Ak (no window)');
subplot(2,2,4); plot(f,xf_b);grid; axis([0 8000 0 1]);
xlabel('Frequency (Hz)'); ylabel('Triangular windowed Ak');
figure(2)
% Generate the sine wave sequence
x=2*sin(2000*pi*[0:1:100]*T); %Generate 101 2000-Hz samples.
% Apply the FFT algorithm
N=length(x);
index_t=[0:1:N-1];
f=[0:1:N-1]*fs/N;
xf=abs(fft(x))/N;
%Using Hamming window
x_hm=x.*hamming(N)';          %Apply Hamming window function
xf_hm=abs(fft(x_hm))/N;      %Calculate amplitude spectrum
subplot(2,2,1);plot(index_t,x);grid
xlabel('Time index n'); ylabel('x(n)');
subplot(2,2,3); plot(index_t,x_hm);grid
xlabel('Time index n'); ylabel('Hamming windowed x(n)');
subplot(2,2,2);plot(f,xf );grid;axis([0 fs 0 1]);
xlabel('Frequency (Hz)'); ylabel('Ak (no window)');
subplot(2,2,4); plot(f,xf_hm);grid;axis([0 fs 0 1]);
xlabel('Frequency (Hz)'); ylabel('Hamming windowed Ak');
figure(3)
% Generate the sine wave sequence
x=2*sin(2000*pi*[0:1:150]*T); % Generate 151 2-kHz samples
% Apply the FFT algorithm
N=length(x);
index_t=[0:1:N-1];
f=[0:1:N-1]*fs/N;
xf=2*abs(fft(x))/N;xf(1)=xf(1)/2; % Single-sided spectrum
%Using Hanning window
```

```
x_hn=x.*hann(N)';
xf_hn=2*abs(fft(x_hn))/N;xf_hn(1)=xf_hn(1)/2; %Single-sided spectrum
subplot(2,2,1);plot(index_t,x);grid
xlabel('Time index n'); ylabel('x(n)');
subplot(2,2,3); plot(index_t,x_hn);grid
xlabel('Time index n'); ylabel('Hanning windowed x(n)');
subplot(2,2,2);plot(f(1:(N-1)/2),xf(1:(N-1)/2));grid;axis([0 fs/2 0 2]);
xlabel('Frequency (Hz)'); ylabel('Ak (no window)');
subplot(2,2,4); plot(f(1:(N-1)/2),xf_hn(1:(N-1)/2));grid;axis([0 fs/2 0 2]);
xlabel('Frequency (Hz)'); ylabel('Hanning windowed Ak');
```

4.4 APPLICATION TO SIGNAL SPECTRAL ESTIMATION

The following plots show the comparisons of amplitude spectral estimation for a speech data (we.dat) with 2001 samples and a sampling rate of 8000 Hz using the rectangular window (no window) function and the Hamming window function. As demonstrated in Fig. 4.22 (two-sided spectrum) and Fig. 4.23

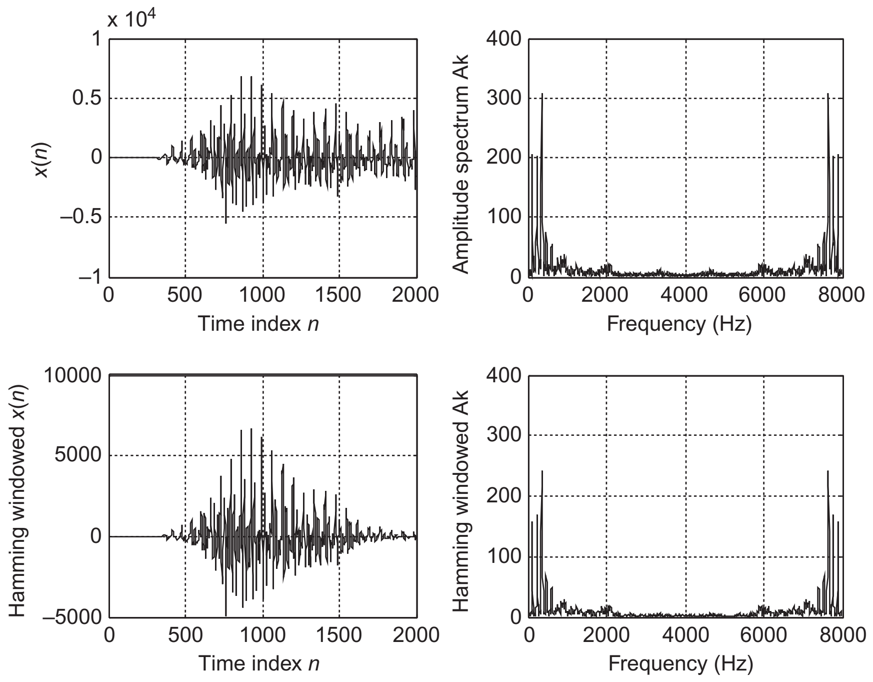

FIG. 4.22

Comparison of a spectrum without using a window function and a spectrum using the Hamming window for speech data.

(one-sided spectrum), there is little difference between the amplitude spectrum using the Hamming window function and the spectrum without using the window function. This is due to the fact that when the data length of the sequence (e.g., 2001 samples) increases, the frequency resolution will be improved and the spectral leakage will become less significant. However, when data length is short, reducing spectral leakage using a window function will become prominent.

Next, we compute the one-sided spectrum for a 32-bit seismic data sampled at 15 Hz (provided by the USGS Albuquerque Seismological Laboratory) with 6700 data samples. The computed spectral plots without using a window function and using the Hamming window are displayed in Fig. 4.24. We can see that most of seismic signal components are below 3 Hz.

We also compute the one-sided spectrum for a standard ECG signal from the MIT-BIH (Massachusetts Institute of Technology-Beth Israel Hospital) Database. The ECG signal contains frequency components ranging from 0.05 to 100 Hz sampled at 500 Hz. As shown in Fig. 4.25, there is a spike located at 60 Hz. This is due to the 60-Hz power line interference when the ECG is acquired via the ADC acquisition process. This 60-Hz interference can be removed by using a digital notch filter which will be studied in Chapter 8.

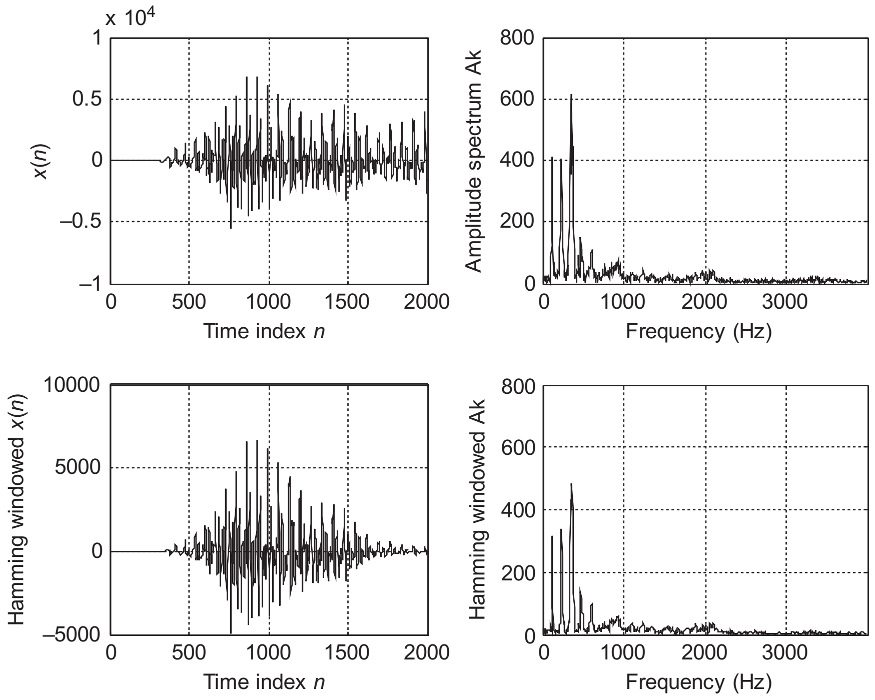

FIG. 4.23

Comparison of a one-sided spectrum without using a window function and a one-sided spectrum using the Hamming window for speech data.

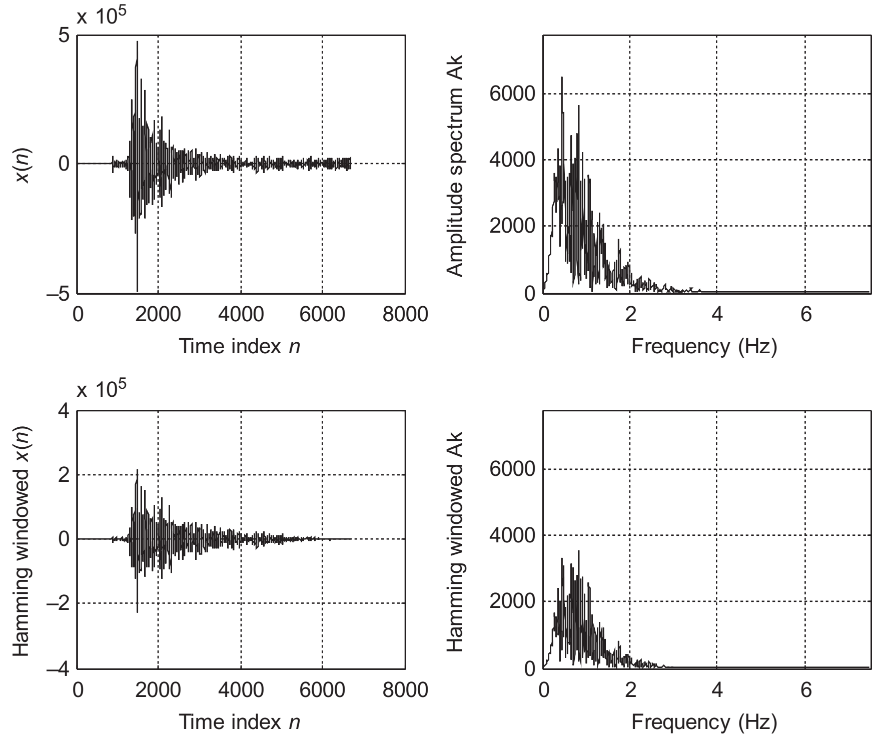

FIG. 4.24

Comparison of a one-sided spectrum without using a window function and a one-sided spectrum using the Hamming window for seismic data.

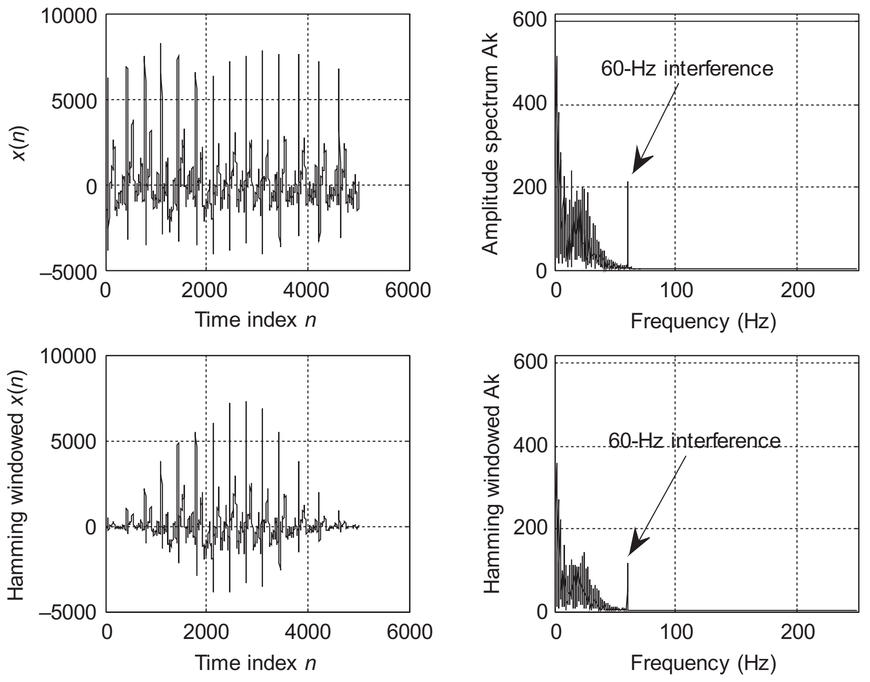

FIG. 4.25

Comparison of a one-sided spectrum without using a window function and a one-sided spectrum using the Hamming window for ECG data.

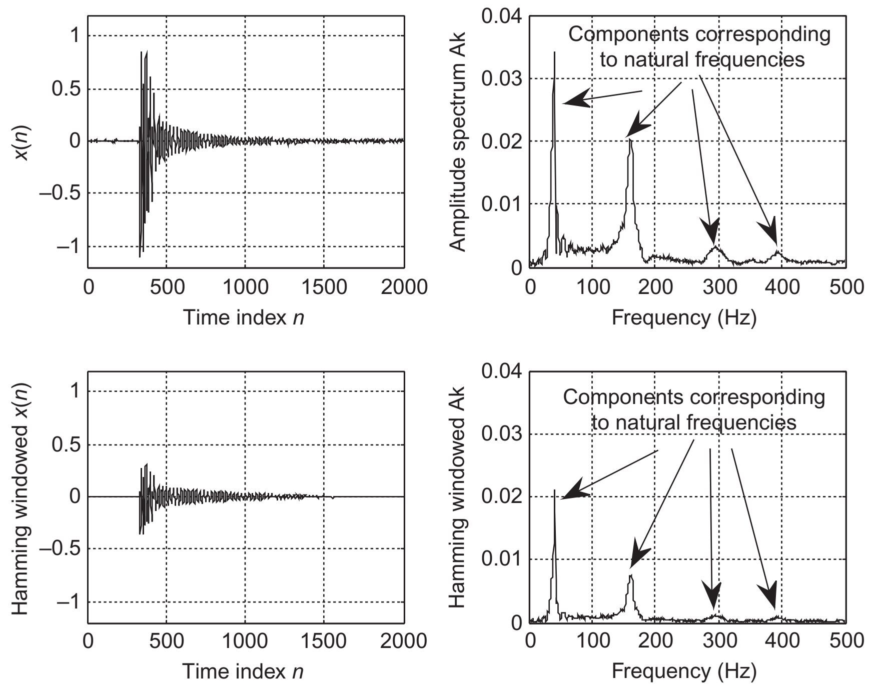

FIG. 4.26

Comparison of a one-sided spectrum without using a window function and a one-sided spectrum using the Hamming window for vibration signal.

Fig. 4.26 shows a vibration signal and its spectrum. The vibration signal is captured using an accelerometer sensor attached to a simple supported beam while an impulse force is applied to the location which is close to the middle of the beam. The sampling rate is 1 kHz. As shown in Fig. 4.26, four dominant modes (natural frequencies corresponding to locations of spectral peaks) can be easily identified from the displayed spectrum.

We now present another practical example for vibration signature analysis of the defected gear tooth described in Section 1.3.5. Fig. 4.27 shows a gearbox containing two straight bevel gears with a transmission ratio of 1.5:1 and the numbers of teeth on the pinion and gear are 18 and 27. The vibration data is collected by an accelerometer installed on the top of the gearbox. The data acquisition system uses a sampling rate of 12.8 kHz. The meshing frequency is determined as $f_m = f_i$(rpm[revolutions per minute]) $\times 18/60 = 300$ Hz, where the input shaft frequency is $f_i = 1000\text{rpm} = 16.67$Hz. Fig. 4.28 shows the baseline vibration signal and its spectrum under the excellent condition, and vibration signal and its spectrum under the different damage severity levels (five levels conducted by SpectraQuest, Inc.). As we can see, the baseline spectrum contains the meshing frequency component of 300 Hz and a sideband frequency component of 283.33 Hz (300 – 16.67). The vibration signatures for the damaged pinions (severity level 1: lightly chipped; severity level 4: heavily chipped; severity level 5: missing tooth) are included in Figs. 4.29–4.31. We can observe that the sidebands

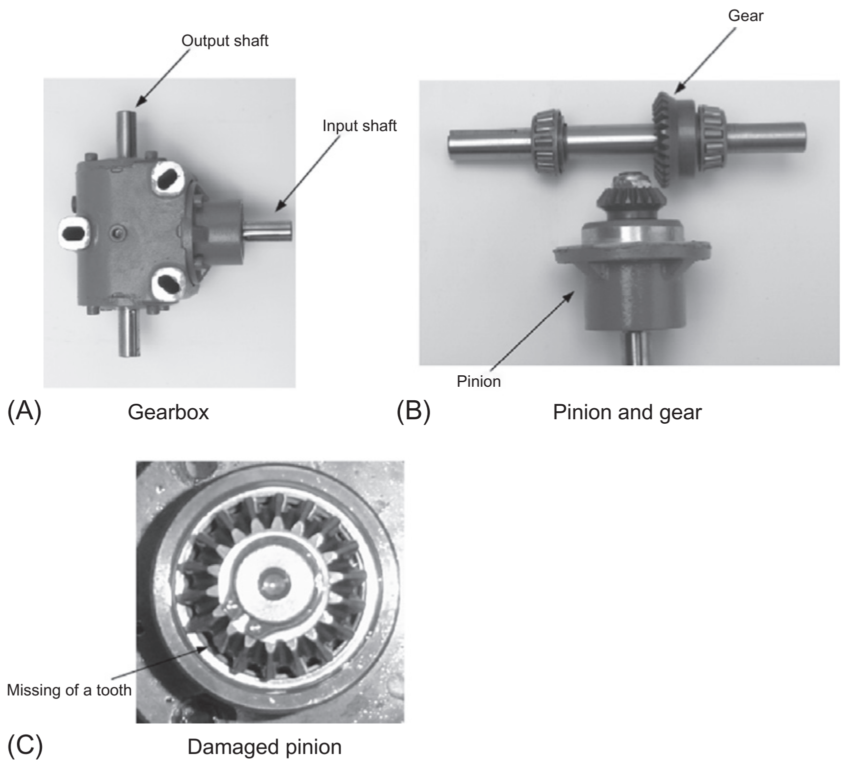

FIG. 4.27

Vibration signature analysis of a gearbox: (A) Gearbox, (B) Pinion and gear, (C) Damaged pinion.

Courtesy of SpectaQuest, Inc.

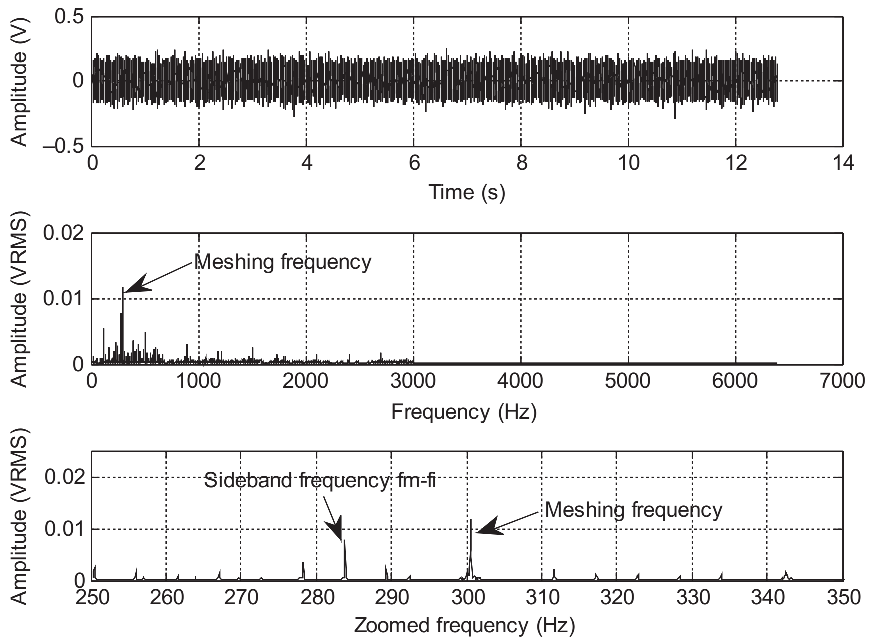

FIG. 4.28

Vibration signal and spectrum from the good condition gearbox.

Data provided by SpectaQuest, Inc.

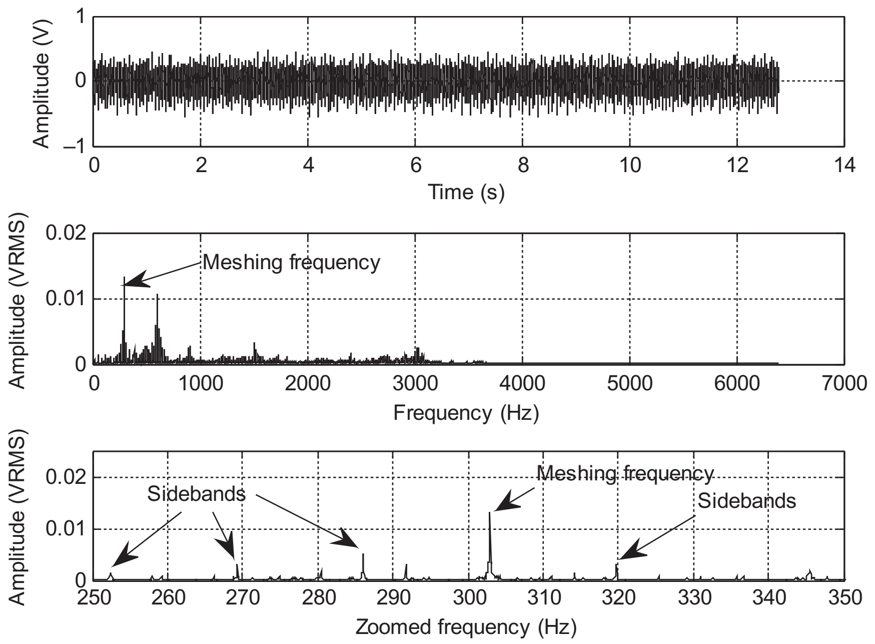

FIG. 4.29

Vibration signal and spectrum for damage severity level 1.

Data provided by SpectaQuest, Inc.

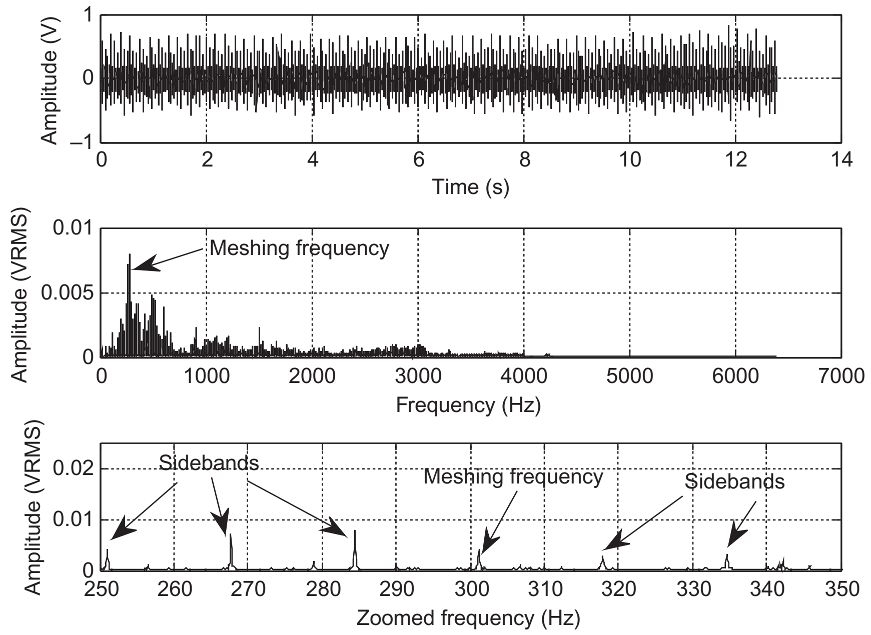

FIG. 4.30

Vibration signal and spectrum for damage severity level 4.

Data provided by SpectaQuest, Inc.

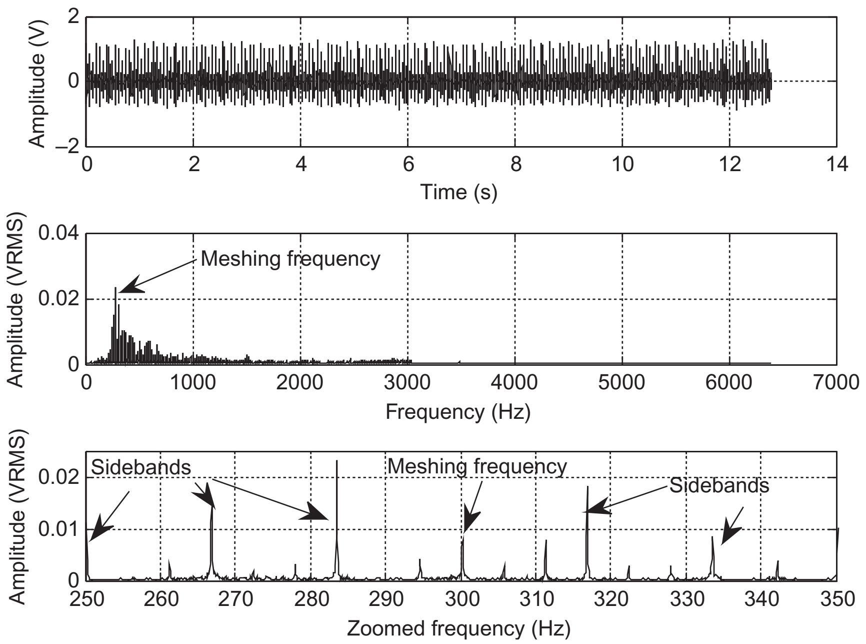

FIG. 4.31

Vibration signal and spectrum for damage severity level 5.

Data provided by SpectaQuest, Inc.

($f_m \pm f_i$, $f_m \pm 2f_i \ldots$) become more dominant when the severity level increases. Hence, the spectral information is very useful for monitoring the health condition of a gearbox.

4.5 FAST FOURIER TRANSFORM

Now we study FFT in detail. The FFT is a very efficient algorithm in computing DFT coefficients and can reduce a very large amount of computational complexity (multiplications). Without loss of generality, we consider the digital sequence $x(n)$ consisting of 2^m samples, where m is a positive integer, that is, the number of samples of the digital sequence $x(n)$ is a power of 2, $N=2, 4, 8, 16$, etc. If $x(n)$ does not contain 2^m samples, then we simply append it with zeros until the number of the appended sequence is a power of 2.

In this section, we focus on two formats. One is called the decimation-in-frequency algorithm, while the other is the decimation-in-time algorithm. They are referred to as the *radix-2* FFT algorithms. Other types of FFT algorithms are the radix-4 and the split radix and their advantages can be explored (see Proakis and Manolakis, 1996).

4.5.1 METHOD OF DECIMATION-IN-FREQUENCY

We begin with the definition of DFT studied in the opening section in this chapter as follows:

$$X(k)=\sum_{n=0}^{N-1}x(n)W_N^{kn} \quad \text{for} \quad k=0,1,\cdots,N-1, \tag{4.38}$$

where $W_N=e^{-j\frac{2\pi}{N}}$ is the twiddle factor, and $N=2$, 4, 8, 16, ⋯, Eq. (4.38) can be expanded as

$$X(k)=x(0)+x(1)W_N^k+\cdots+x(N-1)W_N^{k(N-1)}. \tag{4.39}$$

Again, if we split Eq. (4.39) into

$$\begin{aligned}X(k)=x(0)+x(1)W_N^k+\cdots+x\left(\frac{N}{2}-1\right)W_N^{k(N/2-1)}\\ +x\left(\frac{N}{2}\right)W^{kN/2}+\cdots+x(N-1)W_N^{k(N-1)},\end{aligned} \tag{4.40}$$

then we can rewrite as a sum of the following two parts

$$X(k)=\sum_{n=0}^{(N/2)-1}x(n)W_N^{kn}+\sum_{n=N/2}^{N-1}x(n)W_N^{kn}. \tag{4.41}$$

Modifying the second term in Eq. (4.41) yields

$$X(k)=\sum_{n=0}^{(N/2)-1}x(n)W_N^{kn}+W_N^{(N/2)k}\sum_{n=0}^{(N/2)-1}x\left(n+\frac{N}{2}\right)W_N^{kn}. \tag{4.42}$$

Recall $W_N^{N/2}=e^{-j\frac{2\pi(N/2)}{N}}=e^{-j\pi}=-1$; then we have

$$X(k)=\sum_{n=0}^{(N/2)-1}\left(x(n)+(-1)^kx\left(n+\frac{N}{2}\right)\right)W_N^{kn}. \tag{4.43}$$

Now letting $k=2m$ as an even number achieves

$$X(2m)=\sum_{n=0}^{(N/2)-1}\left(x(n)+x\left(n+\frac{N}{2}\right)\right)W_N^{2mn}, \tag{4.44}$$

while substituting $k=2m+1$ as an odd number yields

$$X(2m+1)=\sum_{n=0}^{(N/2)-1}\left(x(n)-x\left(n+\frac{N}{2}\right)\right)W_N^nW_N^{2mn}. \tag{4.45}$$

Using the fact that $W_N^2=e^{-j\frac{2\pi\times2}{N}}=e^{-j\frac{2\pi}{(N/2)}}=W_{N/2}$, it follows that

$$X(2m)=\sum_{n=0}^{(N/2)-1}a(n)W_{N/2}^{mn}=\text{DFT}\{a(n) \ \text{ with } \ (N/2)\text{points}\}, \tag{4.46}$$

$$X(2m+1)=\sum_{n=0}^{(N/2)-1}b(n)W_N^nW_{N/2}^{mn}=\text{DFT}\left\{b(n)W_N^n \ \text{ with } \ (N/2)\text{points}\right\}, \tag{4.47}$$

where $a(n)$ and $b(n)$ are introduced and expressed as

$$a(n) = x(n) + x\left(n + \frac{N}{2}\right), \quad \text{for} \quad n = 0, 1 \cdots, \frac{N}{2} - 1, \tag{4.48}$$

$$b(n) = x(n) - x\left(n + \frac{N}{2}\right), \quad \text{for} \quad n = 0, 1, \cdots, \frac{N}{2} - 1. \tag{4.49}$$

Eqs. (4.38), (4.46), and (4.47) can be summarized as

$$\text{DFT}\{x(n) \text{ with } N \text{ points}\} = \begin{cases} \text{DFT}\{a(n) \text{ with} (N/2) \text{points}\} \\ \text{DFT}\{b(n) W_N^n \text{ with} (N/2) \text{points}\} \end{cases} \tag{4.50}$$

The computation process can be illustrated in Fig. 4.32.

As shown in this figure, there are three graphical operations, which are illustrated Fig. 4.33.

If we continue the process described in Fig. 4.32, we obtain the block diagrams shown in Figs. 4.34 and 4.35.

Fig. 4.35 illustrates the FFT computation for the 8-point DFT, where there are 12 complex multiplications. This is a big saving as compared with the eight-point DFT with 64 complex multiplications. For a data length of N, the number of complex multiplications for DFT and FFT, respectively, are determined by

$$\text{Complex multiplications of DFT} = N^2, \text{and}$$

$$\text{Complex multiplications of FFT} = \frac{N}{2} \log_2(N).$$

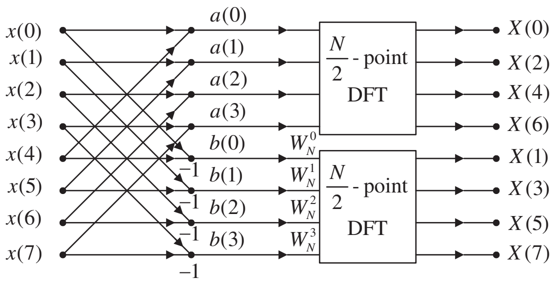

FIG. 4.32

The first iteration of the eight-point FFT.

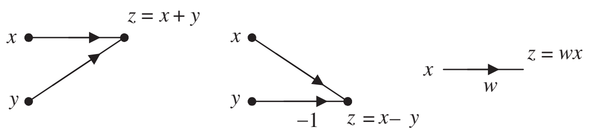

FIG. 4.33

Definitions of the graphical operations.

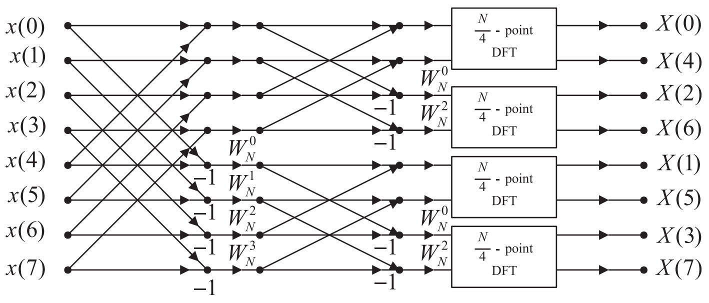

FIG. 4.34

The second iteration of the eight-point FFT.

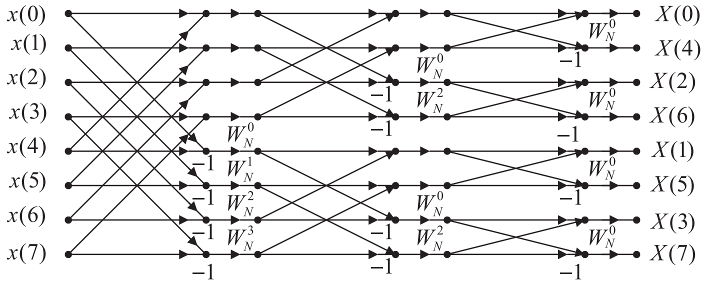

FIG. 4.35

Block diagram for the eight-point FFT (total 12 multiplications).

To see the effectiveness of FFT, let us consider a sequence with 1024 data points. Applying DFT will require $1024 \times 1024 = 1,\ 048,\ 576$ complex multiplications; however, applying FFT will need only $(1024/2)\log_2(1024) = 5120$ complex multiplications. Next, the index (bin number) of the eight-point DFT coefficient $X(k)$ becomes to be 0, 4, 2, 6, 1, 5, 3, and 7, respectively, which are not in the natural order. This can be fixed by index matching. The index matching between the input sequence and output frequency bin number by applying reversal bits is described in Table 4.2.

Table 4.2 Index Mapping for Fast Fourier Transform

Input Data	Index Bits	Reversal Bits	Output Data
$x(0)$	000	000	$X(0)$
$x(1)$	001	100	$X(4)$
$x(2)$	010	010	$X(2)$
$x(3)$	011	110	$X(6)$
$x(4)$	100	001	$X(1)$
$x(5)$	101	101	$X(5)$
$x(6)$	110	011	$X(3)$
$x(7)$	111	111	$X(7)$

Binary	Index	1st split	2nd split	3rd split	Bit reversal
000	0	0	0	0	000
001	1	2	4	4	100
010	2	4	2	2	010
011	3	6	6	6	011
100	4	1	1	1	001
101	5	3	5	5	101
110	6	5	3	3	011
111	7	7	7	7	111

FIG. 4.36

Bit reversal process in FFT.

Fig. 4.36 explains the bit reversal process. First, the input data with indices 0, 1, 2, 3, 4, 5, 6, and 7 are split into two parts. The first half contains even indices—0, 2, 4, 6—while the second half contains odd indices. The first half with indices 0, 2, 4, and 6 at the first iteration continues to split into even indices 0, 4 and odd indices 2, 6 as shown in the second iteration. The second half with indices1, 3, 5, and 7 at the first iteration is split into even indices 1, 5 and odd indices 3, 7 in the second iteration. The splitting process continues till the end at the third iteration. The bit patterns of the output data indices are just the respective reversed bit patterns of the input data indices.

Although Fig. 4.36 illustrates the case of an eight-point FFT, this bit reversal process works as long as N is a power of 2.

MATLAB Program for implementing FFT using decimation-in-frequency method is listed below:

Program 4.3 MATLAB program for FFT with decimation-in-frequency

```
function Xk= fftdinf(x)
% FFT using decimation-in-frequency method
%
XX=fftdinf2(x);
k=bitrev(1:1:length(XX));
Xk=XX(k);
end
function Xk = fftdinf2(x)
% FFT using decimation-in-frequency method
M=ceil(log2(length(x)));
x=[x zeros(1,2^M-length(x))]; % padding zeros to have a length of power of 2
N=length(x);
if (N==1)
Xk=x;
else
a=x(1:N/2)+x(N/2+1:N);
b=x(1:N/2)-x(N/2+1:N);
Xk=[fftdinf2(a) fftdinf2(b.*exp(-2*pi*j*[0:1:N/2-1]/N))];
```

```
end
end
function k = bitrev(x)
% bit reversal in terms of the integer
% x=1:1:2^M
N=length(x);
if N==1
k=x;
else
k=[bitrev(x(1:2:N)) bitrev(x(2:2:N))];
N=N/2;
end
end
```

The inverse FFT is defined as

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)W_N^{-kn} = \frac{1}{N}\sum_{k=0}^{N-1} X(k)\widetilde{W}_N^{kn}, \text{ for } k = 0, 1, \cdots, N-1. \tag{4.51}$$

On comparing Eq. (4.51) with Eq. (4.38), we note the following difference: the twiddle factor W_N is changed to $\widetilde{W}_N = W_N^{-1}$, and the sum is multiplied by a factor of $1/N$. Hence, by modifying the FFT block diagram as shown in Fig. 4.35, we achieve the inverse FFT block diagram shown in Fig. 4.37.

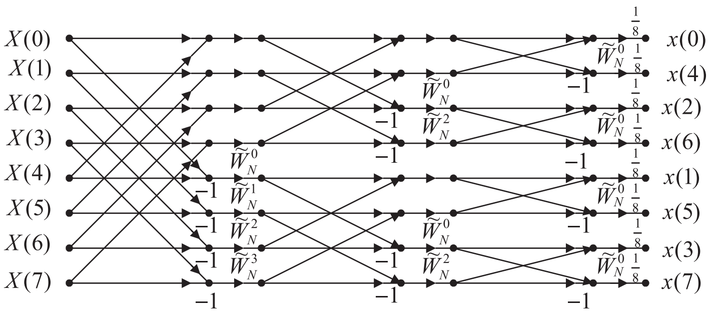

FIG. 4.37

Block diagram for the inverse of eight-point FFT.

EXAMPLE 4.12

Given a sequence $x(n)$ for $0 \le n \le 3$, where $x(0) = 1$, $x(1) = 2$, $x(2) = 3$, and $x(3) = 4$

(a) Evaluate its DFT $X(k)$ using the decimation-in-frequency FFT method.

(b) Determine the number of complex multiplications.

Solution:

(a) Using the FFT block diagram in Fig. 4.35, the result is shown in Fig. 4.38.

(b) From Fig. 4.38, the number of complex multiplications is four, which can also be determined by

$$\frac{N}{2}\log_2(N) = \frac{4}{2}\log_2(4) = 4.$$

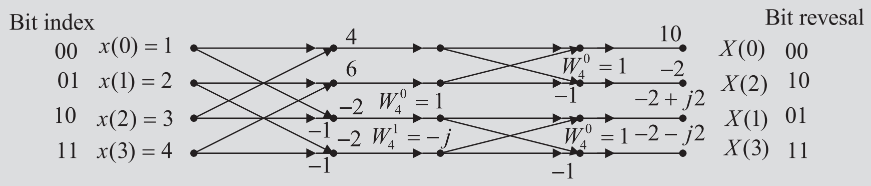

FIG. 4.38

Four-point FFT block diagram in Example 4.12.

EXAMPLE 4.13

Given the DFT sequence $X(k)$ for $0 \le k \le 3$ computed in Example 4.12, evaluate its inverse of DFT $x(n)$ using the decimation-in-frequency FFT method.

Solution:

Using the inverse FFT block diagram in Fig. 4.37, we have the result shown in Fig. 4.39.

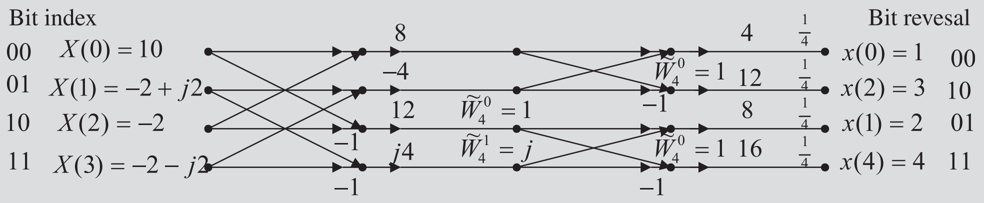

FIG. 4.39

Four-point inverse FFT block diagram in Example 4.13.

4.5.2 METHOD OF DECIMATION-IN-TIME

In this method, we split the input sequence $x(n)$ into the even indexed $x(2m)$ and $x(2m+1)$, each with N data points. Then Eq. (4.38) becomes

$$X(k) = \sum_{m=0}^{(N/2)-1} x(2m)W_N^{2mk} + \sum_{m=0}^{(N/2)-1} x(2m+1)W_N^k W_N^{2mk}, \text{ for } k = 0, 1, \cdots, N-1. \tag{4.52}$$

Using the relation $W_N^2 = W_{N/2}$, it follows that

$$X(k) = \sum_{m=0}^{(N/2)-1} x(2m) W_{N/2}^{mk} + W_N^k \sum_{m=0}^{(N/2)-1} x(2m+1) W_{N/2}^{mk}, \text{ for } k = 0, 1, \cdots, N-1. \tag{4.53}$$

Define new functions as

$$G(k) = \sum_{m=0}^{(N/2)-1} x(2m) W_{N/2}^{mk} = \text{DFT}\{x(2m) \text{with} (N/2) \text{points}\}, \tag{4.54}$$

$$H(k) = \sum_{m=0}^{(N/2)-1} x(2m+1) W_{N/2}^{mk} = \text{DFT}\{x(2m+1) \text{with} (N/2) \text{points}\}. \tag{4.55}$$

Note that

$$G(k) = G\left(k + \frac{N}{2}\right), \quad \text{for} \quad k = 0, 1, \cdots, \frac{N}{2} - 1, \tag{4.56}$$

$$H(k) = H\left(k + \frac{N}{2}\right), \quad \text{for} \quad k = 0, 1, \cdots, \frac{N}{2} - 1. \tag{4.57}$$

Substituting Eqs. (4.56) and (4.57) into Eq. (4.53) yields the first half frequency bins

$$X(k) = G(k) + W_N^k H(k), \quad \text{for} \quad k = 0, 1, \cdots, \frac{N}{2} - 1. \tag{4.58}$$

Considering the following fact and using Eqs. (4.56) and (4.57)

$$W_N^{(N/2+k)} = -W_N^k. \tag{4.59}$$

Then the second half of frequency bins can be computed as follows:

$$X\left(\frac{N}{2} + k\right) = G(k) - W_N^k H(k), \text{ for } k = 0, 1, \cdots, \frac{N}{2} - 1. \tag{4.60}$$

If we perform backward iterations, we can obtain the FFT algorithm. Procedure using Eqs. (4.58) and (4.60) is illustrated in Fig. 4.40, the block diagram for the eight-point FFT algorithm.

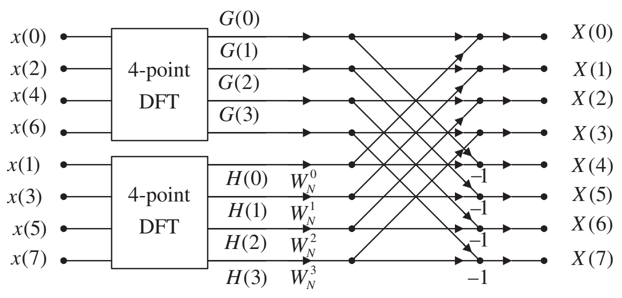

FIG. 4.40

First iteration.

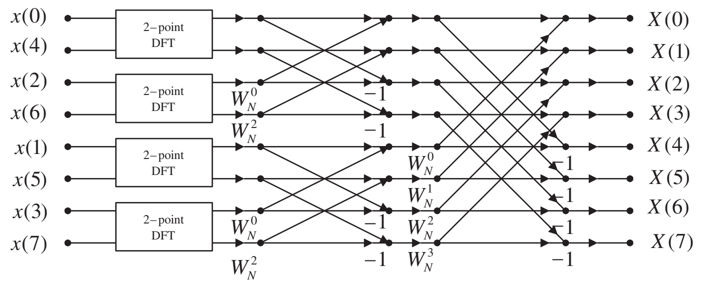

FIG. 4.41

Second iteration.

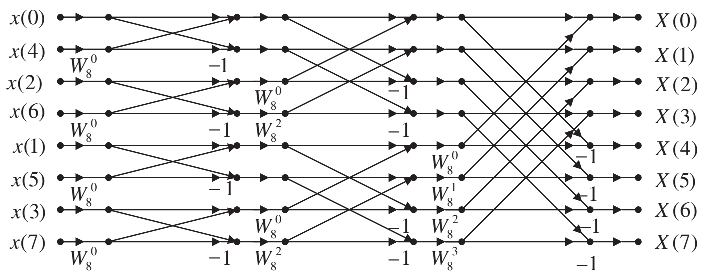

FIG. 4.42

Eight-point FFT algorithm using decimation-in-time (12 complex multiplications).

From a further computation, we obtain Fig. 4.41.

Finally, after three recursions, we end up with the block diagram in Fig. 4.42.

The index for each input sequence element can be achieved by bit reversal of the frequency index in a sequential order. MATLAB program for implementing FFT using decimation-in-time method is listed below:

Program 4.4 MATLAB program for FFT using decimation-in-time method

```
function Xk = fftdint(x)
% FFT using decimation-in-time method, no need for bit reversal algorithm
M=ceil(log2(length(x)));
x=[x zeros(1,2^M-length(x))]; %padding zeros to have a length of power of 2
N=length(x);
if (N==1)
Xk=x;
else
G=fftdint(x(1:2:N));
H=fftdint(x(2:2:N)).*exp(-2*pi*j*[0:1:N/2-1]/N);
Xk=[G+H G-H];
end
end
```

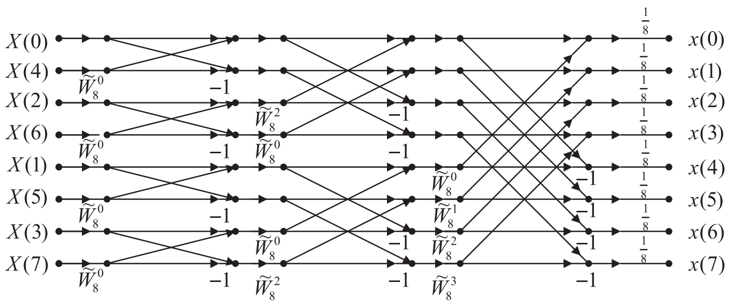

FIG. 4.43

The eight-point IFFT using decimation-in-time.

Similar to the decimation-in-frequency method, after changing W_N to $\widetilde{W}_N$ in Fig. 4.42 and multiplying the output sequence by a factor of $1/N$, we derive the inverse FFT block diagram for the eight-point inverse FFT in Fig. 4.43.

EXAMPLE 4.14

Given a sequence $x(n)$ for $0 \le n \le 3$, where $x(0) = 1$, $x(1) = 2$, $x(2) = 3$, and $x(3) = 4$, evaluate its DFT $X(k)$ using the decimation-in-time FFT method.

Solution:

Using the block diagram in Fig. 4.42 leads to the result shown in Fig. 4.44.

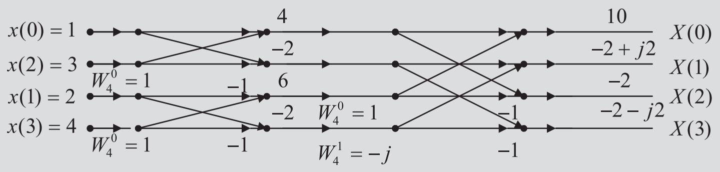

FIG. 4.44

The four-point FFT using decimation-in-time method.

EXAMPLE 4.15

Given the DFT sequence $X(k)$ for $0 \le k \le 3$ computed in Example 4.14, evaluate its inverse of DFT $x(n)$ using the decimation-in-time FFT method.

Solution:

Using the block diagram in Fig. 4.43 yields the result shown in Fig. 4.45

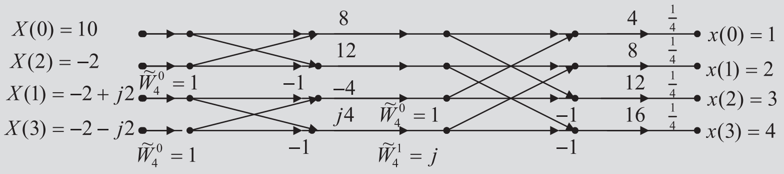

FIG. 4.45

The four-point IFFT using decimation-in-time method.

4.6 SUMMARY

1. The Fourier series coefficients for a periodic digital signal can be used to develop the DFT.
2. The DFT transforms a time sequence to the complex DFT coefficients, while the inverse DFT transforms DFT coefficients back to the time sequence.
3. The *frequency bin number* is the same as the frequency index. *Frequency resolution* is the frequency spacing between two consecutive frequency indices (two consecutive spectrum components).
4. The DFT coefficients for a given digital sequence are applied for computing the amplitude spectrum, power spectrum, or phase spectrum.
5. The spectrum calculated from all the DFT coefficients represents the signal frequency range from 0 Hz to the sampling rate. The spectrum beyond the folding frequency is equivalent to the negative-indexed spectrum from the negative folding frequency to 0 Hz. This two-sided spectrum can be converted into a single-sided spectrum by doubling alternating-current (AC) components from 0 Hz to the folding frequency and retaining the DC component as it is.
6. To reduce the burden of computing DFT coefficients, the FFT algorithm is used, which requires the data length to be a power of 2. Sometime zero padding is used to make up the data length. The zero padding actually does interpolation of the spectrum and does not carry any new information about the signal; even the calculated frequency resolution is smaller due to the zero padded longer length.
7. Applying a window function to the data sequence before DFT reduces the spectral leakage due to abrupt truncation of the data sequence when performing spectral calculation for a short sequence.
8. Two radix-2 FFT algorithms—decimation-in-frequency and decimation-in-time—are developed via the graphical illustrations.

4.7 PROBLEMS

4.1 Given a sequence $x(n)$ for $0 \le n \le 3$, where $x(0)=1$, $x(1)=1$, $x(2)=-1$, and $x(3)=0$, compute its DFT $X(k)$.

4.2 Given a sequence $x(n)$ for $0 \le n \le 3$, where $x(0)=4$, $x(1)=3$, $x(2)=2$, and $x(3)=1$, evaluate its DFT $X(k)$.

4.3 Given a sequence $x(n)$ for $0 \le n \le 3$, where $x(0)=0.2$, $x(1)=0.2$, $x(2)=-0.2$, and $x(3)=0$, compute its DFT $X(k)$.

4.4 Given a sequence $x(n)$ for $0 \le n \le 3$, where $x(0)=0.8$, $x(1)=0.6$, $x(2)=0.4$, and $x(3)=0.2$, evaluate its DFT $X(k)$.

4.5 Given the DFT sequence $X(k)$ for $0 \le k \le 3$ obtained in Problem 4.2, evaluate its inverse of DFT $x(n)$.

4.6 Given a sequence $x(n)$, where $x(0)=4$, $x(1)=3$, $x(2)=2$, and $x(3)=1$ with the last two data zero padded as $x(4)=0$, and $x(5)=0$, evaluate its DFT $X(k)$.

4.7 Given the DFT sequence $X(k)$ for $0 \le k \le 3$ obtained in Problem 4.4, evaluate its inverse of DFT $x(n)$.

4.8 Given a sequence $x(n)$, where $x(0)=0.8$, $x(1)=0.6$, $x(2)=0.4$, and $x(3)=0.2$ with the last two data zero padded as $x(4)=0$, and $x(5)=0$, evaluate its DFT $X(k)$.

4.9 Using the DFT sequence $X(k)$ for $0 \le k \le 5$ computed in Problem 4.6, evaluate the inverse of DFT for $x(0)$ and $x(4)$.

4.10 Consider a digital sequence sampled at the rate of 20,000 Hz. If we use the 8000-point DFT to compute the spectrum, determine

(a) the frequency resolution.

(b) the folding frequency in the spectrum.

4.11 Using the DFT sequence $X(k)$ for $0 \le k \le 5$ computed in Problem 4.8, evaluate the inverse of DFT for $x(0)$ and $x(4)$.

4.12 Consider a digital sequence sampled at the rate of 16,000 Hz. If we use the 4000-point DFT to compute the spectrum, determine

(a) the frequency resolution.

(b) the folding frequency in the spectrum.

4.13 We use the DFT to compute the amplitude spectrum of a sampled data sequence with a sampling rate $f_s = 2000$ Hz. It requires the frequency resolution to be less than 0.5 Hz. Determine the number of data points used by the FFT algorithm and actual frequency resolution in Hz, assuming that the data samples are available for selecting the number of data points.

4.14 Given the sequence in Fig. 4.46 and assuming $f_s = 100$ Hz, compute the amplitude spectrum, phase spectrum, and power spectrum.

4.15 Compute the following window functions for a size of 8:

(a) Hamming window function.

(b) Hanning window function.

4.16 Given the following data sequence with a length of 6

$$x(0)=0, x(1)=1, x(2)=0, x(3)=-1, x(4)=0, x(5)=1,$$

compute the windowed sequence $x_w(n)$ using

(a) Triangular window function.

(b) Hamming window function.

(c) Hanning window function.

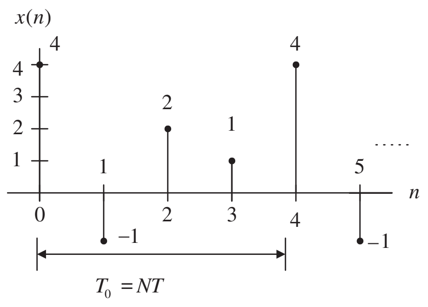

FIG. 4.46

Data sequence in Problem 4.14.

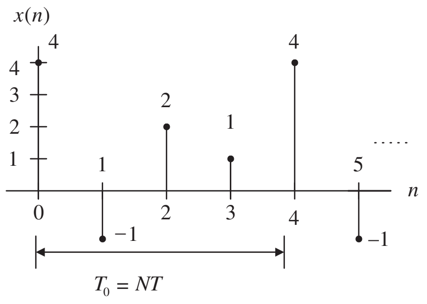

FIG. 4.47

Data sequence in Problem 4.19.

4.17 Compute the following window functions for a size of 10:

(a) Hamming window function.

(b) Hanning window function.

4.18 Given the following data sequence with a length of 6

$$x(0)=0, x(1)=0.2, x(2)=0, x(3)=-0.2, x(4)=0, x(5)=0.2$$

compute the windowed sequence $x_w(n)$ using

(a) Triangular window function.

(b) Hamming window function.

(c) Hanning window function.

4.19 Given a sequence in Fig. 4.47 where $f_s = 100\,\text{Hz}$ and $T = 0.01$ s, compute the amplitude spectrum, phase spectrum, and power spectrum using

(a) Triangular window.

(b) Hamming window.

(c) Hanning window.

4.20 Given a sinusoid

$$x(n) = 2 \times \sin\left(2000 \times 2\pi \times \frac{n}{8000}\right)$$

obtained by using the sampling rate of $f_s = 8000\,\text{Hz}$, we apply the DFT to compute the amplitude spectrum.

(a) Determine the frequency resolution when the data length is 100 samples. Without using the window function, is there any spectral leakage in the computed spectrum? Explain.

(b) Determine the frequency resolution when the data length is 73 samples. Without using the window function, is there any spectral leakage in the computed spectrum? Explain.

4.21 Given a sequence $x(n)$ for $0 \le n \le 3$, where $x(0)=4$, $x(1)=3$, $x(2)=2$, and $x(3)=1$, evaluate its DFT $X(k)$ using the decimation-in-frequency FFT method, and determine the number of complex multiplications.

4.22 Given the DFT sequence $X(k)$ for $0 \le k \le 3$ obtained in Problem 4.21, evaluate its inverse DFT $x(n)$ using the decimation-in-frequency FFT method.

4.23 Given a sequence $x(n)$ for $0 \le n \le 3$, where $x(0) = 0.8$, $x(1) = 0.6$, $x(2) = 0.4$, and $x(3) = 0.2$, evaluate its DFT $X(k)$ using the decimation-in-frequency FFT method, and determine the number of complex multiplications.

4.24 Given the DFT sequence $X(k)$ for $0 \le k \le 3$ obtained in Problem 4.23, evaluate its inverse DFT $x(n)$ using the decimation-in-frequency FFT method.

4.25 Given a sequence $x(n)$ for $0 \le n \le 3$, where $x(0) = 4$, $x(1) = 3$, $x(2) = 2$, and $x(3) = 1$, evaluate its DFT $X(k)$ using the decimation-in-time FFT method, and determine the number of complex multiplications.

4.26 Given the DFT sequence $X(k)$ for $0 \le k \le 3$ computed in Problem 4.25, evaluate its inverse DFT $x(n)$ using the decimation-in-time FFT method.

4.27 Given a sequence $x(n)$ for $0 \le n \le 3$, where $x(0) = 0.8$, $x(1) = 0.4$, $x(2) = -0.4$, and $x(3) = -0.2$, evaluate its DFT $X(k)$ using the decimation-in-time FFT method, and determine the number of complex multiplications.

4.28 Given the DFT sequence $X(k)$ for $0 \le k \le 3$ computed in Problem 4.27, evaluate its inverse DFT $x(n)$ using the decimation-in-time FFT method.

Computer Problems with MATLAB

Use MATLAB to solve Problems 4.29–4.30.

4.29 Given three sinusoids with the following amplitudes and phases:

$$\mathrm{x}_1(\mathrm{t}) = 5\cos(2\pi(500)\mathrm{t})$$

$$x_2(t) = 5\cos(2\pi(1200)t + 0.25\pi)$$

$$x_3(t) = 5\cos(2\pi(1800)t + 0.5\pi)$$

(a) Create a MATLAB program to sample each sinusoid and generate a sum of three sinusoids, that is, $x(n) = x_1(n) + x_2(n) + x_3(n)$, using a sampling rate of 8000 Hz, and plot the sum $x(n)$ over a range of time that will exhibit approximately 0.1 s.

(b) Use the MATLAB function fft() to compute DFT coefficients, and plot and examine the spectrum of the signal $x(n)$.

4.30 Using the sum of sinusoids in Problem 4.29:

(a) Generate the sum of sinusoids for 240 samples using a sampling rate of 8000 Hz.

(b) Write a MATLAB program to compute and plot the amplitude spectrum of the signal $x(n)$ with the FFT and using each of the following window functions:

(1) Rectangular window (no window).

(2) Triangular window.

(3) Hamming window.

(c) Examine the effect of spectral leakage for each window used in (b).

MATLAB Projects

4.31 Signal spectral analysis:

Given below are four practical signals, compute their one-sided spectra and create their time domain plots and spectral plots, respectively:

(a) Speech signal ("speech.dat"), sampling rate = 8000 Hz.
From the spectral plot, identify the first five formants.

(b) ECG signal (“ecg.dat”), sampling rate = 500 Hz.
From the spectral plot, identify the 60 Hz-interference component.
(c) Seismic data (“seismic.dat”), sampling rate = 15 Hz.
From the spectral plot, determine the dominant frequency component.
(d) Vibration signal of the acceleration response from a simple supported beam (“vbrdata.dat”), sampling rate = 1000 Hz.
From the spectral plot, determine the four dominant frequencies (modes).

4.32 Vibration signature analysis:
The acceleration signals measured from the gearbox can be used for monitoring the condition of the gears inside the gearbox. The early diagnosis of the gear condition can prevent the future catastrophic failure of the system. Given the following measurements and specifications (courtesy of SpectraQuest, Inc.):
(a) The input shaft has a speed of 1000 rpm and meshing frequency is approximately 300 Hz.
(b) Data specifications:

Sampling rate = 12.8 kHz
v0.dat: healthy condition
v1.dat: damage severity level 1 (lightly chipped gear)
v2.dat: damage severity level 2 (moderately chipped gear)
v3.dat: damage severity level 3 (chipped gear)
v4.dat: damage severity level 4 (heavily chipped gear)
v5.dat: damage severity level 5 (missing tooth)
Investigate the spectrum for each measurement and identify sidebands.
For each measurement, determine the ratio of the largest sideband amplitude over the amplitude of meshing frequency and investigate the ratio effect related to the damage severity.

Advanced Problems

4.33 Show that
a. $W_N^N = 1$; b. $W_N^{N/2} = -1$; c. $W_N^{N/4} = -j$; d. $W_N^{-m} = W_N^{N-m}$.

4.34 For a real sequence $x(n)$ defined for $0 \leq n < N$, show that

$$X(N-k) = X^*(k).$$

4.35 For $x(n) = \delta(n–m)$ defined for $0 \leq n$, $m < N$, show that

$$X(k) = W_N^{km}.$$

4.36 For $x(n) = 1$ defined for $0 \leq n < N$, show that

$$X(k) = \begin{cases} N & k = 0 \\ 0 & \text{elsewhere} \end{cases}.$$

4.37 Consider a sequence of $x(n)$ defined for $0 \leq n < N$, where N = even

$$x(n) = \begin{cases} 1 & n = \text{even} \\ 0 & n = \text{odd} \end{cases},$$

show that

$$X(k)=\begin{cases} N/2 & k=0, k=N/2 \\ 0 & \text{elsewhere} \end{cases}.$$

4.38 A sequence is shown below:

$$x(n)=\frac{1}{2}\left(1-\cos\frac{2\pi n}{N}\right) \text{ for } 0\le n<N,$$

show that the DFT of $x(n)$ is given by

$$X(k)=\begin{cases} N/2 & k=0 \\ -N/4 & k=1, k=N-1 \\ 0 & \text{elsewhere} \end{cases}.$$

4.39 For $x(n)=n$ defined for $0\le n<N$, show that

$$X(k)=\begin{cases} N(N-1)/2 & k=0 \\ -N/\left(1-W_N^k\right) & \text{elsewhere} \end{cases}.$$

4.40 Show that

$$\frac{1}{N}\sum_{n=0}^{N-1}\cos\left[\frac{\pi k(2n+1)}{2N}\right]\cos\left[\frac{\pi m(2n+1)}{2N}\right]=\begin{cases} 1 & k=m=0 \\ 1/2 & k=m\ne 0 \\ 0 & k\ne m \end{cases}.$$

4.41 For a real sequence $x(n)$ defined for $0\le n<N$, its discrete-cosine transform is given by

$$X_{\mathrm{DCT}}(k)=\sum_{n=0}^{N-1}2x(n)\cos\left[\frac{\pi k(2n+1)}{2N}\right], 0\le k<N.$$

Show that

$$x(n)=\frac{1}{N}\sum_{k=0}^{N-1}a(k)X_{\mathrm{DCT}}(k)\cos\left[\frac{\pi k(2n+1)}{2N}\right], 0\le n<N,$$

where $\alpha(k)=\begin{cases} 1/2 & k=0 \\ 1 & 0<k<N \end{cases}.$

4.42 For a real sequence $x(n)$ defined for $0\le n<N$, its discrete Harley transform is given by

$$X_{\mathrm{DHT}}(k)=\sum_{n=0}^{N-1}2x(n)\left[\cos\left(\frac{2\pi nk}{N}\right)+\sin\left(\frac{2\pi nk}{N}\right)\right], 0\le k<N,$$

show that

$$x(n)=\frac{1}{N}\sum_{k=0}^{N-1}X_{\mathrm{DHT}}(k)\left[\cos\left(\frac{2\pi nk}{N}\right)+\sin\left(\frac{2\pi nk}{N}\right)\right], 0\le n<N.$$

4.43 Modify Program 4.3 to create a MATLAB function [ifftdinf()] for the inverse of FFT using the decimation-in-frequency method.

4.44 Modify Program 4.4 to create a MATLAB function [ifftdint()] for the inverse of FFT using the decimation-in-time method.