TABLE 7.2 Properties of the DFT

Property	Time Domain	Frequency Domain
Notation	$x(n), y(n)$	$X(k), Y(k)$
Periodicity	$x(n) = x(n+N)$	$X(k) = X(k+N)$
Linearity	$a_1x_1(n) + a_2x_2(n)$	$a_1X_1(k) + a_2X_2(k)$
Time reversal	$x(N-n)$	$X(N-k)$
Circular time shift	$x((n-l))_N$	$X(k)e^{-j2\pi kl/N}$
Circular frequency shift	$x(n)e^{j2\pi ln/N}$	$X((k-l))_N$
Complex conjugate	$x^*(n)$	$X^*(N-k)$
Circular convolution	$x_1(n) \circledN x_2(n)$	$X_1(k)X_2(k)$
Circular correlation	$x(n) \circledN y^*(-n)$	$X(k)Y^*(k)$
Multiplication of two sequences	$x_1(n)x_2(n)$	$\frac{1}{N}X_1(k) \circledN X_2(k)$
Parseval's theorem	$\sum_{n=0}^{N-1} x(n)y^*(n)$	$\frac{1}{N}\sum_{k=0}^{N-1} X(k)Y^*(k)$

and

$$\tilde{r}_{xy}(l) = \frac{1}{N}\sum_{k=0}^{N-1} \tilde{R}_{xy}(k)e^{j2\pi kl/N}$$

$$= \frac{1}{N}\sum_{k=0}^{N-1} X(k)Y^*(k)e^{j2\pi kl/N}$$

Hence (7.2.50) follows by evaluating the IDFT at $l = 0$.

The expression in (7.2.50) is the general form of Parseval's theorem. In the special case where $y(n) = x(n)$, (7.2.50) reduces to

$$\sum_{n=0}^{N-1} |x(n)|^2 = \frac{1}{N}\sum_{k=0}^{N-1} |X(k)|^2 \qquad (7.2.51)$$

which expresses the energy in the finite-duration sequence $x(n)$ in terms of the frequency components $\{X(k)\}$.

The properties of the DFT given above are summarized in Table 7.2.

7.3 Linear Filtering Methods Based on the DFT

Since the DFT provides a discrete frequency representation of a finite-duration sequence in the frequency domain, it is interesting to explore its use as a computational tool for linear system analysis and, especially, for linear filtering. We have already established that a system with frequency response $H(\omega)$, when excited with an input signal that has a spectrum $X(\omega)$, possesses an output spectrum $Y(\omega) = X(\omega)H(\omega)$. The output sequence $y(n)$ is determined from its spectrum via the inverse Fourier transform. Computationally, the problem with this frequency-domain approach is that $X(\omega)$, $H(\omega)$, and $Y(\omega)$ are functions of the continuous variable ω. As a consequence, the computations cannot be done on a digital computer, since the computer can only store and perform computations on quantities at discrete frequencies.

On the other hand, the DFT does lend itself to computation on a digital computer. In the discussion that follows, we describe how the DFT can be used to perform linear filtering in the frequency domain. In particular, we present a computational procedure that serves as an alternative to time-domain convolution. In fact, the frequency-domain approach based on the DFT is computationally more efficient than time-domain convolution due to the existence of efficient algorithms for computing the DFT. These algorithms, which are described in Chapter 8, are collectively called fast Fourier transform (FFT) algorithms.

7.3.1 Use of the DFT in Linear Filtering

In the preceding section it was demonstrated that the product of two DFTs is equivalent to the circular convolution of the corresponding time-domain sequences. Unfortunately, circular convolution is of no use to us if our objective is to determine the output of a linear filter to a given input sequence. In this case we seek a frequency-domain methodology equivalent to linear convolution.

Suppose that we have a finite-duration sequence $x(n)$ of length L which excites an FIR filter of length M. Without loss of generality, let

$$\begin{aligned} x(n) &= 0, \qquad n < 0 \text{ and } n \geq L \\ h(n) &= 0, \qquad n < 0 \text{ and } n \geq M \end{aligned}$$

where $h(n)$ is the impulse response of the FIR filter.

The output sequence $y(n)$ of the FIR filter can be expressed in the time domain as the convolution of $x(n)$ and $h(n)$, that is

$$y(n) = \sum_{k=0}^{M-1} h(k)x(n-k) \tag{7.3.1}$$

Since $h(n)$ and $x(n)$ are finite-duration sequences, their convolution is also finite in duration. In fact, the duration of $y(n)$ is $L + M - 1$.

The frequency-domain equivalent to (7.3.1) is

$$Y(\omega) = X(\omega)H(\omega) \tag{7.3.2}$$

If the sequence $y(n)$ is to be represented uniquely in the frequency domain by samples of its spectrum $Y(\omega)$ at a set of discrete frequencies, the number of distinct samples must equal or exceed $L+M-1$. Therefore, a DFT of size $N \geq L+M-1$ is required to represent $\{y(n)\}$ in the frequency domain.

Now if

$$\begin{aligned} Y(k) &\equiv Y(\omega)|_{\omega=2\pi k/N}, \qquad k = 0, 1, \ldots, N-1 \\ &= X(\omega)H(\omega)|_{\omega=2\pi k/N}, \qquad k = 0, 1, \ldots, N-1 \end{aligned}$$

then

$$Y(k) = X(k)H(k), \qquad k = 0, 1, \ldots, N-1 \tag{7.3.3}$$

where $\{X(k)\}$ and $\{H(k)\}$ are the N-point DFTs of the corresponding sequences $x(n)$ and $h(n)$, respectively. Since the sequences $x(n)$ and $h(n)$ have a duration less than N, we simply pad these sequences with zeros to increase their length to N. This increase in the size of the sequences does not alter their spectra $X(\omega)$ and $H(\omega)$, which are continuous spectra, since the sequences are aperiodic. However, by sampling their spectra at N equally spaced points in frequency (computing the N-point DFTs), we have increased the number of samples that represent these sequences in the frequency domain beyond the minimum number (L or M, respectively).

Since the ($N = L + M - 1$)-point DFT of the output sequence $y(n)$ is sufficient to represent $y(n)$ in the frequency domain, it follows that the multiplication of the N-point DFTs $X(k)$ and $H(k)$, according to (7.3.3), followed by the computation of the N-point IDFT, must yield the sequence $\{y(n)\}$. In turn, this implies that the N-point circular convolution of $x(n)$ with $h(n)$ must be equivalent to the linear convolution of $x(n)$ with $h(n)$. In other words, by increasing the length of the sequences $x(n)$ and $h(n)$ to N points (by appending zeros), and then circularly convolving the resulting sequences, we obtain the same result as would have been obtained with linear convolution. Thus with zero padding, the DFT can be used to perform linear filtering.

The following example illustrates the methodology in the use of the DFT in linear filtering.

EXAMPLE 7.3.1

By means of the DFT and IDFT, determine the response of the FIR filter with impulse response

$$h(n) = \{\underset{\uparrow}{1}, 2, 3\}$$

to the input sequence

$$x(n) = \{\underset{\uparrow}{1}, 2, 2, 1\}$$

Solution. The input sequence has length $L = 4$ and the impulse response has length $M = 3$. Linear convolution of these two sequences produces a sequence of length $N = 6$. Consequently, the size of the DFTs must be at least six.

For simplicity we compute eight-point DFTs. We should also mention that the efficient computation of the DFT via the fast Fourier transform (FFT) algorithm is usually performed for a length N that is a power of 2. Hence the eight-point DFT of $x(n)$ is

$$\begin{aligned} X(k) &= \sum_{n=0}^{7} x(n)e^{-j2\pi kn/8} \\ &= 1 + 2e^{-j\pi k/4} + 2e^{-j\pi k/2} + e^{-j3\pi k/4}, \qquad k = 0, 1, \ldots, 7 \end{aligned}$$

This computation yields

$$X(0) = 6, \qquad X(1) = \frac{2+\sqrt{2}}{2} - j\left(\frac{4+3\sqrt{2}}{2}\right)$$

$$X(2) = -1 - j, \qquad X(3) = \frac{2-\sqrt{2}}{2} + j\left(\frac{4-3\sqrt{2}}{2}\right)$$

$$X(4) = 0, \qquad X(5) = \frac{2-\sqrt{2}}{2} - j\left(\frac{4-3\sqrt{2}}{2}\right)$$

$$X(6) = -1 + j, \qquad X(7) = \frac{2+\sqrt{2}}{2} + j\left(\frac{4+3\sqrt{2}}{2}\right)$$

The eight-point DFT of $h(n)$ is

$$\begin{aligned} H(k) &= \sum_{n=0}^{7} h(n)e^{-j2\pi kn/8} \\ &= 1 + 2e^{-j\pi k/4} + 3e^{-j\pi k/2} \end{aligned}$$

Hence

$$H(0) = 6, \qquad H(1) = 1 + \sqrt{2} - j\left(3 + \sqrt{2}\right), \qquad H(2) = -2 - j2$$

$$H(3) = 1 - \sqrt{2} + j\left(3 - \sqrt{2}\right), \qquad H(4) = 2$$

$$H(5) = 1 - \sqrt{2} - j\left(3 - \sqrt{2}\right), \qquad H(6) = -2 + j2$$

$$H(7) = 1 + \sqrt{2} + j\left(3 + \sqrt{2}\right)$$

The product of these two DFTs yields $Y(k)$, which is

$$Y(0) = 36, \qquad Y(1) = -14.07 - j17.48, \qquad Y(2) = j4, \qquad Y(3) = 0.07 + j0.515$$

$$Y(4) = 0, \qquad Y(5) = 0.07 - j0.515, \qquad Y(6) = -j4, \qquad Y(7) = -14.07 + j17.48$$

Finally, the eight-point IDFT is

$$y(n) = \sum_{k=0}^{7} Y(k)e^{j2\pi kn/8}, \qquad n = 0, 1, \ldots, 7$$

This computation yields the result

$$y(n) = \{\underset{\uparrow}{1}, 4, 9, 11, 8, 3, 0, 0\}$$

We observe that the first six values of $y(n)$ constitute the set of desired output values. The last two values are zero because we used an eight-point DFT and IDFT, when, in fact, the minimum number of points required is six.

Although the multiplication of two DFTs corresponds to circular convolution in the time domain, we have observed that padding the sequences $x(n)$ and $h(n)$ with a sufficient number of zeros forces the circular convolution to yield the same output sequence as linear convolution. In the case of the FIR filtering problem in Example 7.3.1, it is a simple matter to demonstrate that the six-point circular convolution of the sequences

$$h(n) = \{\underset{\uparrow}{1}, 2, 3, 0, 0, 0\} \tag{7.3.4}$$

$$x(n) = \{\underset{\uparrow}{1}, 2, 2, 1, 0, 0\} \tag{7.3.5}$$

results in the output sequence

$$y(n) = \{\underset{\uparrow}{1}, 4, 9, 11, 8, 3\} \tag{7.3.6}$$

which is the same sequence obtained from linear convolution.

It is important for us to understand the aliasing that results in the time domain when the size of the DFTs is smaller than $L+M-1$. The following example focuses on the aliasing problem.

EXAMPLE 7.3.2

Determine the sequence $y(n)$ that results from the use of four-point DFTs in Example 7.3.1.

Solution. The four-point DFT of $h(n)$ is

$$H(k) = \sum_{n=0}^{3} h(n)e^{-j2\pi kn/4}$$

$$H(k) = 1 + 2e^{-j\pi k/2} + 3e^{-jk\pi}, \qquad k = 0, 1, 2, 3$$

Hence

$$H(0) = 6, \qquad H(1) = -2 - j2, \qquad H(2) = 2, \qquad H(3) = -2 + j2$$

The four-point DFT of $x(n)$ is

$$X(k) = 1 + 2e^{-j\pi k/2} + 2e^{-j\pi k} + 1e^{-j3\pi k/2}, \qquad k = 0, 1, 2, 3$$

Hence

$$X(0) = 6, \qquad X(1) = -1 - j, \qquad X(2) = 0, \qquad X(3) = -1 + j$$

The product of these two four-point DFTs is

$$\hat{Y}(0) = 36, \qquad \hat{Y}(1) = j4, \qquad \hat{Y}(2) = 0, \qquad \hat{Y}(3) = -j4$$

The four-point IDFT yields

$$\hat{y}(n) = \frac{1}{4}\sum_{k=0}^{3} \hat{Y}(k)e^{j2\pi kn/4}, \qquad n = 0, 1, 2, 3$$

$$= \frac{1}{4}(36 + j4e^{j\pi n/2} - j4e^{j3\pi n/2})$$

Therefore,

$$\hat{y}(n) = \{\underset{\uparrow}{9}, 7, 9, 11\}$$

The reader can verify that the four-point circular convolution of $h(n)$ with $x(n)$ yields the same sequence $\hat{y}(n)$.

If we compare the result $\hat{y}(n)$, obtained from four-point DFTs, with the sequence $y(n)$ obtained from the use of eight-point (or six-point) DFTs, the time-domain aliasing effects derived in Section 7.2.2 are clearly evident. In particular, $y(4)$ is aliased into $y(0)$ to yield

$$\hat{y}(0) = y(0) + y(4) = 9$$

Similarly, $y(5)$ is aliased into $y(1)$ to yield

$$\hat{y}(1) = y(1) + y(5) = 7$$

All other aliasing has no effect, since $y(n) = 0$ for $n \geq 6$. Consequently, we have

$$\hat{y}(2) = y(2) = 9$$

$$\hat{y}(3) = y(3) = 11$$

Therefore, only the first two points of $\hat{y}(n)$ are corrupted by the effect of aliasing [i.e., $\hat{y}(0) \neq y(0)$ and $\hat{y}(1) \neq y(1)$]. This observation has important ramifications in the discussion of the following section, in which we treat the filtering of long sequences.

7.3.2 Filtering of Long Data Sequences

In practical applications involving linear filtering of signals, the input sequence $x(n)$ is often a very long sequence. This is especially true in some real-time signal processing applications concerned with signal monitoring and analysis.

Since linear filtering performed via the DFT involves operations on a block of data, which by necessity must be limited in size due to limited memory of a digital computer, a long input signal sequence must be segmented to fixed-size blocks prior to processing. Since the filtering is linear, successive blocks can be processed one at a time via the DFT, and the output blocks are fitted together to form the overall output signal sequence.

We now describe two methods for linear FIR filtering a long sequence on a block-by-block basis using the DFT. The input sequence is segmented into blocks and each block is processed via the DFT and IDFT to produce a block of output data. The output blocks are fitted together to form an overall output sequence which is identical to the sequence obtained if the long block had been processed via time-domain convolution.

The two methods are called the *overlap-save method* and the *overlap-add method*. For both methods we assume that the FIR filter has duration M. The input data sequence is segmented into blocks of L points, where, by assumption, $L >> M$ without loss of generality.