Chapter 4
The Discrete Fourier Transform

The DTFT of a discrete-time signal is a continuous function of the frequency (ω), and hence, the relation between $X(e^{j\omega})$ and $x(n)$ is not a computationally convenient representation. However, it is possible to develop an alternative frequency representation called the discrete Fourier transform (DFT) for finite duration sequences. The DFT is a discrete-time sequence with equal spacing in frequency. We first obtain the discrete-time Fourier series (DTFS) expansion of a periodic sequence. Next, we define the DFT of a finite length sequence and consider its properties in detail. We also show that the DTFS represents the DFT of a finite length sequence. Further, evaluation of linear convolution using the DFT is discussed. Finally, some fast Fourier transform (FFT) algorithms for efficient computation of DFT are described.

4.1 The Discrete-Time Fourier Series

If a sequence $x(n)$ is periodic with period N, then $x(n) = x(n+N)$ for all n. In analogy with the Fourier series representation of a continuous periodic signal, we can look for a representation of $x(n)$ in terms of the harmonics corresponding to the fundamental frequency of $(2\pi/N)$. Hence, we may write $x(n)$ in the form

$$x(n) = \sum_k b_k e^{j2\pi kn/N} \tag{4.1a}$$

It can easily be verified from Eq. (4.1a) that $x(n) = x(n + N)$. Also, we know that there are only N distinct values for $e^{j2\pi kn/N}$ corresponding to $k = 0, 1, \ldots, N - 1$, these being 1, $e^{j2\pi n/N}$, $\ldots$, $e^{j2\pi k(N-1)/N}$. Hence, we may rewrite Eq. (4.1a) as

© Springer Nature Singapore Pte Ltd. 2018

K. D. Rao and M. N. S. Swamy, *Digital Signal Processing*,
https://doi.org/10.1007/978-981-10-8081-4_4

$$x(n) = \sum_{k=0}^{N-1} a_k e^{j2\pi kn/N} \tag{4.1b}$$

It should be noted that the summation could be taken over any N consecutive values of k. Equation (4.1b) is called the discrete-time Fourier series (DTFS) of the periodic sequence $x(n)$ and a_k as the Fourier coefficients. We will now obtain the expression for the Fourier coefficients a_k. It can easily be shown that $\left\{e^{j2\pi kn/N}\right\}$ is an orthogonal sequence satisfying the relation

$$\sum_{n=0}^{N-1} e^{j2\pi kn/N} e^{-j2\pi ln/N} = \begin{cases} 0 & k \neq l \\ N & k = l \end{cases} \quad (0 \leq k,\ l \leq (N-1)) \tag{4.2}$$

Now, multiplying both sides of Eq. (4.1b) by $e^{-j2\pi ln/N}$ and summing over n between 0 and $(N-1)$, we get

$$\begin{aligned} \sum_{n=0}^{N-1} x(n) e^{-j2\pi ln/N} &= \sum_{n=0}^{N-1} \sum_{k=0}^{N-1} a_k e^{j2\pi kn/N} e^{-j2\pi ln/N} \\ &= \sum_{k=0}^{N-1} a_k \sum_{n=0}^{N-1} e^{j2\pi kn/N} e^{-j2\pi ln/N} \\ &= a_l N, \text{using } (4.2). \end{aligned}$$

Hence,

$$a_k = \frac{1}{N} \sum_{n=0}^{N-1} x(n) e^{-j2\pi kn/N}, \quad k = 0, 1, 2, \ldots, N-1 \tag{4.3}$$

It is common to associate the factor $(1/N)$ with $x(n)$ rather than a_k. This can be done by denoting Na_k by $X(k)$; in such a case, we have

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{j2\pi kn/N} \tag{4.4}$$

where the Fourier coefficients $X(k)$ are given by

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi kn/N}, \quad k = 0, 1, 2, \ldots, N-1 \tag{4.5}$$

It is easily seen that $X(k+N) = X(k)$ that is, the Fourier coefficient sequence $X(k)$, is also periodic of period N. Hence, the spectrum of a signal $x(n)$ that is periodic with period N is also a periodic sequence with the same period. It is also noted that since the Fourier series of a discrete periodic signal is a finite sequence,

the series always converges and the Fourier series gives an exact alternate representation of the discrete sequence $x(n)$.

4.1.1 Periodic Convolution

In the case of two periodic sequences $x_1(n)$ and $x_2(n)$ having the same period N, linear convolution as defined by Eq. (2.38) does not converge. Hence, we define a different form of convolution for periodic signals by the relation

$$y(n) = \sum_{m=0}^{N-1} x_1(m)x_2(n-m) = \sum_{m=0}^{N-1} x_1(n-m)x_2(m) \tag{4.6}$$

The above convolution is called *periodic convolution.* It may be observed that $y(n) = y(n+N)$, that is, the periodic convolution is itself periodic of period N.

Some important properties of the DTFS are given in Table 4.1. In this table, it is assumed that $x_1(n)$ and $x_2(n)$ are periodic sequences having the same period N. The proofs are omitted here, since they are similar to the ones that will be given in Sect. 4.3 for the corresponding properties of the DFT.

Example 4.1 Obtain the DTFS representation of the periodic sequence shown in Fig. 4.1.

Table 4.1 Some important properties of DTFS

Property	Periodic sequence	DTFS coefficients
Linearity	$ax_1(n) + bx_2(n)$ a and b are constants	$aX_1(k) + bX_2(k)$
Time shifting	$x(n-m)$	$e^{-j\left(\frac{2\pi}{N}\right)km}X(k)$
Frequency shifting	$e^{j\left(\frac{2\pi}{N}\right)ln}x(n)$	$X(k-l)$
Periodic convolution	$\sum_{m=0}^{N-1} x_1(m)x_2(n-m)$	$X_1(k)X_2(k)$
Multiplication	$x_1(n)x_2(n)$	$\frac{1}{N}\sum_{l=0}^{N-1} X_1(l)X_2(k-l)$
Symmetry properties	$x^*(n)$	$X^*(-k)$
	$x^*(-n)$	$X^*(k)$
	Re $\{x(n)\}$ jIm $\{x(n)\}$	$X_e(k) = \frac{1}{2}(X(k) + X^*(-k))$ $X_o(k) = \frac{1}{2j}(X(k) - X^*(k))$
	$x_e(n) = \frac{1}{2}[x(n) + x^*(-n)]$ $x_o(n) = \frac{1}{2}[x(n) - x^*(-n)]$	Re$\{X(k)\}$ jIm$\{X(k)\}$
	If $x(n)$ is real $x_e(n) = \frac{1}{2}[x(n) + x(-n)]$ $x_o(n) = \frac{1}{2}[x(n) - x(-n)]$	Re$\{X(k)\}$ jIm$\{X(k)\}$

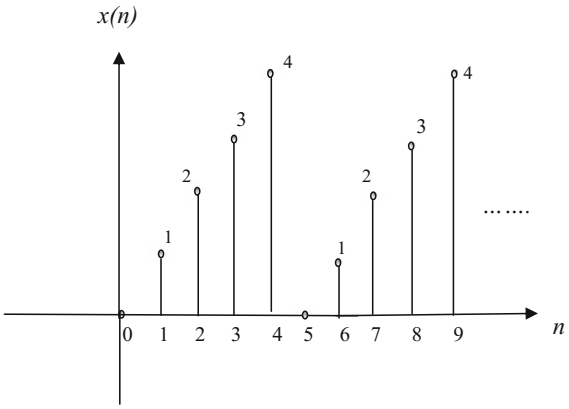


Fig. 4.1 Periodic sequence with period $N = 5$

Solution The sequence is periodic with period $N = 5$. Using Eq. (4.5), the DTFS coefficients are computed as

$$X(0) = \sum_{n=0}^{N-1} x(n)\mathrm{e}^{0} = 0 + 1 + 2 + 3 + 4 = 10$$

$$\begin{aligned} X(1) &= \sum_{n=0}^{4} x(n)\mathrm{e}^{-j2\pi n/5} = 0 + \mathrm{e}^{-j2\pi/5} + 2\mathrm{e}^{-j4\pi/5} + 3\mathrm{e}^{-j6\pi/5} + 4\mathrm{e}^{-j8\pi/5} \\ &= -2.5000 + j3.4410 \end{aligned}$$

$$\begin{aligned} X(2) &= \sum_{n=0}^{4} x(n)\mathrm{e}^{-j4\pi n/5} = 0 + \mathrm{e}^{-j4\pi/5} + 2\mathrm{e}^{-j8\pi/5} + 3\mathrm{e}^{-j12\pi/5} + 4\mathrm{e}^{-j16\pi/5} \\ &= -2.5000 + j0.8123 \end{aligned}$$

$$\begin{aligned} X(3) &= \sum_{n=0}^{4} x(n)\mathrm{e}^{-j6\pi n/5} = 0 + \mathrm{e}^{-j6\pi/5} + 2\mathrm{e}^{-j12\pi/5} + 3\mathrm{e}^{-j18\pi/5} + 4\mathrm{e}^{-j24\pi/5} \\ &= -2.5000 - j0.8123 \end{aligned}$$

$$\begin{aligned} X(4) &= \sum_{n=0}^{4} x(n)\mathrm{e}^{-j8\pi n/5} = 0 + \mathrm{e}^{-j8\pi/5} + 2\mathrm{e}^{-j16\pi/5} + 3\mathrm{e}^{-j24\pi/5} + 4\mathrm{e}^{-j32\pi/5} \\ &= -2.5000 - j3.4410 \end{aligned}$$

Hence, from Eq. (4.4), the DTFS for $x(n)$ is given by

$$\begin{aligned} x(n) = 2 &+ ((-2.5000 + j3.4410)/5)\mathrm{e}^{j-2\pi n/5} + ((-2.5000 + j0.8123))/5\mathrm{e}^{j-4\pi n/5} \\ &+ ((-2.5000 - j0.8123))/5\mathrm{e}^{j-6\pi n/5} + ((-2.5000 - j3.4410))/5\mathrm{e}^{j-8\pi n/5} \end{aligned}$$

Example 4.2 Find the Fourier coefficients in DTFS representation of the sequence $x(n) = \sin\left(\frac{5\pi}{4}\right)n$.

Solution It is clear that the sequence is periodic with period $N = 8$. We may rewrite $x(n)$ in exponential form as

$$x(n) = \frac{1}{2j}e^{\frac{j2\pi 5n}{8}} - \frac{1}{2j}e^{-\frac{j2\pi 5n}{8}} = \frac{1}{2j}e^{\frac{j2\pi 5n}{8}} - \frac{1}{2j}e^{\frac{j2\pi 3n}{8}}$$

Hence, the Fourier coefficients are

$$X(0) = X(1) = X(2) = 0,\ X(3) = -\frac{1}{2j},\ X(4) = \frac{1}{2j},\ X(5) = X(6) = X(7) = 0$$

4.2 The Discrete Fourier Transform

Consider a finite discrete sequence $x(n), 0 \le n \le N-1$. It is known from Eq. (2.69) that the DTFT of the sequence $x(n)$ is given by

$$X(\omega) = \sum_{n=0}^{N-1} x(n)e^{-j\omega n}$$

where $X(\omega)$ is a continuous function of ω in the range $-\pi$ to π or 0–2 π. When $X(\omega)$ is computed at a finite number of values ω_k that are uniformly spaced, we have

$$X(\omega_k) = \sum_{n=0}^{N-1} x(n)e^{-j\omega_k n}, \quad k = 0, 1, 2, \ \ldots, \ M-1$$

where $\omega_k = (2\pi k/M)$. The number of frequency samples may take any value; however, it is chosen as equal N, the length of the discrete sequence $x(n)$. Rewriting $X(\omega_k)$ as $X(k)$, the above equation can be written as

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-j2\pi nk/N}, \quad k = 0, 1, 2, \ \ldots, \ N-1 \tag{4.7}$$

Equation (4.7) is called the discrete Fourier transform of the N-point sequence $x(n)$. One of the main reasons as to why DFT is used to such a great extent is in view of the existence of fast and efficient algorithms for its computation. These algorithms are called fast Fourier transforms (FFTs). Later, in this chapter we consider two of the FFTs.

Given $X(k)$, we now find an expression for $x(n)$ in terms of $X(k)$. For this purpose, we multiply both sides of Eq. (4.7) by $e^{j2\pi lk/N}$ to get

$$X(k)e^{j2\pi lk/N} = \sum_{n=0}^{N-1} x(n)e^{j2\pi lk/N}e^{-j2\pi nk/N}$$

Hence,

$$\sum_{k=0}^{N-1} X(k)e^{j2\pi lk/N} = \sum_{n=0}^{N-1}\sum_{k=0}^{N-1} x(n)e^{j2\pi lk/N}e^{-j2\pi nk/N} \tag{4.8}$$

Using Eq. (4.2), we have

$$\sum_{k=0}^{N-1} x(n)e^{j2\pi lk/N}e^{-j2\pi nk/N} = \begin{cases} 0 & n \neq l \\ N & n = l \end{cases}$$

Substituting the above in Eq. (4.8), we get

$$\sum_{k=0}^{N-1} X(k)e^{j2\pi lk/N} = Nx(l)$$

or

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{j2\pi nk/N}, \quad n = 0, 1, 2, \ldots, N-1 \tag{4.9}$$

The above equation is called the inverse discrete Fourier transform (IDFT). It is seen that $X(k)$ as defined by Eq. (4.7) is periodic with a period N, since $X(k) = X(k+N)$; that is, the IDFT operation results in a periodic sequence of which only the first N values corresponding to one period are evaluated. Also, from Eq. (4.9), we see that $x(n) = x(n+N)$. In other words, we are replacing in effect the finite sequence $x(n)$ by its periodic extension in all the operations that involve DFT and IDFT. In fact, if we now compare Eqs. (4.4) and (4.5) with Eqs. (4.9) and (4.7), we see that the DFT $X(k)$ of a finite sequence of length N can be interpreted as the Fourier coefficient in the DFS expansion of its periodic extension $\tilde{x}(n)$.

If we now define

$$W_N = e^{-j2\pi/N} \tag{4.10}$$

then the DFT and IDFT defined in Eqs. (4.7) and (4.9) can be rewritten as

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk}, \quad k = 0, 1, 2, \ \ldots, \ N-1 \tag{4.11}$$

and

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) W_N^{-nk}, \quad n = 0, 1, 2, \ \ldots, \ N-1 \tag{4.12}$$

For notational convenience, the above DFT and IDFT equations are denoted as

$$X(k) = \text{DFT}\{x(n)\}$$
$$x(n) = \text{IDFT}\{X(k)\}$$

In the DFT expression, W_N^{nk} for $0 \leq n, \ k \leq N-1$, are called the twiddle factors of the DFT. The twiddle factors are periodic and define points on the unit circle in the complex plane. Also, they possess some interesting symmetry properties. Some basic properties of W_N are given below.

1. $W_N^k = W_N^{(k+N)}$
2. $W_N^{N/4} = j$
3. $W_N^{N/2} = -1$
4. $W_N^{3N/4} = j$
5. $W_N^{N/N} = 1$
6. $W_N^{kN} = 1$
7. $W_N^{kN+r} = W_N^r$
8. $W_N^{k+N/2} = -W_N^k$
9. $W_N^{2k} = W_{N/2}^k$
10. $W_N^* = W_N^{-1}$

Example 4.3 Find the twiddle factors for an eight-point DFT.

Solution For $N = 8$, $W_8^k = e^{-j2\pi k/8}$. Hence, the twiddle factors are:

$$W_8^0 = 1, W_8^1 = 0.707 - j0.707, W_8^2 = j, W_8^3 = -0.707 - j0.707$$
$$W_8^4 = -1, W_8^5 = -W_8^1, W_8^6 = -W_8^2, W_8^7 = -W_8^3, \text{ and}$$
$$W_8^{k+N} = W_8^k.$$

Example 4.4 Find the DFT of the sequence $x(n) = \{1, 0, 1, 0\}$.

Solution

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk} \quad k = 0, 1, \ldots, N-1$$

$$= \sum_{n=0}^{3} x(n) W_4^{kn} \quad k = 0, 1, \ldots, 3;$$

$$X(0) = \sum_{n=0}^{3} x(n) = \{1 + 0 + 1 + 0\} = 2;$$

$$X(1) = \sum_{n=0}^{3} x(n) W_4^n = \{1 + 0 - 1 + 0\} = 0;$$

$$X(2) = \sum_{n=0}^{3} x(n) W_4^{2n} = \{1 + 0 + 1 + 0\} = 2;$$

$$X(3) = \sum_{n=0}^{3} x(n) W_4^{3n} = \{1 + 0 - 1 + 0\} = 0;$$

Example 4.5 Determine the eight-point DFT of the sequence $x(n) = \{1, 1, 1, 1, 0, 0, 1, 1\}$.

Solution

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk} \quad k = 0, 1, \ldots, N-1.$$

$$= \sum_{n=0}^{8} x(n) W_8^{kn} \quad k = 0, 1, \ldots, 7.$$

$$X(0) = \sum_{n=0}^{7} x(n) = \{1+1+1+1+0+0+1+1\} = 6;$$

$$X(1) = \sum_{n=0}^{7} x(n)W_8^n = \{1+0.707-j0.707-j-0.707-j0.707 + 0+0+j+0.707+j0.707\} = 1.707-j0.707;$$

$$X(2) = \sum_{n=0}^{7} x(n)W_8^{2n} = \{1-j-1+j+0+0-1+j\} = -1+j;$$

$$X(3) = \sum_{n=0}^{7} x(n)W_8^{3n} = \{1-0.707-j0.707+j+0.707-j0.707 + 0+0-j-0.707+j0.707\} = 0.293-j0.707;$$

$$X(4) = \sum_{n=0}^{7} x(n)W_8^{4n} = \{1-1+1-1+0+0+1-1\} = 0;$$

$$X(5) = \sum_{n=0}^{7} x(n)W_8^{5n} = \{1-0.707+j0.707-j+0.707+j0.707+0 + 0+j-0.707-j0.707\} = 0.293+j0.707;$$

$$X(6) = \sum_{n=0}^{7} x(n)W_8^{6n} = \{1+j-1-j+0+0-1-j\} = -1-j;$$

$$X(7) = \sum_{n=0}^{7} x(n)W_8^{7n} = \{1+0.707+j0.707+j-0.707+j0.707+0 + 0-j+0.707-j0.707\} = 1.707+j0.707;$$

Example 4.6 Find the N-point DFT of the signal $x(n) = b^n$.

Solution

$$X(k) = \sum_{n=0}^{N-1} b^n \mathrm{e}^{-j2\pi nk/N} = \sum_{n=0}^{N-1} \left(b\mathrm{e}^{-j2\pi k/N}\right)^n$$

Hence,

$$X(k) = \frac{1-b^N\mathrm{e}^{-j2\pi k}}{1-b\mathrm{e}^{-j2\pi k/N}}.$$

Example 4.7 A finite duration sequence of length N is given as

$$x(n) = \begin{cases} 1 & 0 \le n \le M-1 \\ 0 & \text{otherwise} \end{cases}$$

Determine the N-point DFT of this sequence.

Solution

$$X(k) = \sum_{n=0}^{M-1} \mathrm{e}^{-j2\pi kn/N}$$
$$= \frac{1-\mathrm{e}^{-j2\pi kM/N}}{1-\mathrm{e}^{-j2\pi k/N}} = \frac{\sin(\pi kM/N)}{\sin(\pi k/N)} \mathrm{e}^{-j2\pi k(M-1)/N}, \quad k = 0, 1, \ldots, N-1$$

Example 4.8 A finite duration sequence $x(n)$ of length eight has the DFT $X(k)$ as shown in Fig. 4.2. A new sequence $y(n)$ of length 16 is defined by

$$y(n) = x\left(\frac{n}{2}\right) \quad \text{for } n \text{ even}$$
$$= 0 \qquad \text{for } n \text{ odd}.$$

Sketch the DFT $Y(k)$ as a function of k.

Solution The 16-point DFT of $y(n)$ is

$$Y(k) = \sum_{n=0}^{15} x(n) W_{16}^{nk}, \quad 0 \le k \le 15$$
$$= \sum_{n=0}^{7} x(n) W_{16}^{2nk}$$

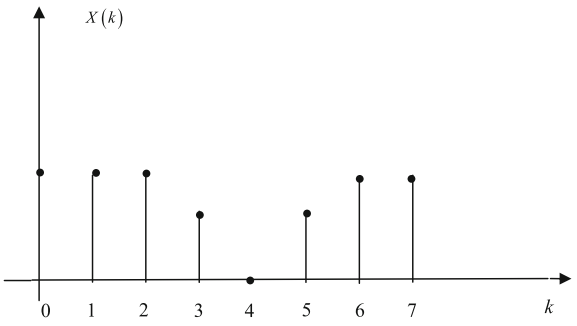


Fig. 4.2 DFT $X(k)$ of $x(n)$ of Example 4.8

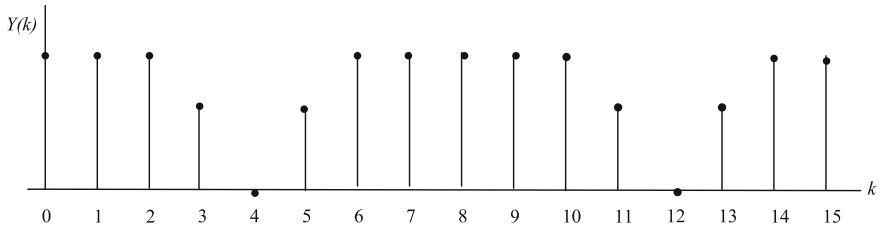


Fig. 4.3 DFT $Y(k)$ of $y(n)$ of Example 4.8

Since $W_N^{2k} = W_{N/2}^k$, the above reduces to

$$Y(k) = \sum_{n=0}^{7} x(n) W_8^{nk}, \quad 0 \le k \le 15$$

Thus, the 16-point DFT $Y(k)$ contains two copies of the eight-point DFT of $x(n)$, and $Y(k)$ has a period of 8. The DFT $Y(k)$ as a function of k is shown in Fig. 4.3.

4.2.1 Circular Operations on a Finite Length Sequence

Circular Shift

Consider a sequence $x(n)$ of length N, $0 \le n \le N-1$. For such a sequence $x(n) = 0$ for $n < 0$ and $n > N-1$. In such a case, if we shift the sequence by an arbitrary integer m, then the shifted sequence is no longer be defined in the range $0 \le n \le N-1$. In order to make sure that the shifted sequence always stays in the range $0 \le n \le N-1$, we define what is known as the *circular shift,* by the relation

$$x_c(n) = x(n-m)_N \tag{4.13a}$$

where

$$(n-m)_N = (n-m) \text{ modulo } N \tag{4.13b}$$

This way, any integer n is related to the modulo N as

$$n = (n)_N + \gamma N \tag{4.14}$$

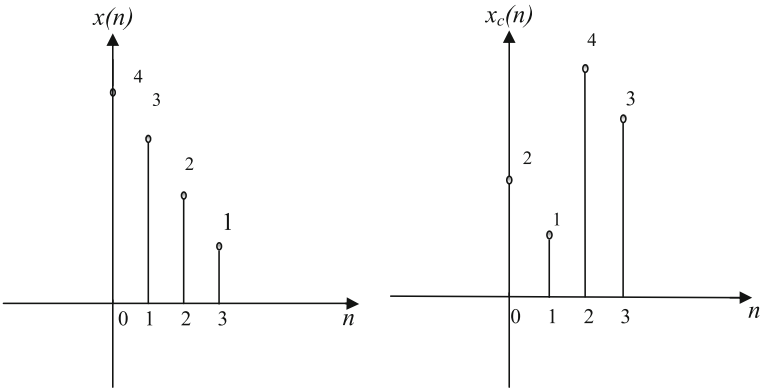


Fig. 4.4 Illustration of circular shift

where γ is an integer and $(n)_N$ is always such that $0 \leq n \leq N-1$. Consequently,

$$x(n-m)_N = \begin{cases} x(n-m) & \text{if } 0 \leq (n-m) \leq N-1 \\ x(\pm N + n - m) & \text{otherwise} \end{cases} \tag{4.15}$$

where $+N$ is used if $m > 0$, and $-N$ is used if $m < 0$.

The circular shift for $m = 2$ is illustrated in Fig. 4.4.

The sequence $x_c(n)$ is related to $x(n)$ by a circular shift of two samples. The samples of $x_c(n)$ can be evaluated using $x_c(n) = x(n-m)_4$. Hence,

$$\boldsymbol{x_c}(0) = \boldsymbol{x}(-2)_4 = \boldsymbol{x}(2);\ \boldsymbol{x_c}(1) = \boldsymbol{x}(-1)_4 = \boldsymbol{x}(3);$$
$$\boldsymbol{x_c}(2) = \boldsymbol{x}(0)_4 = \boldsymbol{x}(0);\ \boldsymbol{x_c}(3) = \boldsymbol{x}(1)_4 = \boldsymbol{x}(1);$$

Circular Time Reversal

For a length-N sequence $x(n)$, $0 \leq n \leq N-1$, the circular time-reversal sequence is also of length-N sequence given by

$$x(-n)_N = x(N-n)_N \tag{4.16}$$

Circular Convolution

Consider two sequences $x(n)$ and $h(n)$, each of length N. Then, the circular convolution of $x(n)$ and $h(n)$ is defined as the length-N sequence $y_c(n)$ given by

$$y_c(n) = \sum_{m=0}^{N-1} x(m)h(n-m)_N \tag{4.17}$$

It is often called as the N-point circular convolution and is denoted by

$$x(n) \textcircled{N} h(n) \tag{4.18}$$

The circular convolution is also commutative like the linear convolution; that is,

$$x(n) \textcircled{N} h(n) = \quad h(n) \textcircled{N} x(n) \tag{4.19}$$

Example 4.9 Find the circular convolution of the three-point sequences $x(n) = \{1, 3, -4\}$ and $h(n) = \{-2, 1, 2\}$.

Solution From Eq. (4.17), $y_c(n) = \sum_{m=0}^{2} x(m)h(n-m)_3$.

Hence,

$$\begin{aligned} y_c(0) &= x(0)h(0) + x(1)h(-1)_3 + x(2)h(-2)_3 \\ &= -2 + 3h(2) - 4h(1) = -2 + 6 - 4 = 0 \\ y_c(1) &= x(0)h(1) + x(1)h(0) + x(2)h(-1)_3 \\ &= 1 + 3h(0) - 4h(2) = 1 - 6 - 8 = -13 \\ y_c(2) &= x(0)h(2) + x(1)h(1) + x(2)h(0) \\ &= 2 + 3 - 8 = -3 \end{aligned}$$

Thus, $y_c(n) = (0, -13, -3)$.

It can also be verified that $\sum_{m=0}^{2} h(m)x(n-m)_3$ leads to the same result, showing that the circular convolution operation is commutative.

Circular Correlation:

Consider two complex-valued sequences $x_1(n)$ and $x_2(n)$, each of length N. Then, the circular correlation of $x_1(n)$ and $x_2(n)$ is defined as the N-point sequence

$$r_{x_1x_2}(m) = \sum_{n=0}^{N-1} x_1(n)x_2^*(n-m)_N \tag{4.20}$$

where $x_2^*(n)$ is the complex conjugate of $x_2(n)$.

4.3 Basic Properties of the Discrete Fourier Transform

In this section, we state and prove some properties of the DFT, which play an important role in digital signal processing applications. We will denote an N-point DFT pair $x(n)$ and $X(k)$ by the following notation

$$\boldsymbol{x(n)} \overset{\mathbf{DFT}}{\underset{N}{\leftrightarrow}} \boldsymbol{X(k)}$$

Linearity:

Consider a sequence $a_1x_1(n)+a_2x_2(n)$ that is a linear combination of $x_1(n)$ and $x_2(n)$, each sequence being of length N, where a_1 and a_2 are arbitrary constants. If the sequences are not of the same length, then the sequence with the lower length is augmented by zeros so that its length is now equal to that of the other sequence. In such a case,

$$\boldsymbol{a_1x_1(n)+a_2x_2(n)} \overset{\mathbf{DFT}}{\underset{N}{\leftrightarrow}} \boldsymbol{a_1X_1(k)+a_2X_2(k)} \tag{4.21}$$

Proof By the definition of the DFT,

$$\begin{aligned}
\mathbf{DFT}(\boldsymbol{a_1x_1(n)+a_2x_2(n)}) &= \sum_{\boldsymbol{n=0}}^{\boldsymbol{N-1}} [\boldsymbol{a_1x_1(n)+a_2x_2(n)}]\boldsymbol{W_N^{kn}} \\
&= \sum_{\boldsymbol{n=0}}^{\boldsymbol{N-1}} [\boldsymbol{a_1x_1(n)}]\boldsymbol{W_N^{kn}} + \sum_{\boldsymbol{n=0}}^{\boldsymbol{N-1}} [\boldsymbol{a_2x_2(n)}]\boldsymbol{W_N^{kn}} \\
&= \boldsymbol{a_1}\sum_{\boldsymbol{n=0}}^{\boldsymbol{N-1}} \boldsymbol{x_1(n)W_N^{kn}} + \boldsymbol{a_2}\sum_{\boldsymbol{n=0}}^{\boldsymbol{N-1}} \boldsymbol{x_2(n)W_N^{kn}} \\
&= \boldsymbol{a_1X_1(k)+a_2X_2(k)}
\end{aligned}$$

Hence, we can write

$$\boldsymbol{a_1x_1(n)+a_2x_2(n)} \overset{\mathbf{DFT}}{\underset{N}{\leftrightarrow}} \boldsymbol{a_1X_1(k)+a_2X_2(k)}$$

Time Reversal of a Sequence:

If $x(n)$ and $X(k)$ are an N-point DFT pair, then

$$\boldsymbol{x(N-n)} \overset{\mathbf{DFT}}{\underset{N}{\leftrightarrow}} \boldsymbol{X(N-k)} \tag{4.22}$$

Proof

$$\mathrm{DFT}\{\boldsymbol{x}(\boldsymbol{N}-\boldsymbol{n})\} = \sum_{\boldsymbol{n}=0}^{\boldsymbol{N}-1} \boldsymbol{x}(\boldsymbol{N}-\boldsymbol{n})\mathrm{e}^{-j2\pi \boldsymbol{k}\boldsymbol{n}/\boldsymbol{N}}$$

Changing the index from n to $m = N - n$ in the RHS of the above equation, we can rewrite it as

$$\begin{aligned}\mathrm{DFT}\{\boldsymbol{x}(\boldsymbol{N}-\boldsymbol{n})\} &= \sum_{\boldsymbol{m}=0}^{\boldsymbol{N}-1} \boldsymbol{x}(\boldsymbol{m})\mathrm{e}^{-j2\pi \boldsymbol{k}(\boldsymbol{N}-\boldsymbol{m})/\boldsymbol{N}} \\ &= \sum_{\boldsymbol{m}=0}^{\boldsymbol{N}-1} \boldsymbol{x}(\boldsymbol{m})\mathrm{e}^{j2\pi \boldsymbol{k}\boldsymbol{m}/\boldsymbol{N}} = \sum_{\boldsymbol{m}=0}^{\boldsymbol{N}-1} \boldsymbol{x}(\boldsymbol{m})\mathrm{e}^{-j2\pi \boldsymbol{m}(\boldsymbol{N}-\boldsymbol{k})/\boldsymbol{N}} = \boldsymbol{X}(\boldsymbol{N}-\boldsymbol{k})\end{aligned}$$

Circular Time Shifting:

The DFT of a circularly time-shifted sequence $x(n-m)_N$ is given by $W_N^{km}X(k)$, that is,

$$\boldsymbol{x}\left[(\boldsymbol{n}-\boldsymbol{m})_{\boldsymbol{N}}\right] \overset{\mathbf{DFT}}{\underset{\boldsymbol{N}}{\leftrightarrow}} \boldsymbol{W}_{\boldsymbol{N}}^{\boldsymbol{km}}\boldsymbol{X}(\boldsymbol{k}) \tag{4.23}$$

Proof By the definition of DFT,

$$\begin{aligned}\mathrm{DFT}\left\{x(n-m)_N\right\} &= \sum_{n=0}^{N-1} x(n-m)_N W_N^{km} \\ &= \sum_{n=0}^{m-1} x(n-m)_N W_N^{km} + \sum_{n=m}^{N-1} x(n-m)W_N^{km}\end{aligned}$$

Since $x(n-m)_N = x(N-m+n)$, we can write the above equation as

$$\begin{aligned}\mathrm{DFT}\left\{x(n-m)_N\right\} &= \sum_{n=0}^{m-1} x(N-m+n)\mathrm{e}^{-j2\pi kn/N} + \sum_{l=0}^{N-1-m} x(l)\mathrm{e}^{-j2\pi k(l+m)/N} \\ &= \sum_{l=N-m}^{N-1} x(l)\mathrm{e}^{-j2\pi k(l+m+N)/N} + \sum_{l=0}^{N-1-m} x(l)\mathrm{e}^{-j2\pi k(l+m)/N} \\ &= \sum_{l=N-m}^{N-1} x(l)\mathrm{e}^{-j2\pi k(l+m)/N} + \sum_{l=0}^{N-1-m} x(l)\mathrm{e}^{-j2\pi k(l+m)/N} \\ &= \sum_{l=0}^{N-1} x(l)\mathrm{e}^{-j2\pi k(l+m)/N} = \mathrm{e}^{-j2\pi km/N}\sum_{l=0}^{N-1} x(l)\mathrm{e}^{-j2\pi kl/N} \\ &= W_N^{km}X(k)\end{aligned}$$

Circular Frequency Shifting:

If $x(n)$ and $X(k)$ are an N-point DFT pair, then

$$W_N^{-mn}x(n) \overset{\text{DFT}}{\underset{N}{\leftrightarrow}} X\left[(k-m)_N\right] \tag{4.24}$$

where $X\left[(k-m)_N\right]$ is a circularly frequency-shifted version of $X(k)$.

Proof

$$\begin{aligned}\text{DFT}\{W_N^{-mn}x(n)\} &= \sum_{n=0}^{N-1} W_N^{-mn}x(n)W_N^{kn} \\ &= \sum_{n=0}^{N-1} x(n)W_N^{n(k-m)} = \sum_{n=0}^{N-1} x(n)W_N^{n(N+k-m)}\end{aligned}$$

Circular Convolution:

The DFT of the circular convolution of two length-N sequences is the product of their N-point DFTs, i.e.,

$$x_1(n) \textcircled{N} x_2(n) \quad \overset{\text{DFT}}{\underset{N}{\leftrightarrow}} X_1(k)X_2(k) \tag{4.25}$$

Proof Let $y_c(n)$ represent the circular convolution of the sequences $x_1(n)$ and $x_2(n)$, i.e.,

$$y_c(n) = \sum_{l=0}^{N-1} x_1(l)x_2(n-l)_N$$

Then, the DFT of $y_c(n)$ is

$$Y_c(k) = \sum_{n=0}^{N-1} y_c(n)W_N^{kn} = \sum_{n=0}^{N-1}\left[\sum_{l=0}^{N-1} x_1(l)x_2(n-l)_N\right]W_N^{kn}$$

By interchanging the order of the summation, we obtain

$$Y_c(k) = \sum_{l=0}^{N-1} x_1(l)\left[\sum_{n=0}^{N-1} x_2(n-l)_N\right]W_N^{kn}$$

Substituting $(n-l)=m$, where m is integer with $0\le m\le N-1$, we get

$$\begin{aligned} Y_c(k) &= \sum_{l=0}^{N-1} x_1(l)\left[\sum_{m=0}^{N-1} x_2(m)\right] W_N^{k(l+m)} = \sum_{l=0}^{N-1} x_1(l)\left[\sum_{m=0}^{N-1} x_2(m) W_N^{km}\right] W_N^{kl} \\ &= \sum_{l=0}^{N-1} x_1(l)[X_2(k)] W_N^{kl} = \left[\sum_{l=0}^{N-1} x_1(l) W_N^{kl}\right][X_2(k)] \\ &= X_1(k)X_2(k) \end{aligned}$$

Circular Correlation:

The DFT of the circular correlation of two complex-valued N-point sequences $x_1(n)$ and $x_2(n)$ is given by $X_1(k)X_2^*(k)$, i.e.,

$$\boldsymbol{r_{x_1x_2}(m)} = \sum_{\boldsymbol{n}=0}^{N-1} \boldsymbol{x_1(n)x_2^*[(n-m)]_N} \overset{\underset{N}{\mathbf{DFT}}}{\leftrightarrow} \boldsymbol{X_1(k)X_2^*(k)} \tag{4.26}$$

Proof From Eq. (4.20), we know that

$$r_{x_1x_2}(m) = \sum_{n=0}^{N-1} x_1(n)x_2^*(n-m)_N = \sum_{n=0}^{N-1} x_1(n)x_2^*(-(m-n)_N) \tag{4.27a}$$

Also, the circular convolution of two sequences $x_1(m)$ and $x_2(m)$ is given by

$$y_c(m) = \sum_{l=0}^{N-1} x_1(l)x_2(m-l)_N = \sum_{n=0}^{N-1} x_1(n)x_2(m-n)_N \tag{4.27b}$$

Comparing Eqs. (4.27a) and (4.27b), we see that $r_{x_1x_2}(m)$ can be considered as the circular convolution of $x_1(m)$ and $x_2^*(-m)_N$. Hence, $\mathrm{DFT}[r_{x_1x_2}(m)] = [\mathrm{DFT}\{x_1(m)\}]\left[\mathrm{DFT}\left\{x_2^*(-m)_N\right\}\right]$. It can be shown that (see Eq. (4.41)) $\mathrm{DFT}\{x_2^*(-m)_N\} = X_2^*(k)$. Thus,

$$\mathrm{DFT}[r_{x_1x_2}(m)] = R_{x_1x_2}(k) = X_1(k)X_2^*(k) \tag{4.28}$$

If $x_1(n) = x_2(n) = x(n)$, then

$$R_{x_1x_2}(k) = |X(k)|^2 \tag{4.29}$$

Parseval's Theorem:

If $x_1(n)$ and $x_2(n)$ are two complex-valued N-point sequences with DFTs $X_1(k)$ and $X_2(k)$, then

$$\sum_{n=0}^{N-1} x_1(n)x_2^*(n) = \frac{1}{N}\sum_{k=0}^{N-1} X_1(k)X_2^*(k) \tag{4.30}$$

Proof From Eq. (4.28), we have $R_{x_1x_2}(k) = X_1(k)X_2^*(k)$. Hence,

$$r_{x_1x_2}(m) = \frac{1}{N}\sum_{k=0}^{N-1} X_1(k)X_2^*(k)W_N^{-km}$$

Evaluating the above at $m = 0$ gives

$$r_{x_1x_2}(0) = \frac{1}{N}\sum_{k=0}^{N-1} X_1(k)X_2^*(k)$$

Hence,

$$\sum_{n=0}^{N-1} x_1(n)x_2^*(n) = \frac{1}{N}\sum_{k=0}^{N-1} X_1(k)X_2^*(k)$$

If $x_1(n) = x_2(n) = x(n)$, then we have

$$\sum_{n=0}^{N-1} |x(n)|^2 = \frac{1}{N}\sum_{k=0}^{N-1} |X(k)|^2 \tag{4.31}$$

The above expression gives a relationship between the energy in a finite duration sequence to the power in the frequency components.

Multiplication of two Sequences:

The DFT of the product of two sequences $x_1(n)$ and $x_2(n)$, each of length N, is given by the circular convolution of their DFTs $X_1(k)$ and $X_2(k)$ divided by N, i.e.,

$$x_1(n)x_2(n) \overset{\text{DFT}}{\underset{N}{\longleftrightarrow}} \frac{1}{N} X_1(k) \textcircled{N} X_2(k) \tag{4.32}$$

This property is dual of the circular convolution property and is left as an exercise for the student.

The above properties are summarized in Table 4.2.

Table 4.2 Basic properties of the discrete Fourier transform

Property	Sequence	DFT
Linearity	$a_1x_1(n)+a_2x_2(n)$	$\boldsymbol{a_1X_1(k)+a_2X_2(k)}$
Periodicity	$\boldsymbol{x(n+N)=x(n)}$	$\boldsymbol{X(k+N)=X(k)}$.
Time reversal	$\boldsymbol{x(N-n)}$	$\boldsymbol{X(N-k)}$
Circular time shifting	$\boldsymbol{x\left[(n-m)_N\right]}$	$\boldsymbol{W_N^{km}X(k)}$
Circular frequency shifting	$\boldsymbol{W_N^{-mn}x(n)}$	$\boldsymbol{X\left[(k-m)_N\right]}$
N-point circular convolution	$x_1(n)$ (N) $x_2(n)$	$\boldsymbol{X_1(k)X_2(k)}$
Circular correlation	$x_1(n)$ (N) $x_2^*(-n)$	$\boldsymbol{X_1(k)X_2^*(k)}$
Multiplication of two sequences	$\boldsymbol{x_1(n)x_2(n)}$	$\frac{1}{N}X_1(k)$ (N) $X_2(k)$
Parseval's theorem $\sum_{n=0}^{N-1}\lvert\boldsymbol{x(n)}\rvert^2=\frac{1}{N}\sum_{k=0}^{N-1}\lvert\boldsymbol{X(k)}\rvert^2$		

4.4 Symmetry Relations of DFT

4.4.1 Symmetry Relations of DFT of Complex-Valued Sequences

Consider a complex-valued sequence $x(n)$, which is expressed as

$$x(n)=x_R(n)+jx_I(n),\qquad 0\le n\le N-1 \tag{4.33}$$

The DFT of $x(n)$ is given by

$$\begin{aligned}X(k)&=\sum_{n=0}^{N-1}x(n)W_N^{kn}=\sum_{n=0}^{N-1}\left[x_R(n)+jx_I(n)\right]\left[\cos\frac{2\pi kn}{N}-j\sin\frac{2\pi kn}{N}\right]\\&=\sum_{n=0}^{N-1}\left[x_R(n)\cos\frac{2\pi kn}{N}+x_I(n)\sin\frac{2\pi kn}{N}\right]-j\sum_{n=0}^{N-1}\left[x_R(n)\sin\frac{2\pi kn}{N}+x_I(n)\cos\frac{2\pi kn}{N}\right]\end{aligned} \tag{4.34}$$

If

$$X(k)=X_R(k)+jX_I(k) \tag{4.35}$$

then

$$X_R(k)=\sum_{n=0}^{N-1}\left[x_R(n)\cos\frac{2\pi kn}{N}+x_I(n)\sin\frac{2\pi kn}{N}\right] \tag{4.36a}$$

and

$$X_I(k) = -\sum_{n=0}^{N-1}\left[x_R(n)\sin\frac{2\pi kn}{N} + x_I(n)\cos\frac{2\pi kn}{N}\right] \tag{4.36b}$$

Similarly, we can show that

$$x_R(n) = \frac{1}{N}\sum_{k=0}^{N-1}\left[X_R(k)\cos\frac{2\pi kn}{N} - X_I(nk)\sin\frac{2\pi kn}{N}\right] \tag{4.37a}$$

and

$$x_I(n) = \frac{1}{N}\sum_{k=0}^{N-1}\left[X_R(k)\sin\frac{2\pi kn}{N} + X_I(k)\cos\frac{2\pi kn}{N}\right] \tag{4.37b}$$

Let us now consider a length-N complex conjugate sequence $x^*(n)$. Taking the complex conjugate on both sides of Eq. (4.11), we get

$$X^*(k) = \left[\sum_{n=0}^{N-1} x(n)\mathrm{e}^{-j2\pi nk/N}\right]^*$$

which can be rewritten as

$$X^*(k) = \sum_{n=0}^{N-1} x^*(n)\mathrm{e}^{j2\pi nk/N} \tag{4.38}$$

Hence,

$$\begin{aligned} X^*\left((-k)_N\right) = X^*(N-k) &= \sum_{n=0}^{N-1} x^*(n)\mathrm{e}^{j2\pi n(N-k)/N} \\ &= \sum_{n=0}^{N-1} x^*(n)\mathrm{e}^{-j2\pi nk/N} = \mathrm{DFT}\{x^*(n)\} \end{aligned}$$

Therefore,

$$\mathrm{DFT}\{x^*(n)\} = X^*\left((-k)_N\right) \tag{4.39}$$

Now, we find the DFT of $x^*\left((-n)_N\right)$ as follows:

$$\begin{aligned}\text{DFT}\left\{x^*\left((-n)_N\right)\right\} &= \sum_{n=0}^{N-1} x^*((-n)_N)\mathrm{e}^{-j2\pi nk/N} \\ &= \sum_{n=0}^{N-1} x^*(N-n)\mathrm{e}^{-j2\pi nk/N}\end{aligned} \tag{4.40a}$$

Replacing n by $(N - n)$ in Eq. (4.38), we have

$$X^*(k) = \sum_{n=0}^{N-1} x^*(N-n)\mathrm{e}^{j2\pi(N-n)k/N} = \sum_{n=0}^{N-1} x^*(N-n)\mathrm{e}^{-j2\pi nk/N} \tag{4.40b}$$

It is seen from Eq. (4.40a) and Eq. (4.40b) that

$$\text{DFT}\left\{x^*\left((-n)_N\right)\right\} = X^*(k) \tag{4.41}$$

Since a complex sequence $x(n)$ can be decomposed into a sum of its real and imaginary parts as

$$x(n) = x_R(n) + jx_I(n) \tag{4.42}$$

where

$$x_R(n) = \frac{1}{2}[x(n) + x^*(n)] \tag{4.43a}$$

and

$$jx_I(n) = \frac{1}{2}[x(n) - x^*(n)] \tag{4.43b}$$

it can be easily shown that the DFTs of the real and imaginary parts of complex sequence are given by

$$\text{DFT}\{x_R(n)\} = \frac{1}{2}\left[X(k) + X^*\left((-k)_N\right)\right] = \frac{1}{2}[X(k) + X^*(N-k)] \tag{4.44a}$$

and

$$\text{DFT}\{jx_I(n)\} = \frac{1}{2}\left[X(k) - X^*\left((-k)_N\right)\right] = \frac{1}{2}[X(k) - X^*(N-k)] \tag{4.44b}$$

A complex sequence $x(n)$ can be represented as the sum of a *circular conjugate symmetric sequence* $x_e(n)$ and a *circular conjugate antisymmetric sequence* $x_o(n)$:

Table 4.3 Symmetry properties of DFT of a complex sequence

Sequence	DFT
$x^*(n)$	$X^*\left((-k)_N\right) = X^*(N-k)$
$x^*\left((-n)_N\right)$	$X^*(k)$
$x_R(n)$	$\frac{1}{2}[X(k) + X^*(N-k)]$
$jx_I(n)$	$\frac{1}{2}[X(k) - X^*(N-k)]$
$x_e(n)$	$X_R(k)$
$x_0(n)$	$jX_I(k)$

$$x(n) = x_e(n) + x_o(n) \tag{4.45}$$

where

$$x_e(n) = \frac{1}{2}\left[x(n) + x^*(-n)_N\right] \tag{4.46a}$$

and

$$x_0(n) = \frac{1}{2}\left[x(n) - x^*(-n)_N\right] \tag{4.46b}$$

Then, the DFTs of $x_e(n)$ and $x_0(n)$ can be easily obtained, using Eq. (4.39), as

$$\text{DFT}\{x_e(n)\} = \frac{1}{2}[X(k) + X^*(k)] = X_R(k) \tag{4.47a}$$

and

$$\text{DFT}\{x_0(n)\} = \frac{1}{2}[X(k) - X^*(k)] = jX_I(k) \tag{4.47b}$$

The symmetry properties of the DFT of a complex sequence are summarized in Table 4.3.

4.4.2 *Symmetry Relations of DFT of Real-Valued Sequences*

For a real-valued sequence $x(n)$, $x_I(n) = 0$. Hence, from Eq. (4.34), we get

$$X(k) = \sum_{n=0}^{N-1}\left[x(n)\cos\frac{2\pi kn}{N} - jx(n)\sin\frac{2\pi kn}{N}\right]$$

From symmetry,

$$X\left((-k)_N\right) = X(n-k) = \sum_{n=0}^{N-1}\left[x(n)\cos\frac{2\pi(n-k)n}{N} - jx(n)\sin\frac{2\pi(n-k)n}{N}\right]$$

$$= \sum_{n=0}^{N-1}\left[x(n)\cos\frac{2\pi kn}{N} + jx(n)\sin\frac{2\pi kn}{N}\right] = X^*(k)$$

Hence, we have the symmetry relation

$$X(n-k) = X\left((-k)_N\right) = X^*(k) \tag{4.48}$$

Also, from Eqs. (4.36a), we have

$$X_R(k) = \sum_{n=0}^{N-1}\left[x(n)\cos\frac{2\pi kn}{N}\right]$$

Hence,

$$X_R\left((-k)_N\right) = X_R(N-k) = \sum_{n=0}^{N-1}\left[x(n)\cos\frac{2\pi(N-k)n}{N}\right] = X_R(k)$$

Thus,

$$X_R(k) = X_R\left((-k)_N\right) = X_R(N-k) \tag{4.49a}$$

Similarly, starting with Eqs. (4.36b), we can show that

$$X_I(k) = -X_I\left((-k)_N\right) = -X_I(N-k) \tag{4.49b}$$

From the above relations, we see that the magnitude of $X(k)$ and $X\left((-k)_N\right)$ is equal and that the phase angle of $X(k)$ is negative of that of the phase angle of $X\left((-k)_N\right)$, i.e.,

$$|\boldsymbol{X}(\boldsymbol{k})| = \left|\boldsymbol{X}((-\boldsymbol{k})_N)\right| \tag{4.50a}$$

and

$$\angle\boldsymbol{X}(\boldsymbol{k}) = -\angle\boldsymbol{X}((-\boldsymbol{k})_N) \tag{4.50b}$$

If $x(n)$ is real and even, that is,

$$x(n) = x(N-n) \qquad 0 \le n \le N-1 \tag{4.51}$$

then, from Eq. (4.36a) and Eq. (4.36b), we see that $X_I(k) = 0$ and that the N-point DFT reduces to

$$X(k) = \sum_{n=0}^{N-1} \left[x(n) \cos \frac{2\pi kn}{N} \right] = X_R(k) \qquad 0 \le k \le N-1 \tag{4.52a}$$

Hence, the DFT of a real finite even sequence is itself real and even. Furthermore, the IDFT reduces to

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} \left[X(k) \cos \frac{2\pi kn}{N} \right] \qquad 0 \le n \le N-1 \tag{4.52b}$$

If $x(n)$ is real and odd, that is,

$$x(n) = -x(N-n) \qquad 0 \le n \le N-1 \tag{4.53}$$

then, from Eq. (4.35a) and (4.35b), we see that $X_R(k) = 0$ and that the N-point DFT reduces to

$$X(k) = -j \sum_{n=0}^{N-1} \left[x(n) \sin \frac{2\pi kn}{N} \right] = jX_j(k) \qquad 0 \le k \le N-1 \tag{4.54a}$$

Hence, the DFT of a real finite odd sequence is purely imaginary and odd. Furthermore, the IDFT reduces to

$$x(n) = j \frac{1}{N} \sum_{k=0}^{N-1} \left[X(k) \sin \frac{2\pi kn}{N} \right] \qquad 0 \le n \le N-1 \tag{4.54b}$$

The symmetry relations of DFT of a real-valued sequence are summarized in Table 4.4.

Table 4.4 Symmetry relations of DFT of a real-valued sequence

Sequence	DFT
Real $x(n)$	$X(n-k) = X\left((-k)_N\right) = X^*(k)$
Real $x(n)$	$X_R(k) = X_R((-k)_N) = X_R(N-k)$
Real $x(n)$	$X_I(k) = -X_I((-k)_N) = -X_I(N-k)$
$x(n)$ real and even	$X_R(k)$
$x(n)$ real and odd	$jX_I(k)$
Real $x(n)$	$\lvert X(k)\rvert = \lvert X((-k)_N)\rvert$, $\angle \boldsymbol{X}(\boldsymbol{k}) = -\angle \boldsymbol{X}((-\boldsymbol{k})_N)$

4.4.3 DFTs of Two Real Sequences from a Single N-Point DFT

Equations (4.44a) and (4.44b) can be used to advantage in finding the DFTs of two real sequences of length N. Suppose $x_1(n)$ and $x_2(n)$ are two real N-point sequences with DFTs $X_1(k)$ and $X_2(k)$. Let us define a complex sequence $x(n)$ by

$$x(n) = x_1(n) + jx_2(n) \tag{4.55}$$

Using Eqs. (4.44a) and (4.44b), we may write the DFTs of the two real sequences as

$$X_1(k) = \frac{1}{2}\left[X(k) + X^*\left((-k)_N\right)\right] = \frac{1}{2}\left[X(k) + X^*(N-k)\right] \tag{4.56a}$$

$$X_2(k) = \frac{1}{2j}\left[X(k) - X^*\left((-k)_N\right)\right] = \frac{1}{2j}\left[X(k) - X^*(N-k)\right] \tag{4.56b}$$

Example 4.10 Find the DFTs of the sequences $x_1(n) = (1, 2, 0, 1)$ and $x_2(n) = (1, 0, 1, 0)$ using a single four-point DFT.

Solution

$$x(n) = x_1(n) + jx_2(n) = (1 + j, 2, j, 1)$$

Hence,

$$X(k) = x(0) + x(1)W_4^k + x(2)W_4^{2k} + x(3)W_4^{3k}, \quad k = 0, 1, 2, 3$$

Thus,

$$X(k) = (4 + 2j, 1 - j, -2 + 2j, 1 + j)$$

Hence,

$$X^*(N-k) = (4 - 2j, 1 - j, -2 - 2j, 1 + j)$$

Substituting the values of $X(k)$ and $X^*(N-k)$ in Eqs. (4.56a) and (4.56b), we get

$$X_1(k) = (4, 1 - j, -2, 1 + j) \quad \text{and} \quad X_2(k) = (2, 0, 2, 0)$$

4.5 Computation of Circular Convolution

4.5.1 Circulant Matrix Method

The circular convolution defined by Eq. (4.17) can be written in a matrix form as

$$\begin{bmatrix} y_c(0) \\ y_c(1) \\ y_c(2) \\ \vdots \\ y_c(N-1) \end{bmatrix} = \begin{bmatrix} x(0) & x(N-1) & x(N-2) & \dots & x(1) \\ x(1) & x(0) & x(N-1) & \dots & x(2) \\ x(2) & x(1) & x(0) & \dots & x(3) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x(N-1) & x(N-2) & x(N-3) & \dots & x(0) \end{bmatrix} \begin{bmatrix} h(0) \\ h(1) \\ h(2) \\ \vdots \\ h(N-1) \end{bmatrix} \tag{4.57}$$

The $(N \times N)$ matrix on the RHS of Eq. (4.57) is called the circular convolution matrix or circulant matrix and denoted by C_x. It may be observed that the first column corresponds to the elements of the sequence $x(n)$, and the rest of the columns are derived from the previous ones in a very simple way.

Example 4.11 Find the circular convolution of the sequences considered in Example 4.9, namely $x(n) = (1, 3, -4,)$ and $h(n) = (-2, 1, 2)$.

Solution The circular convolution matrix C_x is given by

$$\begin{bmatrix} 1 & -4 & 3 \\ 3 & 1 & -4 \\ -4 & 3 & 1 \end{bmatrix}$$

Then, the circular convolution of $x(n)$ and $h(n)$ is given by

$$\begin{bmatrix} y_c(0) \\ y_c(1) \\ y_c(2) \end{bmatrix} = \begin{bmatrix} 1 & -4 & 3 \\ 3 & 1 & -4 \\ -4 & 3 & 1 \end{bmatrix} \begin{bmatrix} -2 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 0 \\ -13 \\ 13 \end{bmatrix}$$

Hence, $y_c(n) = (0, -13, 13)$.

4.5.2 Graphical Method

Evaluation of the circular convolution sum at any sample n consists of the following operations:

(i) The sequences $x(n)$ and $h(n)$ are marked on two concentric circles with one sequence on the inner circle in the clockwise direction and the other on the outer circle in a counter clockwise direction as various points, with equal

spacing. For $n = 0$, $y_c(0)$ is obtained by multiplying the two sequences point by point and summing the products.

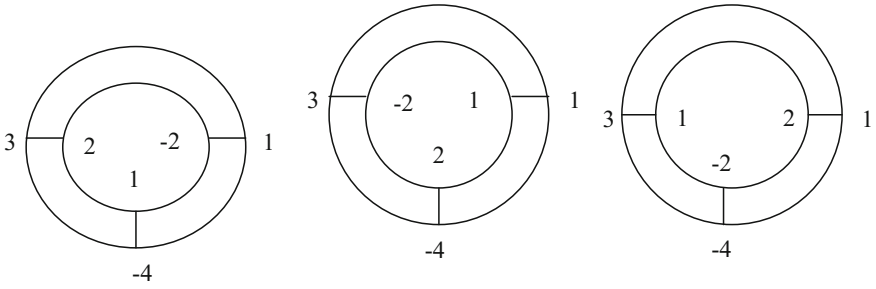
(ii) Keeping the outer circle stationary, rotate the inner in counterclockwise direction by one sample, multiply the two sequences point by point, and sum the products. This gives $y_c(1)$.

(iii) The procedure is continued to find $y_c(n)$ for other values of n.

The following example illustrates the above procedure:

Example 4.12 Find the circular convolution of the three-point sequences of Example 4.11 with $x(n) = (1, 3, -4)$ and $h(n) = (-2, 1, 2)$.

Solution



$$y_c(0) = -2.1 + 2.3 + 1(-4) = 0$$
$$y_c(1) = 1.1 + (-2)3 + 2(-4) = -13$$
$$y_c(2) = 2.1 + 1.3 + (-2)(-4) = 13$$

Hence,

$$y_c(n) = x(n) \textcircled{N} h(n) = (0, -13, 13)$$

4.5.3 DFT Approach

We may obtain the circular convolution $y_c(n)$ of two N-point sequences using the relation given by Eq. (4.25). We first compute the DFTs $X_1(k)$ and $X_2(k)$ of the two sequences and then multiply them to get $Y_c(k) = X_1(k)X_2(k)$, the DFT of the circular convolution. We then perform the IDFT on $Y_c(k)$ to obtain the circular convolution $y_c(n)$. In the next section, we will see how this approach can be used to evaluate linear convolution of two sequences.

Example 4.13 Obtain the circular convolution of the sequences $x_1(n) = (1, 2, 0, 1)$ and $x_2(n) = (1, 0, 1, 0)$ using the DFT approach.

Solution We have already found the DFTs for these two sequences in Example 4.10. These are given by

$$X_1(k) = (4, 1-j, -2, 1+j) \quad \text{and} \quad X_2(k) = (2, 0, 2, 0).$$

Hence,

$$Y_c(k) = (8, 0, -4, 0).$$

Using Eq. (4.12), we now compute the IDFT of the above to obtain the circular convolution $y_c(n)$.

$$\begin{aligned} y_c(n) &= \frac{1}{N}\left[Y_c(0) + Y_c(2)W_4^{-2n} + Y_c(3)W_4^{-3n}\right] \\ &= \frac{1}{N}\left[8 - 4W_4^{-2n}\right] \end{aligned}$$

which gives

$$y_c(n) = (1, 3, 1, 3)$$

4.6 Linear Convolution Using DFT

Linear convolution is an important operation in signal processing applications since it can be used to obtain the response of a linear filter for arbitrary input, once the impulse response of the filter is known. There are efficient algorithms called fast Fourier transforms, two of which will be discussed in the next section, for practical implementation of an N-point DFT. Hence, it is of importance to find methods to implement the linear convolution using the DFT.

4.6.1 Linear Convolution of Two Finite Length Sequences

Consider two sequences $x(n)$ and $h(n)$ of lengths L_1 and L_2, respectively. The linear convolution of these two sequences is a sequence of length $L_1 + L_2 - 1$. Circular convolution cannot be directly used on these two sequences to achieve linear convolution. Now, to obtain linear convolution using circular convolution, we generate two new sequences $x'(n)$ and $h'(n)$, each of length $L_1 + L_2 - 1 = L$ by padding $x(n)$ with $(L_2 - 1)$ zeros and $h(n)$ with $(L_1 - 1)$ zeros. Thus,

$$x'(\mathrm{n}) = [x(0), x(1), \ldots, x(L_1 - 1), \underbrace{0, \ldots, 0}_{L_2 - 1}] \tag{4.58}$$

$$h'(n) = [h(0), h(1), \ldots, h(L_2 - 1), \underbrace{0, \ldots, 0}_{L_1 - 1}] \tag{4.59}$$

The linear convolution of $x'(n)$ and $h'(n)$ is given by

$$x'(n) * h'(n) = \sum_{m=}^{L} x'(m)h'(n-m), \quad 0 \le n \le L-1 \tag{4.60}$$

The above expression can be thought of as a circular convolution of the two padded sequences $x'(n)$ and $h'(n)$; hence, we can use any of the methods described in Sect. 4.5 to evaluate it.

Example 4.14 Find the linear convolution of the sequences $x(n) = (1, 2, 3, 1)$ and $x(n) = (1, 1, 1)$.

Solution The two sequences $x(n)$ and $h(n)$ are of lengths 4 and 3, respectively. By appropriately padding the two sequences by zeros, we obtain the padded sequences $x'(n) = (1, 2, 3, 1, 0, 0)$ and $h'(n) = (1, 1, 1, 0, 0, 0)$, each of length $L = 6$. We may now calculate the circular convolution $y_c(n)$ of $x'(n)$ and $h'(n)$ using the circulant matrix Eq. (4.57)

$$\begin{bmatrix} y_c(0) \\ y_c(1) \\ y_c(2) \\ y_c(3) \\ y_c(4) \\ y_c(5) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 1 & 3 & 2 \\ 2 & 1 & 0 & 0 & 1 & 3 \\ 3 & 2 & 1 & 0 & 0 & 1 \\ 1 & 3 & 2 & 1 & 0 & 0 \\ 0 & 1 & 3 & 2 & 1 & 0 \\ 0 & 0 & 1 & 3 & 2 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \\ 6 \\ 6 \\ 4 \\ 1 \end{bmatrix}$$

Thus, $y_c(n) = (1, 3, 6, 6, 4, 1)$, and therefore, the linear convolution

$$y_l(n) = x(n) * h(n) = (1, 3, 6, 6, 4, 1).$$

Instead of using the circulant matrix, we could have used the DFT approach to find the circular convolution. In this case, we would first find the $L = (L_1 + L_2 - 1)$-point DFTs $X'(k)$ and $H'(k)$ of $x'(n)$ and $h'(n)$. Then, the L-point IDFT of the product $X'(k)H'(k)$ would yield the linear convolution of $x(n)$ and $h(n)$.

The following MATLAB fragments illustrate as to how to obtain the linear convolution using the DFT:

For the above example,

```
x=[1 2 3 1 0 0]; % sequence x(n)
h=[1 1 1 0 0 0];% sequence h(n)
L=length(x)+length(h)-1;%length of convolution sequence
XE=fft(x,L); % DFT of sequence x(n) with zero padding
HE=fft(h,L); % DFT of sequence h(n) with zero padding
yl=ifft(XE.*HE); % linear convolution of sequences x(n) and h(n)
```

After execution of the above MATLAB commands, the linear convolution of $x(n)$ and $h(n)$ is given by

$y_l(n) = x(n) * h(n)$={1, 3, 6, 6, 4, 1}.

4.6.2 Linear Convolution of a Finite Length Sequence with a Long Duration Sequence

There are two methods for the evaluation of the linear convolution using the DFT, called the overlap-add and the overlap-save, when one sequence is of finite length and the other is of infinite length or much greater than the length of the finite length sequence.

(a) **Overlap-Add Method**

Let $x(n)$ be a sequence of long duration and $h(n)$ of finite length L_2. Let the sequence $x(n)$ be divided into a set of subsequences, each having a finite length L, and let each subsequence be padded with L_2-1 zeros to make its length equal to $L + L_2-1$. Then, we have

$$\begin{aligned}
x_1(n) &= [x(0), x(1), \ldots, x(L-1), \underbrace{0, \ldots, 0}_{L_2-1}] \\
x_2(n) &= [x(L), x(L+1), \ldots, x(2L-1), \underbrace{0, \ldots, 0}_{L_2-1}] \\
x_3(n) &= [x(2L), x(2L+1), \ldots, x(3L-1), \underbrace{0, \ldots, 0}_{L_2-1}] \\
&\cdot \\
&\cdot \\
x_m(n) &= [(x((m-1)L), x((m-1)L+1), \ldots, x(mL-1), \underbrace{0, \ldots, 0}_{L_2-1}]
\end{aligned} \tag{4.61}$$

Also, the sequence $h(n)$ is padded with $L - 1$ zeros to form the sequence $h'(n)$. Each of the subsequences is now convolved with $h'(n)$ of length $L + L_2-1$. Since each subsequence is terminated with $L_2 - 1$ zeros, the last $L_2 - 1$ points from each subsequence convolution output are to be overlapped and added to the first $L_2 - 1$ points of the succeeding subsequence convolution output. Hence, this procedure is called the overlap-add method. The following example illustrates this method.

Example 4.15 If the impulse response of a filter is $h(n) = \{1, 0, 1\}$, find its output $y(n) = x(n) * h(n)$ for the input sequence $x(n) = \{3, -1, 0, 1, 2, 1, 0, 1, 2\}$, by using overlap-add method.

Solution Let each subblock of the data be of length 3. Since $L_2 = 3$, two zeros are added to bring the length of each subblock to 5. Two zeros are added to $h(n)$ so that $h'(n)$ is also of length 5. Hence, the sub sequences are

$$x_1(n) = \{3, -1, 0, 0, 0\}; \quad x_2(n) = \{1, 2, 1, 0, 0\}; \quad x_3(n) = \{0, 1, 2, 0, 0\}.$$

and

$$h'(n) = \{1, 0, 1, 0, 0\}$$

Then, the circular convolutions of the subsequences with $h'(n)$ are given by

$$y_1(n) = x_1(n) \textcircled{N} h'(n) = \{3, -1, 3, -1, 0\}$$

$$y_2(n) = x_2(n) \textcircled{N} h'(n) = \{1, 2, 2, 2, 1\}$$

$$y_3(n) = x_3(n) \textcircled{N} h'(n) = \{0, 1, 2, 1, 2\}$$

Hence, the linear convolution of $x(n)$ and $h(n)$ is given by

$$y_l(n) = x(n) * h(n) = (3, -1, 3, 0, 2, 2, 2, 2, 2, 1, 2)$$

The above process is illustrated in Fig. 4.5.

The above procedure can be implemented by using the MATLAB command fftfilt.

```
h = [1 0 1 0 0];

x = [3  − 1 0 1 2 1 0 1 2 0 0];

y = fftfilt(h, x);
```

Thus after the execution of the above MATLAB statements, we get $y_l(n)$ as

$$y_l(n) = \{3, -1, 3, 0, 2, 2, 2, 2, 2, 1, 2\}.$$

(b) **Overlap-Save Method**

In this method, the sequence $x(n)$ is divided into a set of overlapping subsequences, each having a finite length $L + L_2 - 1$. Each subsequence contains the last $L_2 - 1$ samples of the previous subsequence, followed by the next L samples of $x(n)$. The first $L_2 - 1$ samples of the first subsequence are set to zero. Hence, the subsequences are

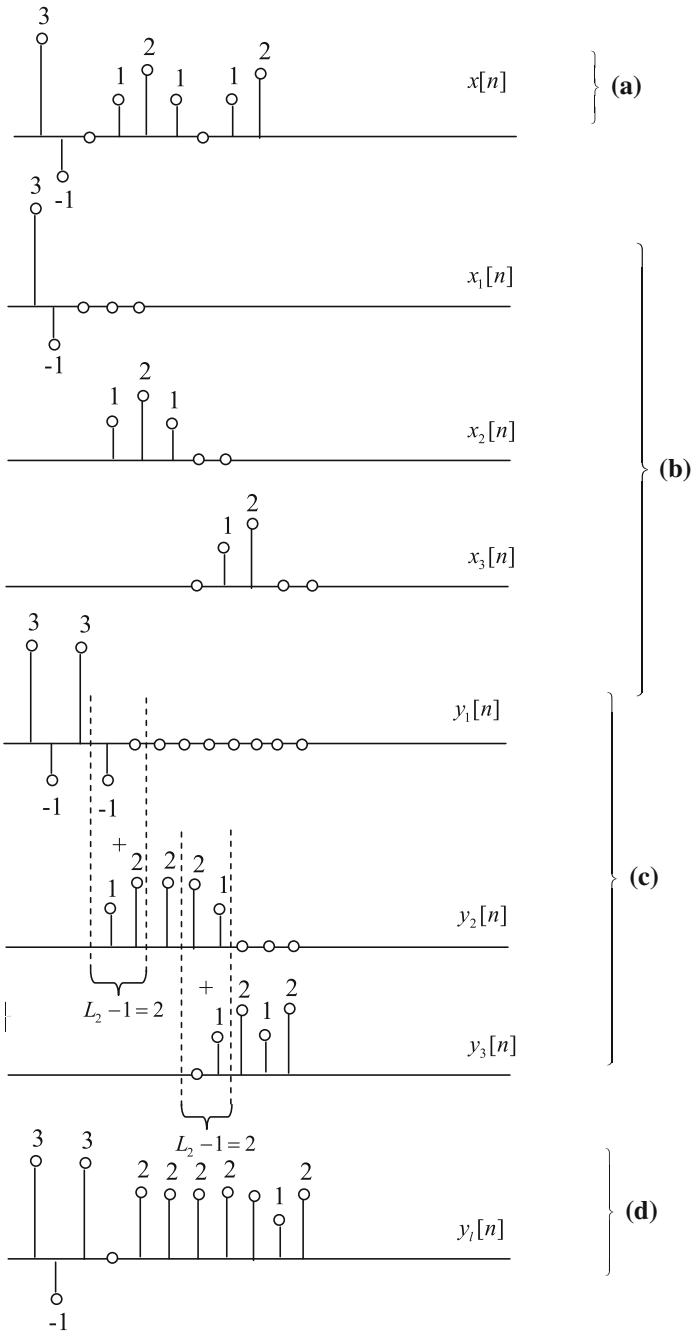


Fig. 4.5 **a** Original signal $x(n)$, **b** subblocks of $x(n)$, **c** circular convolution of the subblocks of $x(n)$ and $h'(n)$, and **d** linear convolution of $x(n)$ and $h(n)$

$$x_1(n) = [\underbrace{0,\ldots,0}_{L_2-1}, x(0), x(1), \ldots, x(L-1)]$$

$$x_2(n) = [\underbrace{x(L+1-L_2),\ldots,x(L-1)}_{L_2-1 \text{ samples from } x_1(n)}, \underbrace{x(L+1),\ldots,x(2L-1)}_{L \text{ new samples}}]$$

$$x_3(n) = [\underbrace{x(2L+1-L_2),\ldots,x(2L-1)}_{L_2-1 \text{ samples from } x_2(n)}, \underbrace{x(2L),\ldots,x(3L-1)}_{L \text{ new samples}}]$$

and so on. Now, the length of the sequence $h(n)$ is increased to $L + L_2 - 1$ by padding it with $L - 1$ zeros to form the sequence $h'(n)$. Then, each of the subsequences is convolved with $h'(n)$. The first $L_2 - 1$ points of the circular convolution of each of the subsequences with $h'(n)$ do not agree with the linear convolution output of each subsequence with $h'(n)$ due to aliasing, and the remaining L points are in agreement with the linear convolution output. Hence, the first $L_2 - 1$ points of the circular convolution of each subsequence with $h'(n)$ output are to be discarded and the remaining L points from each subsequence convolution output are to be abutted to obtain the linear convolution output of $x(n)$ and $h(n)$. The following example illustrates this method:

Example 4.16 Find the filter output $y(n) = x(n) * h(n)$ for the input $x(n)$ and the impulse response $h(n)$ of Example 4.15.

Solution The subsequences of $x(n)$ are

$$x_1(n) = \{0, 0, 3, -1, 0\},\ x_2(n) = \{-1, 0, 1, 2, 1\},$$
$$x_3(n) = \{2, 1, 0, 1, 2\},\ x_4(n) = \{1, 2, 0, 0, 0\}$$

and

$$h'(n) = \{1, 0, 1, 0, 0\}$$

Then, the circular convolution of the subsequences with *h(n)* is given by

$$y_1(n) = x_1(n) \textcircled{N} h'(n) = \{0, 0, 3, -1, 3\}$$

$$y_2(n) = x_2(n) \textcircled{N} h'(n) = \{-1, 0, 0, 2, 2\}$$

$$y_3(n) = x_3(n) \textcircled{N} h'(n) = \{2, 1, 2, 2, 2\}$$

$$y_4(n) = x_4(n) \textcircled{N} h'(n) = \{1, 2, 1, 2, 0\}$$

Hence, the linear convolution of $x(n)$ and $h(n)$ is given by

$$y_l(n) = \{3, -1, 3, 0, 2, 2, 2, 2, 2, 1, 2\}$$

This process is illustrated in Fig. 4.6a, b.

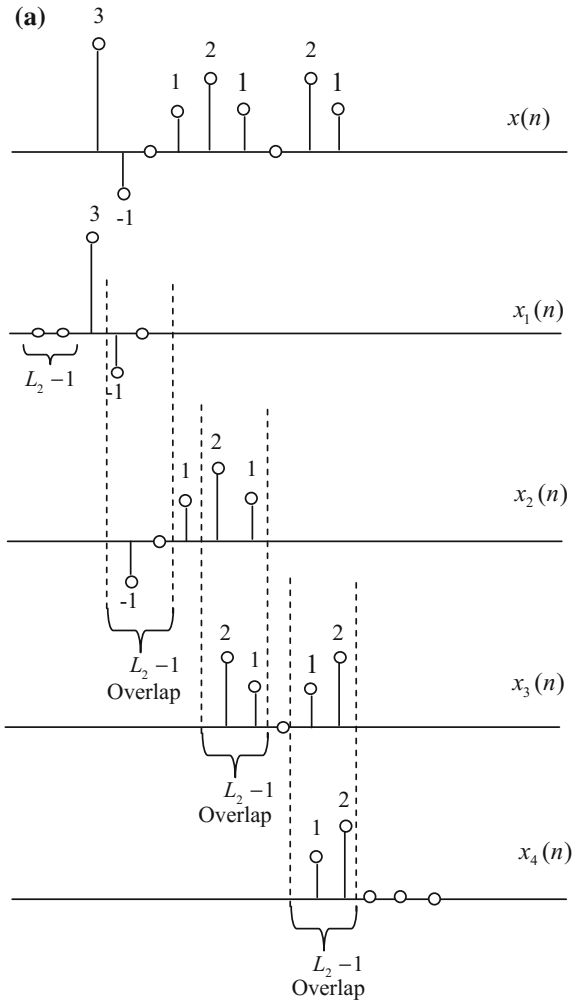


Fig. 4.6 **a** Original input $x(n)$ and subsections of $x(n)$ and **b** circular convolution of subsections of $x(n)$ and $h'(n)$, and the linear convolution of $x(n)$ and $h(n)$

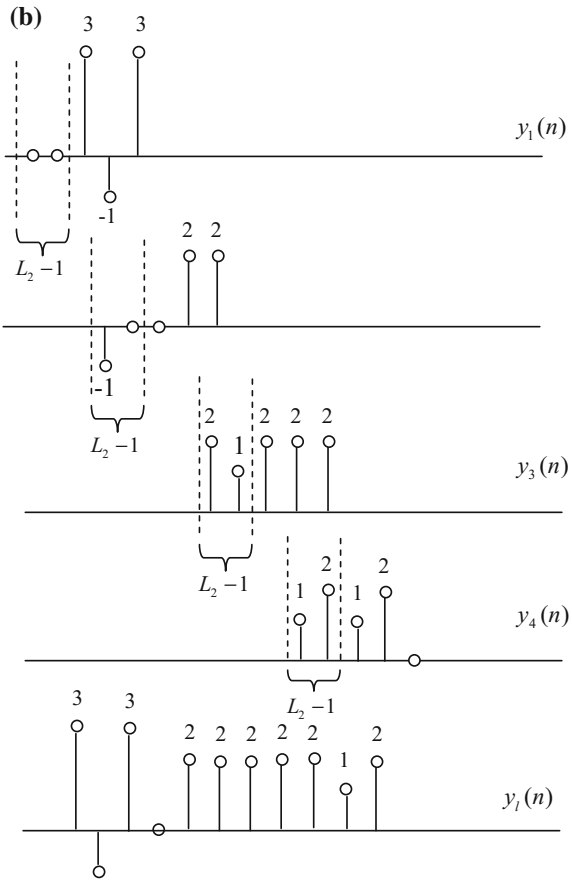


Fig. 4.6 (continued)

4.7 Fast Fourier Transform

It is evident from Eqs. (4.11) that a direct evaluation of each value of $X(k)$ requires N complex multiplications and $(N-1)$ complex additions. As such, N^2 complex multiplications and $N(N-1)$ complex additions are necessary for the computation of an N-point DFT. Consequently, for large N, the computational complexity in terms of the arithmetic operations is high in direct evaluation of the DFT. Therefore, a number of efficient algorithms have been developed for the computation of the DFT. These efficient algorithms collectively have become known *fast Fourier transforms*. The FFT algorithms decompose successively the computation of the discrete Fourier transform of a sequence of length N into smaller and smaller discrete Fourier transforms. The two most basic FFT algorithms are the

decimation-in-time and decimation-in frequency [1, 2], and these are considered in the following sections.

4.7.1 Decimation-in-Time FFT Algorithm with Radix-2

The *decimation-in-time* (DIT) is the process that decomposes the input sequence successively into smaller and smaller subsequences. Here, the radix-2 means the number of output points N can be expressed as a power of 2; that is, $N = 2^v$, where v is an integer. Let the input sequence be decomposed into an even sequence $g_1(n)$ and an odd sequence $g_2(n)$ as

$$g_1(n) = x(2n),\ n = 0, 1, \ldots, \frac{N}{2} - 1 \tag{4.62}$$

$$g_2(n) = x(2n),\ n = 0, 1, \ldots, \frac{N}{2} - 1 \tag{4.63}$$

We know from Eq. (4.11) that

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk}, \quad k = 0, 1, \ldots, N-1 \tag{4.64}$$

Substituting Eqs. (4.62) and (4.63) in (4.64), we get

$$X(k) = \sum_{n=0}^{(N/2)-1} x(2n) W_N^{2nk} + \sum_{n=0}^{(N/2)-1} x(2n+1) W_N^{(2n+1)k} \tag{4.65}$$

Using $W_N^2 = W_{N/2}$ in Eq. (4.65) yields

$$X(k) = \sum_{n=0}^{(N/2)-1} x(2n) W_{N/2}^{nk} + W_N^k \sum_{n=0}^{(N/2)-1} x(2n+1) W_{N/2}^{nk} \tag{4.66}$$

The RHS may be identified as the sum of two (N/2)-point DFTs, $G_1(k)$ and $G_2(k)$ of the even and odd sequences $g_1(n)$ and $g_2(n)$:

$$G_1(k) = \sum_{n=0}^{(N/2)-1} g_1(n) W_{N/2}^{nk} \tag{4.67}$$

$$G_2(k) = \sum_{n=0}^{(N/2)-1} g_2(n) W_{N/2}^{nk} \tag{4.68}$$

Hence, $X(k)$ in Eq. (4.66) can be written as

$$X(k) = G_1(k) + W_N^k G_2(k) \quad k = 0, 1, \ldots, N-1 \tag{4.69}$$

Also, since $G_1(k)$ and $G_2(k)$ are periodic with a period of $(N/2)$, $G_1(k+N/2) = G_1(k)$ and $G_2(k+N/2) = G_2(k)$, and the twiddle constant $W_N^{k+N/2} = -W_N^k$. Hence, Eq. (4.69) can be written as

$$X(k) = G_1(k) + W_N^k G_2(k) \qquad k = 0, 1, \ldots, (N/2) - 1 \tag{4.70a}$$

$$X(k+N/2) = G_1(k) - W_N^k G_2(k) \quad k = 0, 1, \ldots, (N/2) - 1 \tag{4.70b}$$

Repeating the process for each of the sequences $g_1(n)$ and $g_2(n)$, $g_1(n)$ yields two $(N/4)$-point sequences

$$\begin{aligned} g_{11}(n) &= g_1(2n) \qquad n = 0, 1, \ldots, (N/4) - 1 \\ g_{12}(n) &= g_1(2n+1) \qquad n = 0, 1, \ldots, (N/4) - 1 \end{aligned} \tag{4.71a}$$

and $g_2(n)$ yields

$$\begin{aligned} g_{21}(n) &= g_2(2n) \qquad n = 0, 1, \ldots, (N/4) - 1 \\ g_{22}(n) &= g_2(2n+1) \qquad n = 0, 1, \ldots, (N/4) - 1 \end{aligned} \tag{4.71b}$$

and their DFTs satisfy

$$\begin{aligned} G_1(k) &= G_{11}(k) + W_{N/2}^k G_{12}(k) \qquad k = 0, 1, \ldots, (N/4) - 1 \\ G_1\left(k + \frac{N}{4}\right) &= G_{11}(k) - W_{N/2}^k G_{12}(k) \qquad k = 0, 1, \ldots, (N/4) - 1 \end{aligned} \tag{4.72a}$$

$$\begin{aligned} G_2(k) &= G_{21}(k) + W_{N/2}^k G_{22}(k) \qquad k = 0, 1, \ldots, (N/4) - 1 \\ G_2\left(k + \frac{N}{4}\right) &= G_{21}(k) - W_{N/2}^k G_{22}(k) \qquad k = 0, 1, \ldots, (N/4) - 1 \end{aligned} \tag{4.72b}$$

This process can be continued until we are left with only two-point transforms. For example, for $N = 4$, Eqs. (4.70a) and (4.70b) become

$$\begin{aligned} X(k) &= G_1(k) + W_N^k G_2(k) \qquad k = 0, 1 \\ X(k+2) &= G_1(k) - W_N^k G_2(k) \quad k = 0, 1 \end{aligned} \tag{4.73}$$

Equation (4.73) can be represented by the flow graph as shown in Fig. 4.7. This is usually referred to as the butterfly diagram for four-point DFT. In the first stage, two 2-point DFTs and, in the second stage, one 4-point DFT are computed.

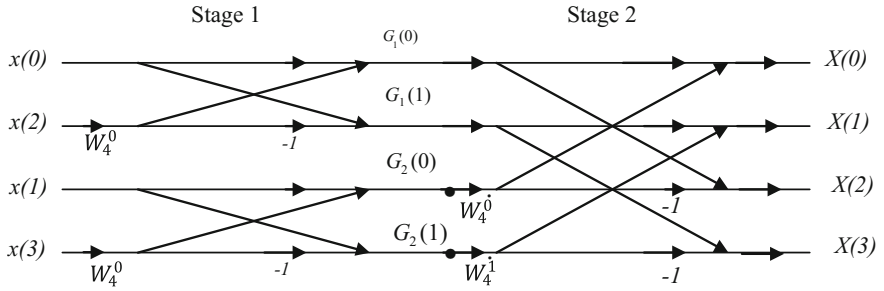


Fig. 4.7 Decomposition of a four-point DFT using DIT

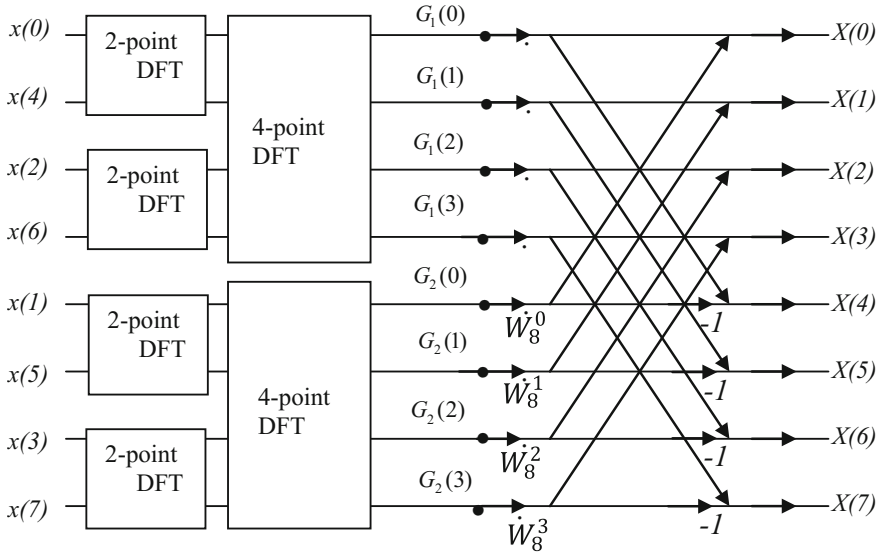


Fig. 4.8 Decomposition of an eight-point DFT using DIT

For $N = 8$, Eqs. (4.70a) and (4.70b) become

$$\begin{aligned} X(k) &= G_1(k) + W_N^k G_2(k) \qquad k = 0, 1, 2, 3 \\ X(k+2) &= G_1(k) - W_N^k G_2(k) \qquad k = 0, 1, 2, 3 \end{aligned} \tag{4.74}$$

The computation of an eight-point DFT is performed in three stages as shown in Fig. 4.8.

It is observed from the flow graph that in the first stage, four 2-point DFTs, in the second stage, two 4-point DFTs, and finally, in the third stage, one 8-point DFT are computed. Also, the number of complex multiplications carried out at each stage is equal to $4 = N/2$, and the number of additions performed is N. Hence, the total

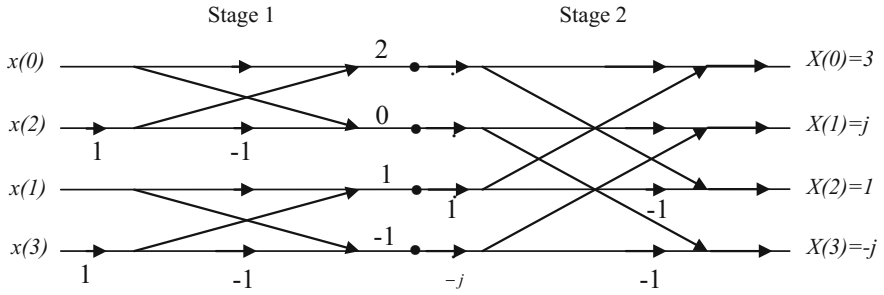


Fig. 4.9 Decomposition of the four-point DFT of Example 4.17 using the DIT algorithm

number of complex multiplications and additions in computing all the 8 samples is 12 and 24, respectively. Following the same argument, it can be observed that in the general case of $N = 2^v$, the number of stages of computation will be $v = \log_2 N$; hence, the total number of complex multiplications and additions needed in computing all the N DFT samples is $(N/2)\log_2 N$, and the number of complex additions is $N \log_2 N$.

Example 4.17 Find the four-point FFT of $x(n) = \{1, 0, 1, 1\}$ using the decimation-in-time algorithm.

Solution With $N = 4$, the two twiddle factors are

$$W_4^0 = 1 \quad \text{and} \quad W_4^1 = e^{-j2\pi/4} = \cos(\pi/2) - j\sin(\pi/2) = -j.$$

Since it is a four-point DFT, the DIT flow graph consists of two stages as shown in Fig. 4.9. The outputs of the first and second stages are computed as follows:

Stage 1

$$\begin{aligned} x_1(0) &= x(0) + W_4^0 x(2) = 1 + 1 = 2; \\ x_1(2) &= x(0) - W_4^0 x(2) = 1 - 1 = 0; \\ x_1(1) &= x(1) + W_4^0 x(3) = 0 + 1 = 1; \\ x_1(3) &= x(1) - W_4^0 x(3) = 0 - 1 = -1; \end{aligned}$$

where the sequence $x_1(n)$ represents the intermediate output after the first stage and becomes the input to the second (final) stage.

Stage 2

$$\begin{aligned} X(0) &= x_1(0) + W_4^0 x_1(1) = 2 + 1 = 3; \\ X(2) &= x_1(0) - W_4^0 x_1(1) = 2 - 1 = 1; \end{aligned}$$

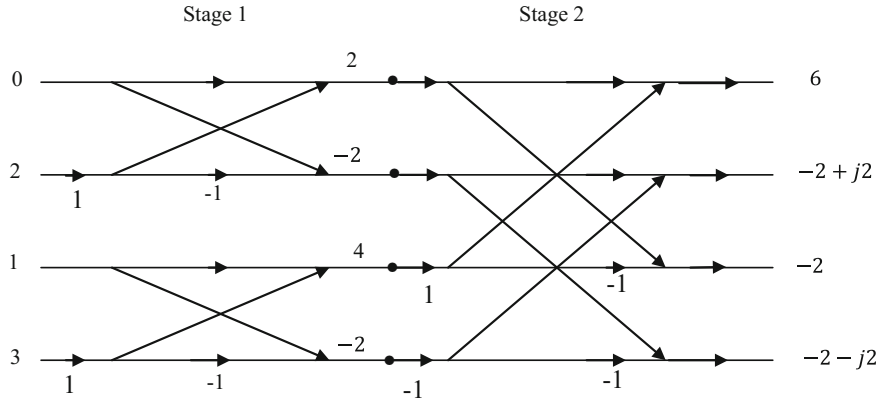


Fig. 4.10 Decomposition of the four-point DFT of Example 4.18 using the DIT algorithm

$$X(1) = x_1(2) + W_4^1 x_1(3) = 1 + (-j)(-1) = j;$$
$$X(3) = x_1(2) - W_4^1 x_1(3) = 0 - (-j)(-1) = -j;$$

Example 4.18 Consider an input data string of $x(n) = (0, 1, 2, 3)$. Draw the butterfly diagram of the FFT showing the input, intermediate outputs, and the final output to compute the DFT of $x(n)$.

Solution By computing the outputs of the first and second stages as was done in the previous example, the required butterfly diagram is shown in Fig. 4.10.

Example 4.19 Find the eight-point FFT of $x(n) = \{1, 0, 1, 1, 1, 1, 1, 0\}$ using the DIT algorithm.

Solution With $N = 8$, the four twiddle factors are

$$W_8^0 = 1;$$
$$W_8^1 = \mathrm{e}^{-j2\pi/8} = \cos(\pi/4) - j\,\sin(\pi/4) = 0.707 - j0.707;$$
$$W_8^2 = \mathrm{e}^{-j4\pi/8} = -j;$$
$$W_8^3 = \mathrm{e}^{-j6\pi/8} = -0.707 - j0.707;$$

Since it is an eight-point DFT with radix-2, the DIT flow graph consists of three stages as shown in Fig. 4.11. The outputs of the three stages are computed as follows:

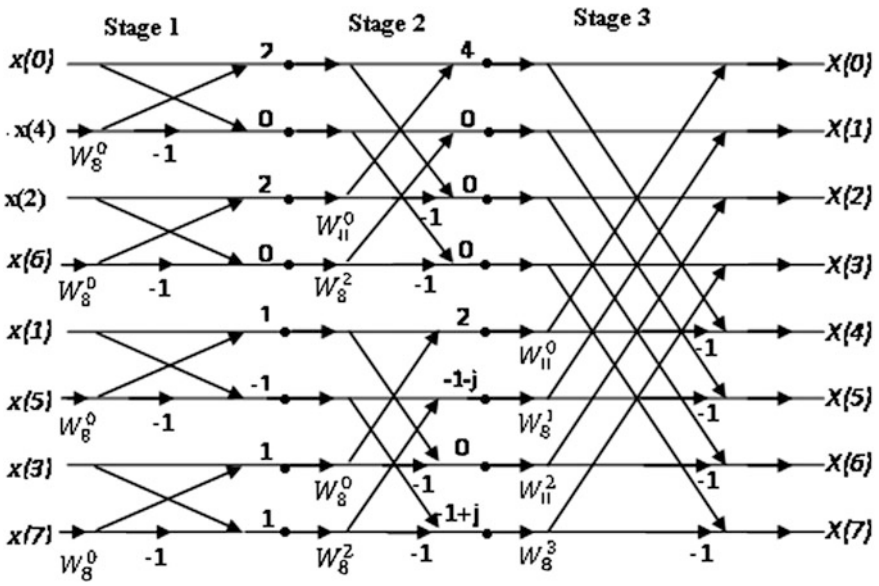


Fig. 4.11 Decomposition of the eight-point DFT of Example 4.19 using the DIT algorithm

Stage 1

$$
\begin{aligned}
x_1(0) &= x(0) + W_8^0 x(4) = 1 + 1 = 2;\\
x_1(4) &= x(0) - W_8^0 x(4) = 1 - 1 = 0;\\
x_1(2) &= x(2) + W_8^0 x(6) = 1 + 1 = 2;\\
x_1(6) &= x(2) - W_8^0 x(6) = 1 - 1 = 0;\\
x_1(1) &= x(1) + W_8^0 x(5) = 0 + 1 = 1;\\
x_1(5) &= x(1) - W_8^0 x(5) = 0 - 1 = -1;\\
x_1(3) &= x(3) + W_8^0 x(7) = 1 + 0 = 1;\\
x_1(7) &= x(3) - W_8^0 x(7) = 1 - 0 = 1;
\end{aligned}
$$

where the sequence $x_1(n)$ represents the intermediate output after the first stage and becomes the input to the second stage.

Stage 2

$$
\begin{aligned}
x_2(0) &= x_1(0) + W_8^0 x_1(2) = 2 + 2 = 4;\\
x_2(4) &= x_1(4) + W_8^2 x_1(6) = 0 + (-j)0 = 0;\\
x_2(2) &= x_1(0) - W_8^0 x_1(2) = 2 - 2 = 0;\\
x_2(6) &= x_1(4) - W_8^2 x_1(6) = 0 + (-j)0 = 0;\\
x_2(1) &= x_1(1) + W_8^0 x_1(3) = 1 + 1 = 2;\\
x_2(5) &= x_1(5) + W_8^2 x_1(7) = -1 + (-j) = -1 - j;\\
x_2(3) &= x_1(1) - W_8^0 x_1(3) = 1 - 1 = 0;\\
x_2(7) &= x_1(5) - W_8^2 x_1(7) = -1 - (-j) = -1 + j;
\end{aligned}
$$

where the second-stage output sequence $x_2(n)$ becomes the input sequence to the final stage.

Stage 3

$$
\begin{aligned}
X(0) &= x_2(0) + W_8^0 x_2(1) = 4 + 2 = 6;\\
X(1) &= x_2(4) + W_8^1 x_2(5) = 0 + (0.707 - j0.707)(-1 - j) = -1.414;\\
X(2) &= x_2(2) + W_8^2 x_2(3) = 0 + (-j)0 = 0;\\
X(3) &= x_2(6) + W_8^3 x_2(7) = 0 + (-0.707 - j0.707)(-1 + j) = 1.414;\\
X(4) &= x_2(0) - W_8^0 x_2(1) = 4 - 2 = 2;\\
X(5) &= x_2(4) - W_8^1 x_2(5) = 0 - (0.707 - j0.707)(-1 - j) = 1.414;\\
X(6) &= x_2(2) - W_8^2 x_2(3) = 0 - (-j)(0) = 0;\\
X(7) &= x_2(6) - W_8^3 x_2(7) = 0 - (-0.707 - j0.707)(-1 + j) = -1.414;
\end{aligned}
$$

Example 4.20 Find the 16-point FFT of the sequence $x(n) = \{1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0\}$ using the DIT algorithm.

Solution With $N = 16$, eight twiddle factors need to be calculated; these are

$$W_{16}^{0} = 1; W_{16}^{1} = e^{-j2\pi/16} = 0.9238 - j0.3826;$$
$$W_{16}^{2} = e^{-j4\pi/16} = 0.707 - j0.707;$$
$$W_{16}^{3} = e^{-j6\pi/16} = 0.3826 - j0.9238;$$
$$W_{16}^{4} = e^{-j8\pi/16} = 0 - j;$$
$$W_{16}^{5} = e^{-j10\pi/16} = -0.3826 - j0.9238;$$
$$W_{16}^{6} = e^{-j12\pi/16} = -0.707 - j0.707;$$
$$W_{16}^{7} = e^{-j14\pi/16} = -0.9238 - j0.3826.$$

Since it is a 16-point DFT with radix-2, the DIT flow graph consists of four stages as shown in Fig. 4.12. The outputs of the four stages are computed as follows:

Stage 1

$$x_1(0) = x(0) + W_{16}^{0}x(8) = 1 + 1 = 2;$$
$$x_1(8) = x(0) - W_{16}^{0}x(8) = 1 - 1 = 0;$$
$$x_1(4) = x(4) + W_{16}^{0}x(12) = 0 + 1 = 1;$$
$$x_1(12) = x(4) - W_{16}^{0}x(12) = 0 - 1 = -1;$$
$$x_1(2) = x(2) + W_{16}^{0}x(10) = 1 + 0 = 1;$$
$$x_1(10) = x(2) - W_{16}^{0}x(10) = 1 - 0 = 1;$$
$$x_1(6) = x(6) + W_{16}^{0}x(14) = 1 + 1 = 2;$$
$$x_1(14) = x(6) - W_{16}^{0}x(14) = 1 - 1 = 0;$$
$$x_1(1) = x(1) + W_{16}^{0}x(9) = 0 + 0 = 0;$$
$$x_1(9) = x(1) - W_{16}^{0}x(9) = 0 - 0 = 0;$$
$$x_1(5) = x(5) + W_{16}^{0}x(13) = 1 + 1 = 2;$$
$$x_1(13) = x(5) - W_{16}^{0}x(13) = 1 - 1 = 0;$$
$$x_1(3) = x(3) + W_{16}^{0}x(11) = 1 + 1 = 2;$$
$$x_1(11) = x(3) - W_{16}^{0}x(11) = 1 - 1 = 0;$$
$$x_1(7) = x(7) + W_{16}^{0}x(15) = 0 + 0 = 0;$$
$$x_1(15) = x(7) - W_{16}^{0}x(15) = 0 - 0 = 0;$$

where the sequence $x_1(n)$ represents the intermediate output after the first iteration and becomes the input to the second stage.

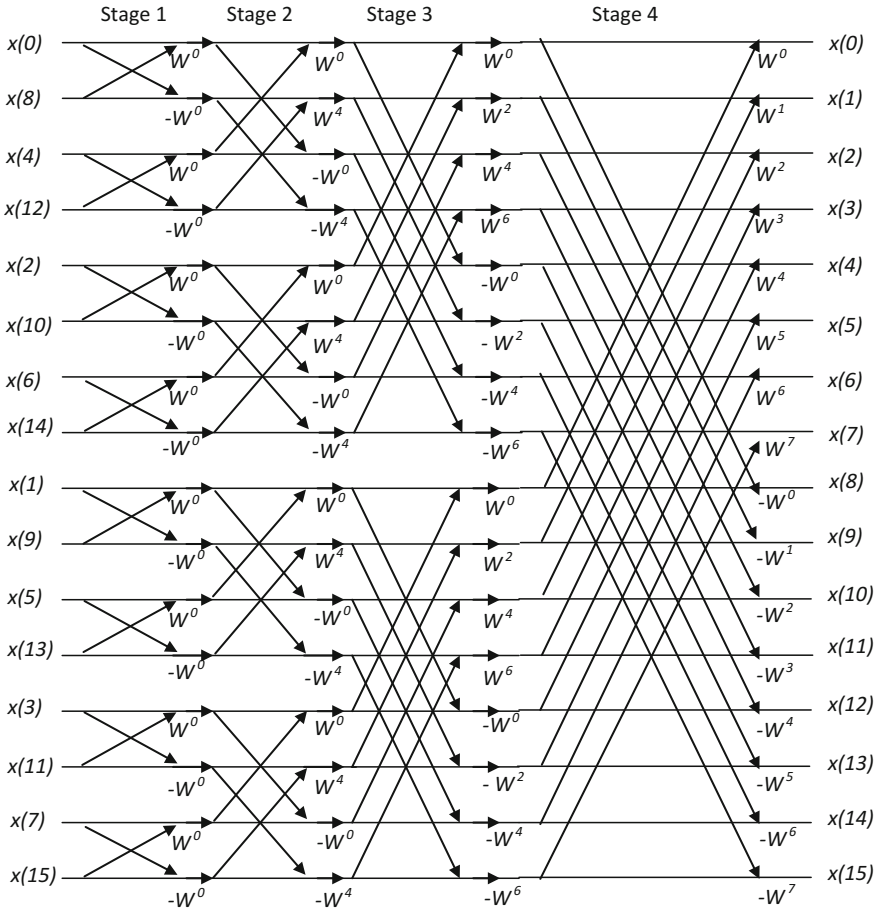


Fig. 4.12 Decomposition of the 16-point DFT of Example 4.20 using the DIT algorithm

Stage 2

$$
\begin{aligned}
x_2(0) &= x_1(0) + W_{16}^0 x_1(4) = 2 + 1 = 3;\\
x_2(8) &= x_1(8) + W_{16}^4 x_1(12) = 0 + (-j)(-1) = j;\\
x_2(4) &= x_1(0) - W_{16}^0 x_1(4) = 2 - 1 = 1;\\
x_2(12) &= x_1(8) - W_{16}^4 x_1(12) = 0 - (-j)(-1) = -j;\\
x_2(2) &= x_1(2) + W_{16}^0 x_1(6) = 1 + 2 = 3;\\
x_2(10) &= x_1(10) + W_{16}^4 x_1(14) = 1 - (-j)0 = 1;\\
x_2(6) &= x_1(2) - W_{16}^0 x_1(6) = 1 - 2 = -1;\\
x_2(14) &= x_1(10) - W_{16}^4 x_1(14) = 1 - (-j)0 = 1;\\
x_2(1) &= x_1(1) + W_{16}^0 x_1(5) = 0 + 2 = 2;\\
x_2(9) &= x_1(9) + W_{16}^4 x_1(13) = 0 + (-j)0 = 0;\\
x_2(5) &= x_1(1) - W_{16}^0 x_1(5) = 0 - 2 = -2;\\
x_2(13) &= x_1(9) - W_{16}^4 x_1(13) = 0 - (-j)0 = 0;\\
x_2(3) &= x_1(3) + W_{16}^0 x_1(7) = 2 + 0 = 2;\\
x_2(11) &= x_1(11) + W_{16}^4 x_1(15) = 0 + (-j)0 = 0;\\
x_2(7) &= x_1(3) - W_{16}^0 x_1(7) = 2 - 0 = 2;\\
x_2(15) &= x_1(11) - W_{16}^4 x_1(15) = 0 - (-j)0 = 0;
\end{aligned}
$$

where the intermediate second-stage output sequence $x_2(n)$ becomes the input sequence to the next one.

Stage 3

$$
\begin{aligned}
x_3(0) &= x_2(0) + W_{16}^0 x_2(2) = 3 + 3 = 6;\\
x_3(8) &= x_2(8) + W_{16}^2 x_2(10) = j + (0.707 - j0.707)(1) = 0.707 + j0.2929;\\
x_3(4) &= x_2(4) + W_{16}^4 x_2(6) = 1 + (-j)(-1) = 1 + j;\\
x_3(12) &= x_2(12) + W_{16}^6 x_2(14) = (-j) + (-0.707 - j0.707)(1) = -0.707 - j1.707;\\
x_3(2) &= x_2(0) - W_{16}^0 x_2(2) = 3 - 3 = 0;\\
x_3(10) &= x_2(8) - W_{16}^2 x_2(10) = j - (0.707 - j0.707)(1) = -0.707 + j1.707;\\
x_3(6) &= x_2(4) - W_{16}^4 x_2(6) = 1 - (-j)(-1) = 1 - j;\\
x_3(14) &= x_2(12) - W_{16}^6 x_2(14) = (-j) - (-0.707 - j0.707)(1) = 0.707 - j0.2929;\\
x_3(1) &= x_2(1) + W_{16}^0 x_2(3) = 2 + 2 = 4;\\
x_3(9) &= x_2(9) + W_{16}^2 x_2(11) = 0 + (0.707 - j0.707)0 = 0;\\
x_3(5) &= x_2(5) + W_{16}^4 x_2(7) = -2 + (-j)2 = -2 - 2j;\\
x_3(13) &= x_2(13) + W_{16}^6 x_2(15) = 0 - (-0.707 - j0.707)0 = 0;\\
x_3(3) &= x_2(1) - W_{16}^0 x_2(3) = 2 - 2 = 0;\\
x_3(11) &= x_2(9) - W_{16}^2 x_2(11) = 0 - (0.707 - j0.707)0 = 0;\\
x_3(7) &= x_2(5) - W_{16}^4 x_2(7) = -2 - 0 = -2;\\
x_3(15) &= x_2(13) - W_{16}^6 x_2(15) = 0 - (-j)0 = 0;
\end{aligned}
$$

where the intermediate third-stage output sequence $x_3(n)$ becomes the input sequence to the final stage.

Stage 4

$$
\begin{aligned}
X(0) &= x_3(0) + W_{16}^0 x_3(1) = 6 + 4 = 10;\\
X(1) &= x_3(8) + W_{16}^1 x_3(9) = 0.707 + j0.2929;\\
X(2) &= x_3(4) + W_{16}^2 x_3(5) = -1.8284 + j;\\
X(3) &= x_3(12) + W_{16}^3 x_3(13) = -0.707 - j1.707;\\
X(4) &= x_3(2) + W_{16}^4 x_3(3) = 0;\\
X(5) &= x_3(10) + W_{16}^5 x_3(11) = -0.707 + j1.707;\\
X(6) &= x_3(6) + W_{16}^6 x_3(7) = 3.8284 - j;\\
X(7) &= x_3(14) + W_{16}^7 x_3(15) = 0.707 - j0.2929;\\
X(8) &= x_3(0) - W_{16}^0 x_3(1) = 2;\\
X(9) &= x_3(8) - W_{16}^1 x_3(9) = 0.7071 + j0.2929;\\
X(10) &= x_3(4) - W_{16}^2 x_3(5) = 3.8284 + j;\\
X(11) &= x_3(12) - W_{16}^3 x_3(13) = -0.707 - j1.707;\\
X(12) &= x_3(2) - W_{16}^4 x_3(3) = 0;\\
X(13) &= x_3(10) - W_{16}^5 x_3(11) = -0.707 + j1.707;\\
X(14) &= x_3(6) - W_{16}^6 x_3(7) = -1.8284 - j;\\
X(15) &= x_3(14) - W_{16}^7 x_3(15) = 0.707 - j0.2929;
\end{aligned}
$$

4.7.2 In-Place Computation

In the implementation of the DIT FFT algorithm, only one complex array of N storage registers is physically necessary, since the complex numbers resulting from the mth stage can be stored in the same registers that had stored the complex numbers resulting from the $(m-1)$th stage, once the output variables of the mth stage have been determined from the output numbers of the $(m-1)$th stage. This type of computation is referred to as *in-place computation*. Thus, for in-place computation in the DIT algorithm in which the DFT samples appear in the natural order (i.e., $X(k)$, $k = 0, 1, \ldots, N-1$), the input sequence samples are to be stored in index *bit-reversed* order. If $x(b_2b_1b_0)$ represents the sample $x(n)$ in the index bit-reversed binary form, then the sample $x(b_2b_1b_0)$ would appear in the location of the sample $x(b_0b_1b_2)$ of the input sequence to the DIT algorithm. For an eight-point DFT, the bit-reversal process is shown in Table 4.5.

Table 4.5 Bit-reversal process for $N = 8$

Input sequence samples	Input sequence samples with index binary representation	Input sequence samples with bit-reversed binary index	Index bit-reversed samples
$x(0)$	$x(000)$	$x(000)$	$x(0)$
$x(1)$	$x(001)$	$x(100)$	$x(4)$
$x(2)$	$x(010)$	$x(010)$	$x(2)$
$x(3)$	$x(011)$	$x(110)$	$x(6)$
$x(4)$	$x(100)$	$x(001)$	$x(1)$
$x(5)$	$x(101)$	$x(101)$	$x(5)$
$x(6)$	$x(110)$	$x(011)$	$x(3)$
$x(7)$	$x(111)$	$x(111)$	$x(7)$

4.7.3 *Decimation-in-Frequency FFT Algorithm with Radix-2*

The basic idea in the decimation-in-time (DIT) algorithm was to decompose the input sequence successively into smaller and smaller subsequences. In the case of *decimation-in-frequency* (DIF) algorithm, we decompose the N-point DFT sequence $X(k)$ successively into smaller and smaller subsequences. Consider an input sequence $x(n)$, and divide it into two halves. Then, the DFT of $x(n)$ can be written as

$$X(k) = \sum_{n=0}^{(N/2)-1} x(n)W_N^{nk} + \sum_{n=N/2}^{(N/2)-1} x(n)W_N^{nk} \tag{4.75a}$$

The above equation can be rewritten as

$$X(k) = \sum_{n=0}^{(N/2)-1} x(n)W_N^{nk} + W_N^{kN/2} \sum_{n=N/2}^{(N/2)-1} x\left(n+\frac{N}{2}\right)W_N^{nk} \tag{4.75b}$$

Since $W_N^{Nk/2} = (-1)^k$, Eq. (4.75b) becomes

$$X(k) = \sum_{n=0}^{(N/2)-1} \left[x(n) + (-1)^k x\left(n+\frac{N}{2}\right)\right] W_N^{nk} \tag{4.75c}$$

Now, splitting $X(k)$ into even-indexed and odd-indexed samples, Eq. (4.75c) can be written as consisting of two (N/2)-point DFTs for $k = 0,1, \ldots, (N/2)-1$.

$$X(2k) = \sum_{n=0}^{(N/2)-1} \left[x(n) + x\left(n + \frac{N}{2}\right) \right] W_{N/2}^{nk} \tag{4.76a}$$

$$X(2k+1) = \sum_{n=0}^{(N/2)-1} \left[x(n) - x\left(n + \frac{N}{2}\right) \right] W_N^n W_{N/2}^{nk} \tag{4.76b}$$

Let

$$x_1(n) = x(n) + x\left(n + \frac{N}{2}\right) \quad n = 0, 1, 2, \ldots, \left(\frac{N}{2}\right) - 1 \tag{4.77a}$$

$$x_2(n) = x(n) - x\left(n + \frac{N}{2}\right) W_N^n \quad n = 0, 1, 2, \ldots, \left(\frac{N}{2}\right) - 1 \tag{4.77b}$$

Then, the even- and odd-indexed $X(k)$'s are found from the (N/2)-point transforms of $x_1(n)$ and $x_2(n)$ as

$$X(2k) = \sum_{n=0}^{(N/2)-1} x_1(n) W_{N/2}^{nk} \tag{4.78a}$$

and

$$X(2k+1) = \sum_{n=0}^{(N/2)-1} x_2(n) W_{N/2}^{nk} \tag{4.78b}$$

Repeating the process for each of the sequences $x_1(n)$ and $x_2(n)$ yields the two (N/4)-point sequences

$$\begin{aligned} x_{11}(n) &= x_1(n) + x_1(n + \frac{N}{4}) \qquad n = 0, 1, \ldots, (N/4) - 1 \\ x_{12}(n) &= \left(x_1(n) - x_1\left(n + \frac{N}{4}\right) \right) W_N^{2n} \qquad n = 0, 1, \ldots, (N/4) - 1 \end{aligned} \tag{4.79a}$$

and $x_2(n)$ yields

$$\begin{aligned} x_{21}(n) &= x_2(n) + x_2(n + \frac{N}{4}) \qquad n = 0, 1, \ldots, (N/4) - 1 \\ x_{22}(n) &= \left(x_2(n) - x_2\left(n + \frac{N}{4}\right) \right) W_N^{2n} \qquad n = 0, 1, \ldots, (N/4) - 1 \end{aligned} \tag{4.79b}$$

Then, the even- and odd-indexed $X(k)$'s are found from the (N/4)-point transforms of $x_{11}(n), x_{12}(n), x_{21}(n)$ and $x_{22}(n)$ as

$$
\begin{aligned}
X(4k) &= \sum_{n=0}^{(N/4)-1} x_{11}(n) W_{N/4}^{nk} \\
X(4k+2) &= \sum_{n=0}^{(N/4)-1} x_{12}(n) W_{N/4}^{nk} \\
X(4k+1) &= \sum_{n=0}^{(N/4)-1} x_{21}(n) W_{N/4}^{nk} \\
X(4k+3) &= \sum_{n=0}^{(N/4)-1} x_{22}(n) W_{N/4}^{nk}
\end{aligned}
\tag{4.80}
$$

The process is to be continued until they reduce to two-point transforms.

For example, for $N = 4$, the two twiddle factors needed are $W_4^0 = 1$ and $W_4^1 = -j$. The DIF flow graph for a four-point DFT contains two stages as shown in Fig. 4.13. The outputs of the two stages are computed as follows:

Stage 1

$$
\begin{aligned}
x_1(0) &= x(0) + x(2) \\
x_1(1) &= x(1) + x(3) \\
x_1(2) &= [x(0) - x(2)] w_4^0 \\
x_1(3) &= [x(1) - x(3)] w_4^1
\end{aligned}
$$

where $x_1(0)$, $x_1(1)$, $x_1(2)$ and $x_1(3)$ represent the intermediate output sequence after the first stage, which become the input to the second stage.

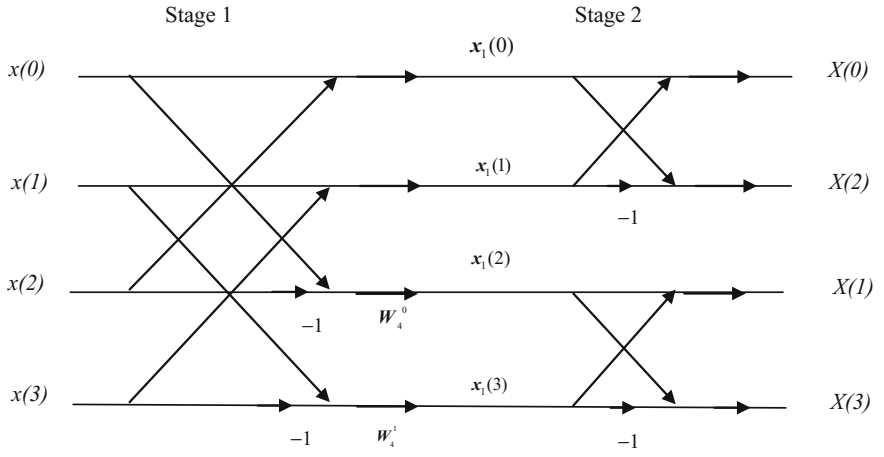


Fig. 4.13 Decomposition of a four-point DFT using the DIF algorithm

Stage 2

$$X(0) = x_1(0) + x_1(1)$$
$$X(1) = x_1(2) + x_1(3)$$
$$X(2) = x_1(0) - x_1(1)$$
$$X(3) = x_1(2) - x_1(3)$$

For $N = 8$, the decomposition of an 8-point DFT into two 4-point DFTS with DIF algorithm is shown in Fig. 4.14.

Example 4.21 Find the DFT of the sequence $x(n) = (1, 2, 3, 4)$ using the DIF algorithm.

Solution The two twiddle factors needed are $W_4^0 = 1$ and $W_4^1 = -j$.

The DIF flow graph for four-point DFT consists of two stages as shown in Fig. 4.15. The outputs of the two stages are computed as follows:

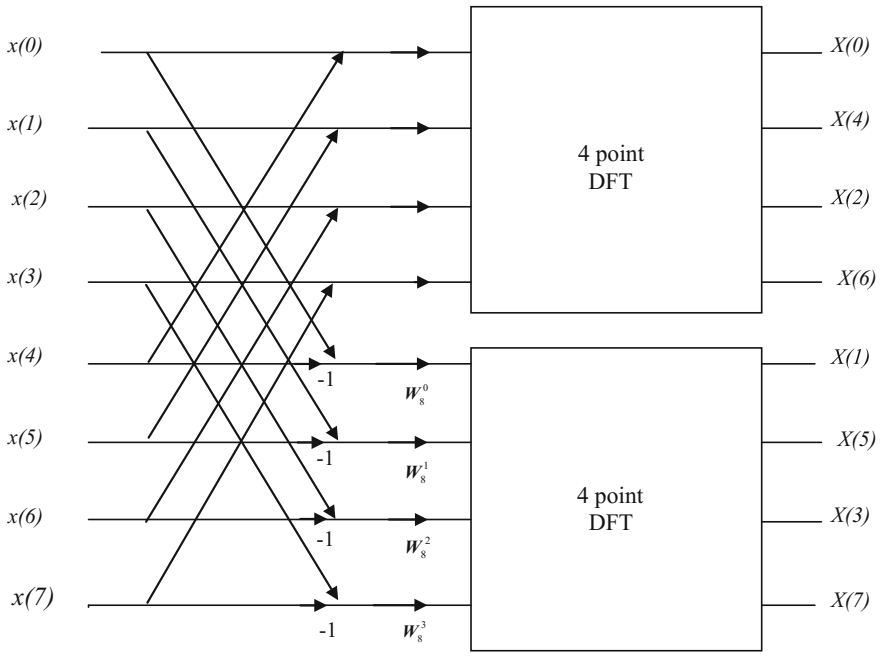


Fig. 4.14 Decomposition of an eight-point DFT using the DIF algorithm decimation-in-frequency

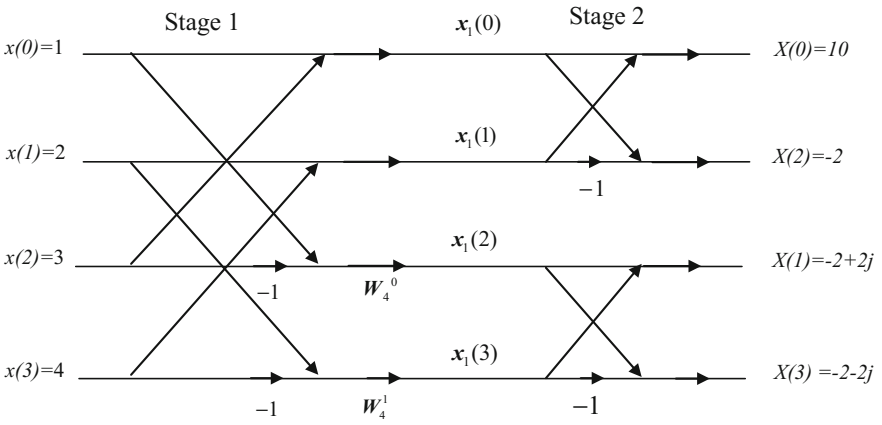


Fig. 4.15 Flow graph for the four-point FFT of Example 4.21 using the DIF algorithm

Stage 1

$$\begin{aligned}
\boldsymbol{x}_1(0) &= \boldsymbol{x}(0) + \boldsymbol{x}(2) = 4 \\
\boldsymbol{x}_1(1) &= \boldsymbol{x}(1) + x(3) = 6 \\
\boldsymbol{x}_1(2) &= [\boldsymbol{x}(0) - \boldsymbol{x}(2)]\boldsymbol{W}_4^0 = -2 \\
\boldsymbol{x}_1(3) &= [\boldsymbol{x}(1) - \boldsymbol{x}(3)]\boldsymbol{W}_4^1 = 2\boldsymbol{j}
\end{aligned}$$

where $x_1(0)$, $x_1(1)$, $x_1(2)$ and $x_1(3)$ represent the intermediate output sequence after the first stage, which become the input to the second stage.

Stage 2

$$\begin{aligned}
\boldsymbol{X}(0) &= \boldsymbol{x}_1(0) + \boldsymbol{x}_1(1) = 10 \\
\boldsymbol{X}(2) &= \boldsymbol{x}_1(0) - \boldsymbol{x}_1(1) = -2 \\
\boldsymbol{X}(1) &= \boldsymbol{x}_1(2) + \boldsymbol{x}_1(3) = -2 + 2\boldsymbol{j} \\
\boldsymbol{X}(3) &= \boldsymbol{x}_1(2) - \boldsymbol{x}_1(3) = -2 - 2\boldsymbol{j}
\end{aligned}$$

Example 4.22 Find the DFT of a sequence $x(n) = (1, 1, 1, 1, 1, 1, 0, 0)$ using the DIF algorithm.

Solution With $N = 8$, the four twiddle factors needed are

$$\begin{aligned}
\boldsymbol{W}_8^0 &= 1;\\
\boldsymbol{W}_8^1 &= \mathbf{e}^{-j2\pi/8} = \mathbf{e}^{-j\pi/4} = 0.707 - \boldsymbol{j}0.707;\\
\boldsymbol{W}_8^2 &= \mathbf{e}^{-j4\pi/8} = \mathbf{e}^{-j\pi/2} = -\boldsymbol{j};\\
\boldsymbol{W}_8^3 &= \mathbf{e}^{-j6\pi/8} = \mathbf{e}^{-j3\pi/4} = -0.707 - \boldsymbol{j}0.707;
\end{aligned}$$

Stage 1

$$\begin{aligned}
x_1(0) &= x(0) + x(4) = 2;\\
x_1(1) &= x(1) + x(5) = 2;\\
x_1(2) &= x(2) + x(6) = 1;\\
x_1(3) &= x(3) + x(7) = 1;\\
x_1(4) &= [x(0) - x(4)]W_8^0 = 0;\\
x_1(5) &= [x(1) - x(5)]W_8^1 = 0;\\
x_1(6) &= [x(2) - x(6)]W_8^2 = -j;\\
x_1(7) &= [x(3) - x(7)]W_8^3 = -0.707 - j0.707;
\end{aligned}$$

where $x_1(0), x_1(1), \ldots, x_1(7)$ represent the intermediate output sequence after the first stage, which become the input to the second stage.

Stage 2

$$\begin{aligned}
x_2(0) &= x_1(0) + x_1(2) = 3;\\
x_2(1) &= x_1(1) + x_1(3) = 3;\\
x_2(2) &= [x_1(0) - x_1(2)]W_8^0 = 1;\\
x_2(3) &= [x_1(1) - x_1(3)]W_8^2 = -j;\\
x_2(4) &= x_1(4) + x_1(6) = -j;\\
x_2(5) &= x_1(5) + x_1(7) = -0.707 - j0.707;\\
x_2(6) &= [x_1(4) - x_1(6)]W_8^0 = j;\\
x_2(7) &= [x_1(5) - x_1(7)]W_8^2 = 0.707 - j0.707;
\end{aligned}$$

where $x_2(0),\ x_2(1),\ \ldots,\ x_2(7)$ represent the intermediate output sequence after the second stage, which become the input to the final stage.

Stage 3

We now use the notation of X's to represent the final output sequence. The values $X(0),\ X(1),\ \ldots,\ X(7)$ form the output sequence.

$$
\begin{aligned}
X(0) &= x_2(0) + x_2(1) = 6;\\
X(4) &= x_2(0) - x_2(1) = 0;\\
X(2) &= x_2(2) + x_2(3) = 1 - j1;\\
X(6) &= x_2(2) - x_2(3) = 1 + j1;\\
X(1) &= x_2(4) + x_2(5) = -0.707 - j1.707;\\
X(5) &= x_2(4) - x_2(5) = 0.707 - j0.2929;\\
X(3) &= x_2(6) + x_2(7) = 0.707 + j0.2929;\\
X(7) &= x_2(6) - x_2(7) = -0.707 + j1.707;
\end{aligned}
$$

The DIF flow graph for eight-point DFT consists of three stages as shown in Fig. 4.16. The outputs of the three stages are computed in Fig. 4.16.

It should be noted that flow graph representing the DIF FFT may be considered as an in-place computation, just as in the case of the DIT FFT. Further, it should be noted that the input sequence $x(n)$ is in order, while the output sequence $X(k)$ is in bit-reversed order. The number of multiplications and additions for computing an N-point by DIF FFT is the same as in the case of the DIT FFT, namely $(N/2)\log_2 N$ and $N\log_2 N$, respectively.

It is worth pointing out that the flow graphs of DIT FFT and DIF FFT algorithms are transposes of one another.

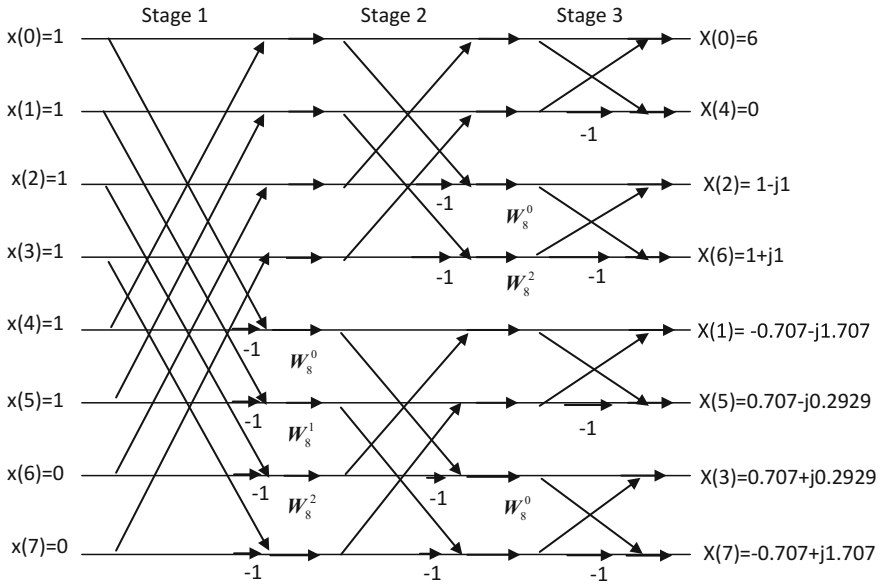


Fig. 4.16 Flow graph of the eight-point FFT for the Example 4.22 using DIF algorithm

4.7.4 Radix-4 DIF FFT Algorithm

If $N = 2^{2v}$, then we can use radix-4 algorithms rather than radix-2 algorithms, and this gives us a reduction in the number of multiplications to be performed. Here, we will consider the radix-4 DIF algorithm. Radix-4 DIT algorithm can be developed in a way similar to that of the radix-2 DIT algorithm.

Consider a sequence $x(n)$, and divide it into four parts so that the DFT of $x(n)$ can be written as

$$X(k) = \sum_{n=0}^{(N/4)-1} x(n)W_N^{nk} + \sum_{n=N/4}^{(N/2)-1} x(n)W_N^{nk} + \sum_{n=N/2}^{(3N/4)-1} x(n)W_N^{nk} + \sum_{n=3N/4}^{N-1} x(n)W_N^{nk} \tag{4.81}$$

The above equation can be rewritten as

$$\begin{aligned} X(k) = &\sum_{n=0}^{(N/4)-1} x(n)W_N^{nk} + W^{kN/4} \sum_{n=0}^{(N/4)-1} x(n+N/4)W_N^{nk} \\ &+ W^{kN/2} \sum_{n=0}^{(N/4)-1} x(n+N/2)W_N^{nk} + W^{k3N/4} \sum_{n=0}^{(N/4)-1} x(n+3N/4)W_N^{nk} \end{aligned} \tag{4.82}$$

Substituting

$$W_N^{kN/4} = e^{-jk\pi/2} = (-j)^k; W_N^{kN/2} = e^{-jk\pi} = (-1)^k; W_N^{3kN/4} = (j)^k$$

in the above equation, we get

$$X(k) = \sum_{n=0}^{\frac{N}{4}-1} \left[x(n) + (-j)^k x\left(n + \frac{N}{4}\right) + (-1)^k x\left(n + \frac{N}{2}\right) + (j)^k x\left(n + \frac{3N}{4}\right) \right] W_N^{nk} \tag{4.83}$$

Since the twiddle factor depends on N, the above relation is not $N/4$-point DFT. To represent it as an $N/4$-point DFT, the DFT sequence is divided into four $N/4$-point subsequences, $X(4k)$, $X(4k+1)$, $X(4k+2)$ and $X(4k+3)$ for $k = 0, 1, \ldots \left(\frac{N}{4} - 1\right)$. Thus, the DIF FFT with radix-4 can be represented as

$$X(4k) = \sum_{n=0}^{(N/4)-1} [x(n) + x(n+N/4) + x(n+N/2) + x(n+3N/4)] W_{N/4}^{nk} \tag{4.84}$$

$$\boldsymbol{X}(4\boldsymbol{k}+1)=\sum_{\boldsymbol{n}=0}^{(\boldsymbol{N}/4)-1}[\boldsymbol{x}(\boldsymbol{n})-\boldsymbol{j}\boldsymbol{x}(\boldsymbol{n}+\boldsymbol{N}/4)-\boldsymbol{x}(\boldsymbol{n}+\boldsymbol{N}/2)+\boldsymbol{j}\boldsymbol{x}(\boldsymbol{n}+3\boldsymbol{N}/4)]\boldsymbol{W}_{\boldsymbol{N}}^{\boldsymbol{n}}\boldsymbol{W}_{\boldsymbol{N}/4}^{\boldsymbol{nk}} \tag{4.85}$$

$$\boldsymbol{X}(4\boldsymbol{k}+2)=\sum_{\boldsymbol{n}=0}^{(\boldsymbol{N}/4)-1}[\boldsymbol{x}(\boldsymbol{n})-\boldsymbol{x}(\boldsymbol{n}+\boldsymbol{N}/4)+\boldsymbol{x}(\boldsymbol{n}+\boldsymbol{N}/2)-\boldsymbol{x}(\boldsymbol{n}+3\boldsymbol{N}/4)]\boldsymbol{W}_{\boldsymbol{N}}^{\boldsymbol{2n}}\boldsymbol{W}_{\boldsymbol{N}/4}^{\boldsymbol{nk}} \tag{4.86}$$

$$X(4k+3)=\sum_{n=0}^{(N/4)-1}[x(n)+jx(n+N/4)-x(n+N/2)-jx(n+3N/4)]W_N^{3n}W_{N/4}^{nk} \tag{4.87}$$

The following example illustrates a 16-point radix-4 FFT using the DIF procedure.

Example 4.23 Find the DFT of a sequence $x(n)=\{1,1,0,1,1,0,1,1,0,1,1,1,1,1,1,1\}$ using the radix-4 DIF algorithm.

Solution The **t**widdle factors for 16-point radix-4 FFT are

$$W_{16}^0=1; W_{16}^1=0.9238-j0.3826; W_{16}^2=0.707-j0.707;$$
$$W_{16}^3=0.3826-j0.9238; W_{16}^4=0-j; W_{16}^5=-0.3826-j0.9238;$$
$$W_{16}^6=-0.707-j0.707; W_{16}^7=-0.9238-j0.3826.$$
$$W_4^0=1; W_4^1=-j; W_4^2=-1; W_4^3=+j; W_4^4=1; W_4^5=-j;$$
$$W_4^6=-1; W_4^7=+j;$$

The outputs of the two stages are computed as follows:

Stage 1

$$x_1(0) = [x(0) + x(4) + x(8) + x(12)]W_{16}^0 = 1 + 1 + 0 + 1 = 3;$$
$$x_1(1) = [x(1) + x(5) + x(9) + x(13)]W_{16}^0 = 1 + 0 + 1 + 1 = 3;$$
$$x_1(2) = [x(2) + x(6) + x(10) + x(14)]W_{16}^0 = 0 + 1 + 1 + 1 = 3;$$
$$x_1(3) = [x(3) + x(7) + x(11) + x(15)]W_{16}^0 = 1 + 1 + 1 + 1 = 4;$$
$$x_1(4) = [x(0) - jx(4) - x(8) + jx(12)]W_{16}^0 = 1 - j - 0 + j = 1;$$
$$x_1(5) = [x(1) - jx(5) - x(9) + jx(13)]W_{16}^1 = (1 - 0 - 1 + j)W_{16}^1 = 0.3826 + j0.9238;$$
$$x_1(6) = [x(2) - jx(6) - x(10) + jx(14)]W_{16}^2 = (0 - j - 1 + j)W_{16}^2 = -0.707 + j0.707;$$
$$x_1(7) = [x(3) - jx(7) - x(11) + jx(15)]W_{16}^3 = (1 - j - 1 + j)W_{16}^3 = 0;$$
$$x_1(8) = [x(0) - x(4) + x(8) - x(12)]W_{16}^0 = 1 - 1 + 0 - 1 = -1;$$
$$x_1(9) = [x(1) - x(5) + x(9) - x(13)]W_{16}^2 = (1 - 0 + 1 - 1)W_{16}^2 = 0.707 - j0.707;$$
$$x_1(10) = [x(2) - x(6) + x(10) - x(14)]W_{16}^4 = (0 - 1 + 1 - 1)W_{16}^4 = j;$$
$$x_1(11) = [x(3) - x(7) + x(11) - x(15)]W_{16}^6 = (1 - 1 + 1 - 1)W_{16}^6 = 0;$$
$$x_1(12) = [x(0) + jx(4) - x(8) - jx(12)]W_{16}^0 = (1 + j - 0 - j)W_{16}^0 = 1;$$
$$x_1(13) = [x(1) + jx(5) - x(9) - jx(13)]W_{16}^3 = (1 - 0 - 1 - j)W_{16}^3 = -0.9238 - j0.3826;$$
$$x_1(14) = [x(2) + jx(6) - x(10) - jx(14)]W_{16}^6 = (0 + j - 1 - j)W_{16}^6 = 0.707 + j0.707;$$
$$x_1(15) = [x(3) + jx(7) - x(11) - jx(15)]W_{16}^9 = (1 + j - 1 - j)W_{16}^9 = (1 + j - 1 - j)W_{16}^{-1} = 0;$$

Stage 2

$$X(0) = [x_1(0) + x_1(1) + x_1(2) + x_1(3)]W_{16}^0 = 3 + 3 + 3 + 4 = 13;$$
$$X(1) = [x_1(4) + x_1(5) + x_1(6) + x_1(7)]W_{16}^0 = 0.6756 + j1.6310;$$
$$X(2) = [x_1(8) + x_1(9) + x_1(10) + x_1(11)]W_{16}^0 = -0.2929 + j0.2929;$$
$$X(3) = [x_1(12) + x_1(13) + x_1(14) + x_1(15)]W_{16}^0 = 0.7832 + j0.3244;$$
$$X(4) = x_1(0) + jx_1(1) - x_1(2) + jx_1(3) = j;$$
$$X(6) = x_1(8) + jx_1(9) - x_1(10) + jx_1(11) = -1.7071 - j1.7071;$$
$$X(7) = x_1(12) + jx_1(13) - x_1(14) + jx_1(15) = -0.0898 + j0.2168;$$
$$X(8) = x_1(0) + x_1(1) - x_1(2) - x_1(3) = -1;$$
$$X(9) = x_1(4) + x_1(5) - x_1(6) - x_1(7) = -0.0898 - j0.2168;$$
$$X(10) = x_1(8) + x_1(9) - x_1(10) - x_1(11) = -1.7071 + j1.7071;$$
$$X(11) = x_1(12) + x_1(13) - x_1(14) - x_1(15) = 2.6310 + j1.0898;$$
$$X(12) = x_1(0) - jx_1(1) + x_1(2) - jx_1(3) = -j;$$
$$X(13) = x_1(4) - jx_1(5) + x_1(6) - jx_1(7) = 0.7832 - j0.3244;$$
$$X(14) = x_1(8) - jx_1(9) + x_1(10) - jx_1(11) = -0.2929 - j0.2929;$$
$$X(15) = x_1(12) - jx_1(13) + x_1(14) - jx_1(15) = 0.6756 - j1.6310;$$

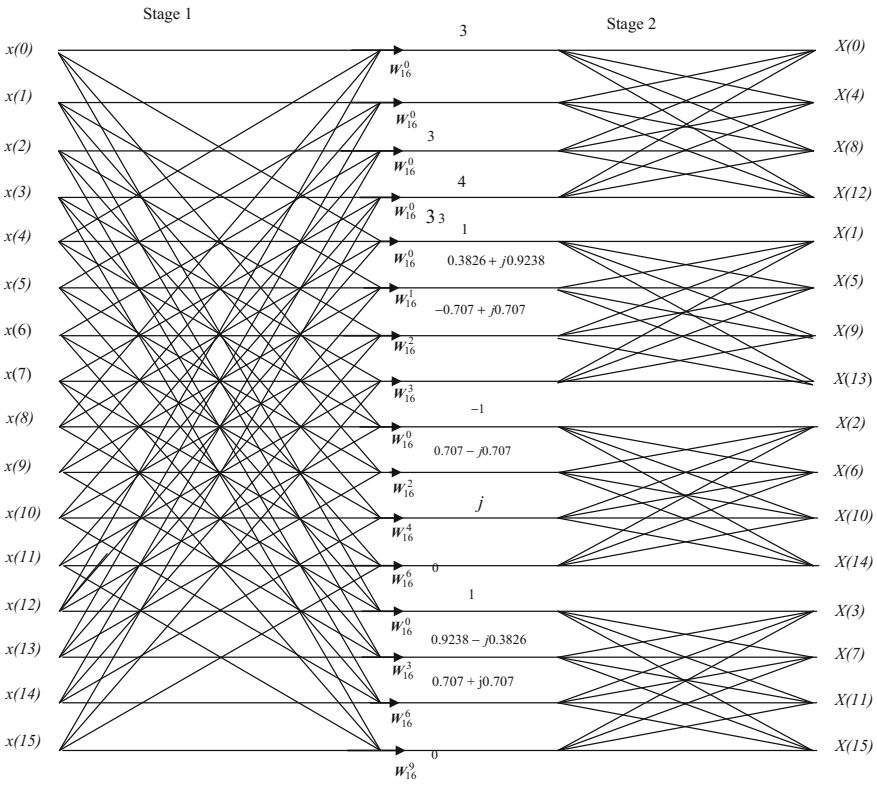


Fig. 4.17 Sixteen-point DFT of Example 4.23 using radix-4 DIF algorithm

The flow graph for the 16-point radix-4 DIF FFT is shown in Fig. 4.17. The (±) j and −1 are not shown in stage 2 for the four-point butterfly of the flow graph.

4.8 Comparison of Computational Complexity

As mentioned earlier, the number of complex multiplications required in the radix-2 FFT of an N-point sequence is $(N/2)\log_2 N$ while the number of complex additions needed is $N\log_2 N$.

In the radix-4 FFT of an N-point sequence, there are $\log_4 N = (1/2)\log_2 N$ stages and (N/4) butterflies per stage. Each radix-4 butterfly requires three complex multiplications and eight complex additions. Thus, it requires $(3N/4)(1/2)\log_2 N = (3N/8)\log_2 N$ complex multiplications and $(8N/4)(1/2)\log_2 N = N\log_2 N$ complex additions.

A comparison of the computational complexity in terms of the number of complex multiplications needed to compute the DFT of an N-point sequence

Table 4.6 Comparison of the computational complexity for direct DFT and FFT

Number of points N	Number of complex multiplications			FFT speed improvement factor	
	Direct DFT N^2	Radix-2 FFT $(\frac{N}{2})\log_2 N$	Radix-4 FFT $(\frac{3N}{8})\log_2 N$	Radix-2	Radix-4
16	256	32	24	8	10.6667
64	4096	192	144	21.3333	28.4444
256	65536	1024	768	64	85.3333
1024	1,048,576	5120	3840	204.8	273.0667

directly is compared to that required using radix-2 and radix-4 FFTs as given in Table 4.6.

4.9 DFT Computation Using the Goertzel Algorithm and the Chirp Transform

While the fast Fourier transform's various incarnations have gained considerable popularity, careful selection of an appropriate algorithm for computing the DFT in practice need not be limited to choosing between these so-called fast implementations. In this section, it is focused on two other techniques, namely the Goertzel algorithm and the chirp transform for computing the DFT.

4.9.1 The Goertzel Algorithm

The Goertzel algorithm [3] uses the periodicity of the sequence W_N^{nk} to reduce the computational complexity. From the definition of DFT, it is known that

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk}, \; W_N = e^{\frac{-j2\pi}{N}} \tag{4.88a}$$

Equation (4.88a) can be rewritten as

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{-k(N-n)}, \; W_N^{-kN} = 1$$

If a sequence $y_k(n)$ is defined as

$$y_k(n) = \sum_{r=0}^{N-1} x(r) W_N^{-k(n-r)} \tag{4.88b}$$

implying that passing a signal $x(n)$ through an LTI filter with impulse response $h(n) = W_N^{-nk} u(n)$ and evaluating the result, $y_k(n)$ at $n = N$ will give the corresponding N-point DFT coefficient $X(k) = y_k(n)$.

Representing the filter by its z-transform, we obtain

$$\begin{aligned} H_k(z) &= \sum_{n=0}^{\infty} W_N^{-nk} z^{-n} \\ &= \frac{1}{1 - W_N^{-k} z^{-1}} \end{aligned} \tag{4.89}$$

having a pole on the unit circle at the frequency $\omega_k = \frac{2\pi k}{N}$. Hence, the DFT can be computed by passing the block of input data into a parallel bank of N filters each filter having a pole at the frequency of the corresponding DFT. The DFT can be computed by using the following difference equation corresponding to the filter expressed by Eq. (4.89)

$$y_k(n) = W_N^{-k} y_k(n-1) + x(n) y_k(-1) = 0. \tag{4.90a}$$

The inherent complex multiplications and addition in Eq. (4.90a) can be avoided by using the following two-pole filter having complex conjugate pole pairs equivalent to the filtering operation represented by Eq. (4.89).

$$\begin{aligned} H_k(z) &= \frac{1 - W_N^k z^{-1}}{1 - W_N^k z^{-1}} \frac{1}{1 - W_N^{-k} z^{-1}} \\ &= \frac{1 - W_N^k z^{-1}}{1 - \left(2\cos\frac{2\pi k}{N}\right) z^{-1} + z^{-2}} \\ &= \frac{Y_k(z)}{X(z)} \end{aligned} \tag{4.90b}$$

where $= H_{1k}(z) H_{2k}(z)$

$$H_{2k}(z) = \frac{Y_k(z)}{v_k(z)} = 1 - W_N^k z^{-1}$$

$$H_{1k}(z) = \frac{v_k(z)}{X(z)} = \frac{1}{1 - \left(2\cos\frac{2\pi k}{N}\right) z^{-1} + z^{-2}}$$

From $H_{1k}(z)$ and $H_{2k}(z)$, we obtain the following difference equations

$$v_k(n) = 2\cos\frac{2\pi k}{N} v_k(n-1) - v_k(n-2) + x(n) \tag{4.91a}$$

$$y_k(n) = v_k(n) - W_N^k v_k(n-1) \tag{4.91b}$$

with initial conditions $v_k(-1) = v_k(-2) = 0$.

The Goertzel algorithm evaluates $X(k)$ at any M values of k instead of evaluating at all N values of k. Hence, it is more efficient than FFT [4] for computing DFT, when $M \leq \log_2(N)$.

Example 4.24 Considering the sequence $x(n) = \{1, 2, 1, 1\}$, compute DFT coefficient $X(1)$ and the corresponding spectral amplitude at the frequency bin $k = 1$ using the Goertzel algorithm.

Solution We have $k = 1$, $N = 4$, $x(0) = 1,\ x(1) = 2,\ x(2) = 1,\ x(3) = 1$.

$$2\cos\frac{2\pi}{4} = 0, W_4^1 = e^{-\frac{j2\pi}{4}} = \cos\frac{\pi}{2} - j\,\sin\frac{\pi}{2} = -j$$

For $n = 0, 1, \ldots, 4$

$$v_1(n) = -v_1(n-2) + x(n)$$
$$y_1(n) = v_1(n) + jv_1(n-1)$$

Then, $X(1) = y_1(4) \quad |X(1)|^2 = v_1^2(4) + v_1^2(3)$

$$X(1) = y_1(4) = v_1(4) + jv_1(3)$$
$$v_1(0) = -v_1(-2) + x(0) = 1$$
$$y_1(0) = v_1(0) + jv_1(-1) = 1$$
$$v_1(1) = -v_1(-1) + x(1) = 2$$
$$y_1(1) = v_1(1) + jv_1(0) = 2 + j1$$
$$v_1(2) = -v_1(0) + x(2) = 0$$
$$y_1(2) = v_1(2) + jv_1(1) = j2$$
$$v_1(3) = -v_1(1) + x(3) = -1$$
$$y_1(3) = v_1(3) + jv_1(2) = -1$$
$$v_1(4) = -v_1(2) + x(4) = 0$$
$$y_1(4) = v_1(4) + jv_1(3) = -j$$
$$X(1) = y_1(4) = -j$$
$$|X(1)|^2 = v_1^2(4) + v_1^2(3) = 1$$

4.9.2 The Chirp Transform Algorithm

The chirp transform algorithm [5] is also based on expressing DFT as a convolution. As it can be used to compute the Fourier transform of any set of equally spaced samples on the unit circle, it is more flexible than the FFT.

If it is desired to compute the values of the z-transform of $x(n)$ at a set of points $\{z_k\}$, then,

$$X(z_k) = \sum_{n=0}^{N-1} x(n) z_k^{-n} \qquad k = 0, 1, \ldots, M-1 \tag{4.92a}$$

Equation (4.92a) can be rewritten as

$$X\left(e^{j\omega_k}\right) = \sum_{n=0}^{N-1} x(n) e^{-j\omega_k n} \qquad k = 0, 1, \ldots, M-1 \tag{4.92b}$$

where

$$\omega_k = \omega_0 + k\Delta\omega \qquad k = 0, 1, \ldots, M-1 \tag{4.92c}$$

Equation (4.92b) can be rewritten as,

$$X\left(e^{j\omega_k}\right) = \sum_{n=0}^{N-1} x(n) e^{-j(\omega_0 + k\Delta\omega)n} \qquad k = 0, 1, \ldots, M-1 \tag{4.92d}$$

For the DFT computation, $\omega_0 = 0$, $\Delta\omega = \frac{2\pi}{N}$ and $M = N$.
Hence, Eq. (4.92d) becomes

$$X\left(e^{j\omega_k}\right) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi}{N}nk} \qquad k = 0, 1, \ldots, M-1 \tag{4.93a}$$

$$X\left(e^{j\omega_k}\right) = \sum_{n=0}^{N-1} x(n) W_N^{nk} \qquad k = 0, 1, \ldots, M-1 \tag{4.93b}$$

Using the identity

$$nk = \frac{1}{2}\left(n^2 + k^2 - (k-n)^2\right)$$

Equation (4.93b) can be written as

$$X(z_k) = W_N^{\frac{k^2}{2}} \sum_{n=0}^{N-1} g(n) W_N^{\frac{-(k-n)^2}{2}} \qquad k = 0, 1, \ldots, M-1 \tag{4.93c}$$

where

$$g(n) = x(n) W_N^{\frac{n^2}{2}}$$

For notation convenience, replacing n by k and k by n in Eq. (4.93c), it can be rewritten as

$$X(z_n) = W_N^{\frac{n^2}{2}} \sum_{n=0}^{N-1} g(k) W_N^{\frac{-(n-k)^2}{2}} \qquad n = 0, 1, \ldots, M-1 \tag{4.94a}$$

Equation (4.94a) can also be expressed as

$$X\left(e^{j\omega_n}\right) = W_N^{\frac{n^2}{2}} \sum_{n=0}^{N-1} g(k) W_N^{\frac{-(n-k)^2}{2}} \qquad n = 0, 1, \ldots, M-1 \tag{4.94b}$$

implying that $X(e^{j\omega_n})$ is the convolution of the sequence $g(n)$ with the sequence $W_N^{\frac{-n^2}{2}}$, premultiplied by the sequence $W_N^{\frac{n^2}{2}}$, and the chirp filter impulse response is

$$h(n) = W_N^{\frac{-n^2}{2}} = \cos\frac{\pi n^2}{N} + j \sin\frac{\pi n^2}{N} \tag{4.95}$$

Thus, the block diagram of chirp transform system for DFT computation is shown in Fig. 4.18.

4.10 Decimation-in-Time FFT Algorithm for a Composite Number

In the previous sections, we discussed FFT algorithms for radix-2 and radix-4 cases. However, it may not be possible in all cases to choose N to be a power of 2 or 4. We now consider the case where N is a composite number composed of a product

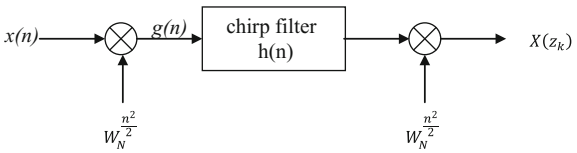


Fig. 4.18 Block diagram of chirp transform system for DFT computation

of two factors n_1 and n_2, i.e., $N = n_1 n_2$, so that we can divide the sequence $x(n)$ into n_1 subsequences of length n_2. Then, $X(K)$ can be written as

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn} \tag{4.88}$$

$$\begin{aligned} &= \sum_{i=0}^{n_2-1} x(n_1 i) W_N^{n_1 ik} + \sum_{i=0}^{n_2-1} x(n_1 i + 1) W_N^k W_N^{n_1 ik} + \cdots \\ &\quad + \sum_{i=0}^{n_2-1} x(n_1 i + n_1 - 1) W_N^{(n_1-1)k} W_N^{n_1 ik} \end{aligned} \tag{4.89}$$

The above equation can be rewritten as

$$X(k) = \sum_{j=0}^{n_1-1} W_N^{jk} \sum_{i=0}^{n_2-1} x(n_1 i + j) W_N^{n_1 ik} \tag{4.90}$$

Define

$$F_j(k) = \sum_{i=0}^{n_2-1} x(n_1 i + j) W_N^{n_1 ik} \tag{4.91}$$

Then, $X(k)$ can be expressed in terms of n_1 DFTs of sequences of length n_2 samples as

$$X(k) = \sum_{j=0}^{n_1-1} F_j(k) W_N^{jk} \tag{4.92}$$

For illustration, consider computation of a 12-point DIT FFT (N = 12 = 3.4). The original sequence is divided into three sequences, each of length 4.

First sequence: $x(0)x(3)x(6)x(9)$; second sequence: $x(1)x(4)x(7)x(10)$;

Third sequence: $x(2)x(5)x(8)x(11)$. Then, $X(k)$ can be expressed as

$$\begin{aligned} X(k) &= \sum_{j=0}^{2} W_{12}^{jk} \sum_{i=0}^{3} x(3i + j) W_{12}^{3ik} \\ &= F_0(k) + W_{12}^k F_1(k) + W_{12}^{2k} F_2(k) \end{aligned} \tag{4.93}$$

The flow graph of the 12-point DFT is shown in Fig. 4.19.

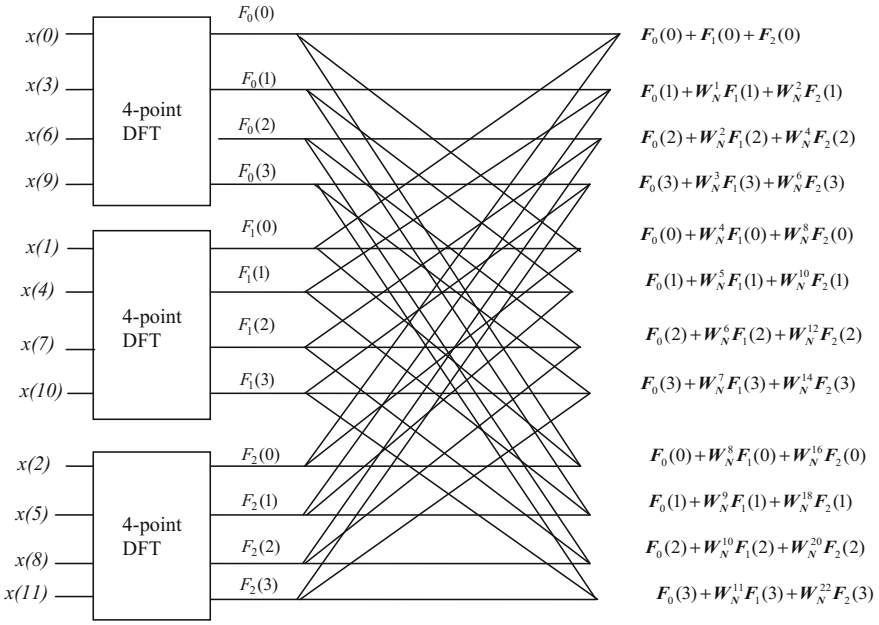


Fig. 4.19 Flow graph of a 12-point DIT FFT

4.11 The Inverse Discrete Fourier Transform

An FFT algorithm for computing the DFT can be effectively used to compute the inverse DFT. The inverse of an N-point DFT $X(k)$ is given by

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k) W_N^{-nk} \tag{4.94}$$

where $W = \mathrm{e}^{-j2\pi/N}$. Multiplying both sides of the above expression by N and taking complex conjugates, we obtain

$$Nx^*(n) = \sum_{k=0}^{N-1} X^*(k)\ W_N^{nk} \tag{4.95}$$

The RHS of Eq. (4.94) is the DFT of the sequence $X^*(k)$ and can be rewritten as

$$Nx^*(n) = \mathrm{DFT}\{X^*(k)\} \tag{4.96}$$

Taking the complex conjugate on both sides of Eq. (4.96) and using the FFT for the computation of DFT yield

$$Nx(n) = \{\text{FFT}\{X^*(k)\}\}^*$$

Hence,

$$x(n) = \frac{1}{N}\{\text{FFT}\{X^*(k)\}\}^* \tag{4.97}$$

The following example illustrates the IDFT computation using the DIT FFT algorithm:

Example 4.25 Find the eight-point IDFT using DIT algorithm.

Solution Let the input be

$$X(k) = \{20, -5.828 - j2.279, 0, -0.172 - j0.279, 0, -0.172 + j0.279, 0, -5.828 + j2.279\}$$

Hence,

$$X^*(k) = \{20, -5.828 + j2.279, 0, -0.172 + j0.279, 0, -0.172 - j0.279, 0, -5.828 - j2.279\}$$

With $N = 8$, the four twiddle factors are

$$W_8^0 = 1; W_8^1 = e^{-j2\pi/8} = \cos(\pi/4) - j\sin(\pi/4) = 0.707 - j0.707;$$
$$W_8^2 = e^{-j4\pi/8} = -j; W_8^3 = e^{-j6\pi/8} = -0.707 - j0.707;$$

The flow diagram for the eight-point inverse DFT using the DIT algorithm is shown in Fig. 4.20.

Stage 1

$$x_1(0) = X^*(0) + W_8^0 X^*(4) = 20 + 0 = 20$$
$$x_1(4) = X^*(0) - W_8^0 X^*(4) = 20 - 0 = 20$$
$$x_1(2) = X^*(2) + W_8^0 X^*(6) = 0 + 0 = 0$$
$$x_1(6) = X^*(2) - W_8^0 X^*(4) = 0 - 0 = 0$$
$$x_1(1) = X^*(1) + W_8^0 X^*(5) = -5.828 + j2.279 - 0.172 - j0.279 = -6 + j2$$
$$x_1(5) = X^*(1) - W_8^0 X^*(5) = -5.828 + j2.279 + 0.172 + j0.279 = -5.656 + j2.558$$
$$x_1(3) = X^*(3) + W_8^0 X^*(7) = -0.172 + j0.279 - 5.828 - j2.279 = -6 - j2$$
$$x_1(7) = X^*(3) - W_8^0 X^*(7) = -0.172 + j0.279 + 5.828 + j2.279 = 5.656 + j2.558$$

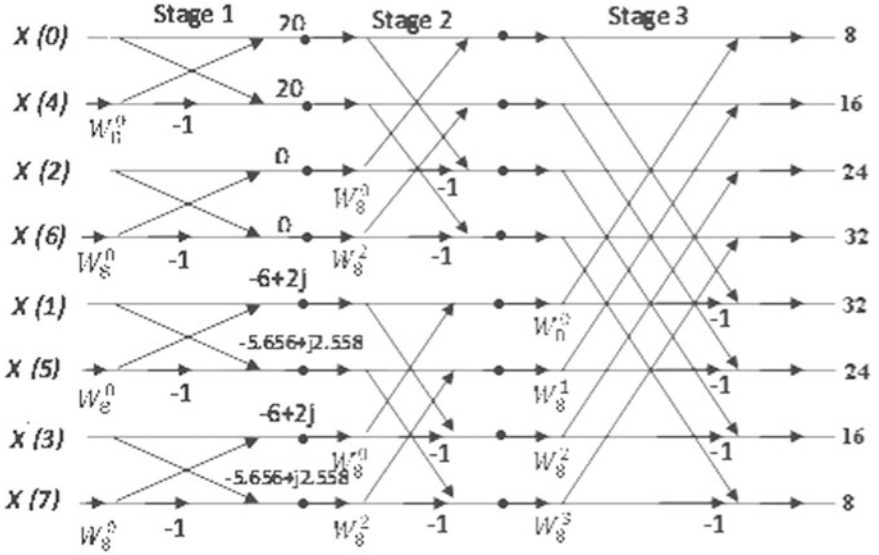


Fig. 4.20 Eight-point inverse DFT of Example 4.24 using the DIT algorithm

Stage 2

$$x_2(0) = x_1(0) + W_8^0 x_1(2) = 20 + 0 = 20;$$
$$x_2(4) = x_1(4) + W_8^2 x_1(6) = 20 + 0 = 20;$$
$$x_2(2) = x_1(0) - W_8^0 x_1(2) = 20;$$
$$x_2(6) = x_1(4) - W_8^2 x_1(6) = 20;$$
$$x_2(1) = x_1(1) + W_8^0 x_1(3) = -6 + 2j - 6 - 2j = -12;$$
$$x_2(5) = x_1(5) + W_8^2 x_1(7) = -5.656 + j2.558 + (-j)(5.656 + j2.558) = -3.098 - j3.098;$$
$$x_2(3) = x_1(1) - W_8^0 x_1(3) = -6 + 2j + 6 + 2j = 4j;$$
$$x_2(7) = x_1(5) - W_8^2 x_1(7) = -5.656 + j2.558 + j5.656 - 2.558 = -8.214 + j8.224;$$

Stage 3

$$x_3(0) = x_2(0) + W_8^0 x_2(1) = 20 - 12 = 8;$$
$$x_3(1) = x_2(4) + W_8^1 x_2(5) = 20 + (-3.098 - j3.098)(0.707 - j0.707) = 16.0006;$$
$$x_3(2) = x_2(2) + W_8^2 x_2(3) = 20 + (-j)(4j) = 24;$$
$$x_3(3) = x_2(6) + W_8^3 x_2(7) = 20 + (-0.707 - j0.707)(-8.214 + j8.214) = 31.9982;$$
$$x_3(4) = x_2(0) - W_8^0 x_2(1) = 20 + 12 = 32;$$
$$x_3(5) = x_2(4) - W_8^1 x_2(5) = 20 - (-3.098 - j3.098)(0.707 - j0.707) = 23.9994;$$
$$x_3(6) = x_2(2) - W_8^2 x_2(3) = 20 - (-j)(4j) = 16;$$
$$x_3(7) = x_2(6) - W_8^3 x_2(7) = 20 - (-0.707 - j0.707)(-8.214 + j8.214) = 8.0018;$$

Therefore,

$$8x^*(n) = \{8, 16, 24, 32, 32, 24, 16, 8\}$$

Hence,

$$x(n) = \{1, 2, 3, 4, 4, 3, 2, 1\}$$

4.12 Computation of DFT and IDFT Using MATLAB

The built-in MATLAB functions **fft(x)** and **ifft(x)** can be used for the computation of the DFT and the IDFT, respectively. The functions use computationally efficient FFT algorithms.

Example 4.26 Consider the input sequence $x(n) = \{1, 1, 1, 1, 0, 0, 1, 1\}$ of Example 4.5. Compute the DFT using MATLAB.

Solution Execution of fft(x) yields the DFT of $x(n)$ as

$$6.000 \quad 1.7071 - 0.7071i \quad -1.0000 + 1.0000i \quad 0.2929 - 0.7071i \quad 0$$
$$0.2929 + 0.7071i \quad -1.0000 - 1.0000i \quad 1.7071 + 0.7071i$$

which is equivalent to the DFT computed using the definition of DFT as in Example 4.3.

Example 4.27 Consider the input

$$X(k) = \{20, -5.828 - j2.279, 0, -0.172 - j0.279, 0, \\ -0.172 + j0.279, 0, -5.828 + j2.279\}$$

of Example 4.24. Compute IDFT using MATLAB.

Solution Execution of ifft(X) yields the IDFT of X as

1.0 2.0 3.0 4.0 4.0 3.0 2.0 1.0

which is the same as the result obtained in Example 4.24.

4.13 Application Examples

4.13.1 Detection of Signals Buried in Noise

One of the applications of the DFT-based spectral analysis is to detect the signals buried in noise. For example, consider a noisy signal with K sinusoidal components with unknown frequencies $f_1, f_2, \ldots, f_K$ given by

$$x(n) = \sum_{i=1}^{K} \frac{2\pi n f_i}{F_T} + \eta(n) \quad 0 \le n \le N \tag{4.98}$$

where $\eta(n)$ is additive white noise. The unknown frequencies $f_1, f_2, \ldots, f_K$ can be detected by using DFT. For simulation, a signal with two (K = 2) sinusoidal components N = 1024 and the sampling frequency F_T = 1000 Hz are assumed. The following MATLAB program is used to generate the noisy signal and to detect the unknown frequencies by applying the DFT on the generated noisy signal.

Program 4.1 Detection of signals buried in noise

```
clear;clc;
N = 1024;
K=2;
x =randn(1,N);% random noise generation
F_T =1000; % sampling frequency
T = 1/F_T; % sampling time period
k=1:N;
f=(F_T/2)*rand(1,K); %random generation of unknown frequencies
for i=1:K
  x=x+sin(2*pi*f(i)*k*T); % noisy signal with sinusoidal components
```

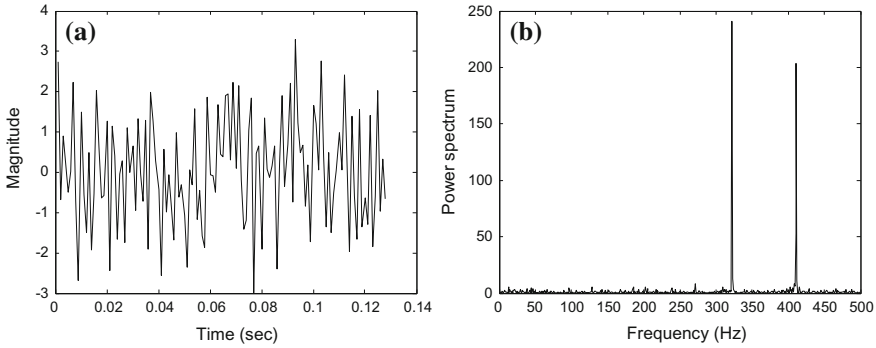


Fig. 4.21 **a** Noisy signal and **b** power spectrum density of the noisy signal

```
end
t = k*T;
figure(1),plot (t(1:N/8),x(1:N/8))
xlabel('Time(sec)');ylabel('Magnitude');
% Compute and plot power density spectrum
figure(2),
X= abs(fft(x));
S = X.^2/N;
f = linspace (0,(N-1)*Fs/N,N);
plot (f(1:N/2),S(1:N/2))
set(gca,'Xlim',[0,Fs/2])
xlabel('Frequency (Hz)');
ylabel('Power spectrum')
% Finding frequencies
s = f_prompt ('Enter threshold for locating peaks',0,max(S),.7*max(S));
for i = 1: N/2
  if (S(i)>s)
    fprintf ('f = %.0f Hz\n',f(i))
  end
end
```

For a random run of the above program, the noisy signal and its power spectral density are shown in Fig. 4.21a, b, respectively, and the two unknown frequencies are identified as $f_1 = 322$ Hz and $f_2 = 411$ Hz.

4.13.2 Denoising of a Speech Signal

The DFT can be applied to Fourier domain filtering which is equivalent to circular convolution of a sequence of finite length with an ideal impulse response of finite length. This approach is useful in denoising a signal for suppressing high-frequency

noise from a low-frequency signal corrupted with noise. For purpose of illustration, we considered the speech signal 'To take good care of yourself' from sound file 'goodcare.wav'. The following MATLAB program is used to read the speech signal from the wav file and to add noise to the speech signal and to reconstruct the original speech signal by performing circular convolution of the noisy speech signal with finite length impulse response.

%Program 4.2 Denoising using circular convolution

```
clear;clc;
[x,fs]=wavread('goodcare.wav');
wavplay(x,fs)% listen to original speech signal
no=0.075*randn(1,length(x));% noise generation
xn=x+no';%add noise to original speech signal
wavplay(xn,fs)%listen to noisy speech signal
figure(1),plot(x);xlabel('Number of samples');ylabel('Amplitude');
figure(2),plot(xn);xlabel('Number of samples');ylabel('Amplitude');
h=ones(1,64)/64;y=fftfilt(h,xn);%perform denoising
wavplay(12*y,fs);% listen to recovered speech signal
figure(3),plot(12*y);xlabel('Number of samples');ylabel('Amplitude');
```

The speech signal, the noisy speech signal, and the recovered speech signal after denoising, obtained from the above MATLAB program, are shown in Figs. 4.22a–c, respectively. From these figures, it can be observed that the recovered speech signal after denoising is nearly same as the original signal.

4.13.3 DTMF Tone Detection Using Goertzel Algorithm

Dual-tone multifrequency (DTMF) signaling is widely used worldwide for voice communications in modern telephony to dial numbers and configure switch boards. It is also used in voice mail, electronic mail, and telephone banking.

DTMF signaling uses two tones to represent each key on the touch pad. There are 12 distinct tones. When any key is pressed, the tone of the column and the tone of the row are generated. As an example, pressing the '5' button generates the tones 770 Hz and 1336 Hz. In this example, use the number 10 to represent the '*' key and 11 to represent the '#' key.

The frequencies were chosen to avoid harmonics: No frequency is a multiple of another, the difference between any two frequencies does not equal any of the frequencies, and the sum of any two frequencies does not equal any of the frequencies.

The industry standard frequency specifications for all the keys are listed in Fig. 4.23.

The DTMF signals for each button on telephone pad are shown in Fig. 4.24. The MATLAB program to generate the DTMF signals is listed in Program 4.3.

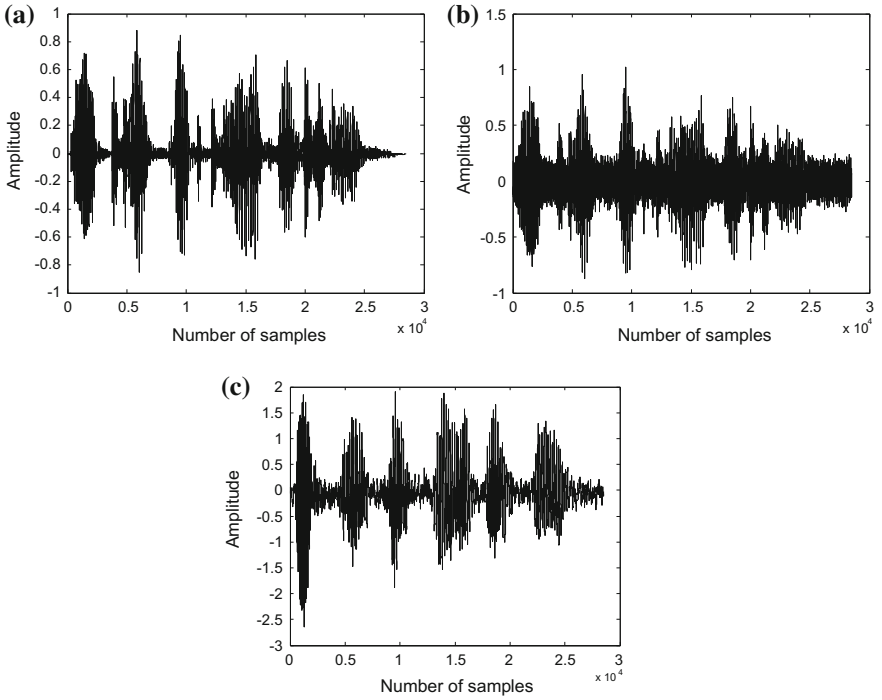


Fig. 4.22 **a** Speech signal, **b** noisy speech signal, and **c** recovered speech signal after denoising

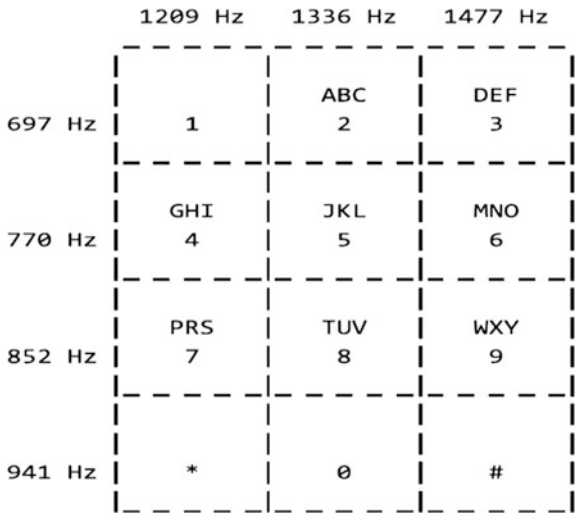


Fig. 4.23 DTMF tone specifications

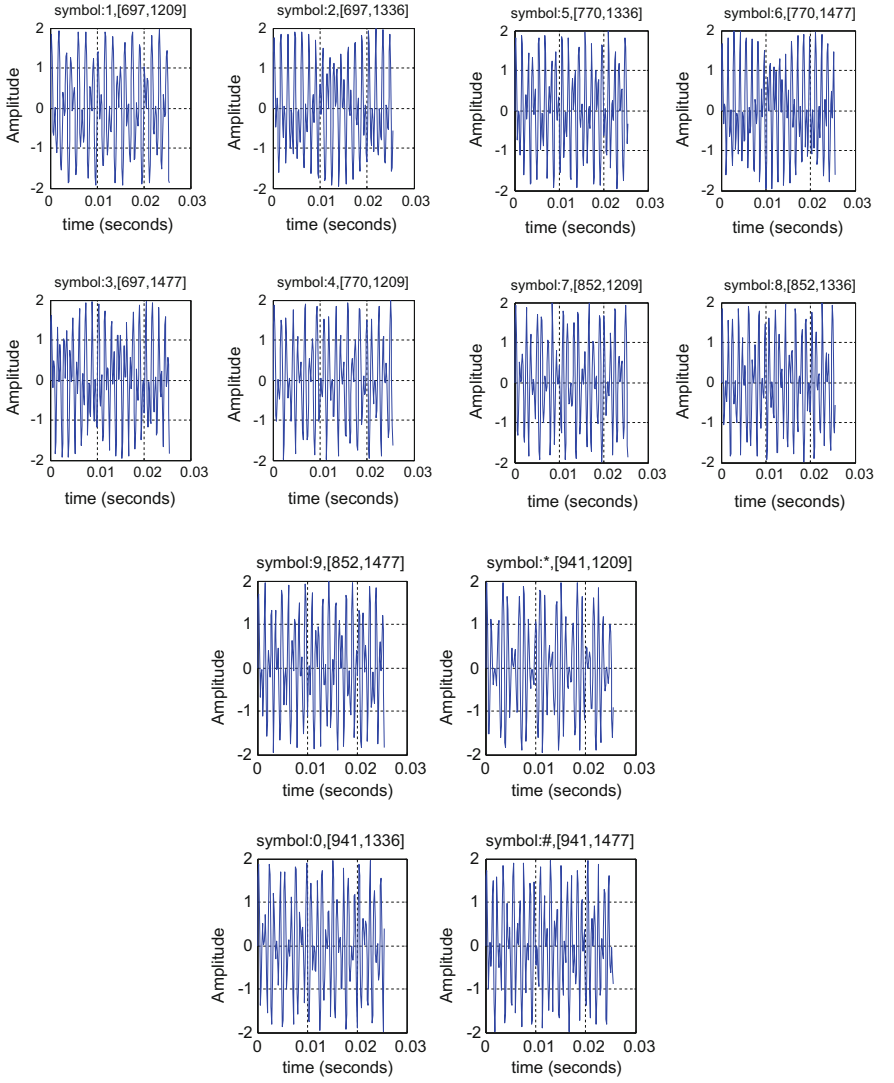


Fig. 4.24 Time responses of each tone of the telephone pad

Program 4.3

```
%MATLAB program DTMF tones generation
clear all;clc;
Fs = 8000;N = 205;t=[0:1:204]/Fs;
lf=[697;770;852;941];hf=[1209;1336;1477];
ylf1=sin(2*pi*lf(1)*(0:N-1)/Fs);ylf2=sin(2*pi*lf(2)*(0:N-1)/Fs);
ylf3=sin(2*pi*lf(3)*(0:N-1)/Fs);ylf4=sin(2*pi*lf(4)*(0:N-1)/Fs);
```

```
yhf1=sin(2*pi*hf(1)*(0:N-1)/Fs);yhf2=sin(2*pi*hf(2)*(0:N-1)/Fs);
yhf3=sin(2*pi*hf(3)*(0:N-1)/Fs);
y1=ylf1+yhf1;y2=ylf1+yhf2;y3=ylf1+yhf3;y4=ylf2+yhf1;
y5=ylf2+yhf2;y6=ylf2+yhf3;y7=ylf3+yhf1;y8=ylf3+yhf2;
y9=ylf3+yhf3;ystar=ylf4+yhf1;y0=ylf4+yhf2;yhash=ylf4+yhf3;
figure(1)
subplot(2,2,1);plot(t,y1);xlabel('time (seconds)')
ylabel('Amplitude');grid;title('symbol:1,[697,1209]');
subplot(2,2,2);plot(t,y2);xlabel('time (seconds)')
ylabel('Amplitude');grid;title('symbol:2,[697,1336]');
subplot(2,2,3);plot(t,y3);xlabel('time (seconds)')
ylabel('Amplitude');grid;title('symbol:3,[697,1477]');
subplot(2,2,4);plot(t,y4);
xlabel('time (seconds)')
ylabel('Amplitude');grid;title('symbol:4,[770,1209]');
figure(2)
subplot(2,2,1);plot(t,y5);xlabel('time (seconds)')
ylabel('Amplitude');grid;title('symbol:5,[770,1336]');
subplot(2,2,2);plot(t,y6);xlabel('time (seconds)')
ylabel('Amplitude');grid;title('symbol:6,[770,1477]');
subplot(2,2,3);plot(t,y7);xlabel('time (seconds)')
ylabel('Amplitude');grid;title('symbol:7,[852,1209]');
subplot(2,2,4);plot(t,y8);xlabel('time (seconds)')
ylabel('Amplitude');grid;title('symbol:8,[852,1336]');
figure(3)
subplot(2,2,1);plot(t,y9);xlabel('time (seconds)')
ylabel('Amplitude');grid;title('symbol:9,[852,1477]');
subplot(2,2,2);plot(t,ystar);xlabel('time (seconds)')
ylabel('Amplitude');grid;title('symbol:*,[941,1209]');
subplot(2,2,3);plot(t,y0);xlabel('time (seconds)')
ylabel('Amplitude');grid;title('symbol:0,[941,1336]');
subplot(2,2,4);plot(t,yhash);xlabel('time (seconds)')
ylabel('Amplitude');grid;title('symbol:#,[941,1477]');
```

DTMF tone detection

The DTMF detection relies on the Goertzel algorithm (Goertzel filter). The main purpose of using the Goertzel filters is to calculate the spectral value at the specified frequency index using the filtering method. Its advantage includes the reduction of the required computations and avoidance of complex algebra. The detection of frequencies using Goertzel algorithm contained in each tone of the telephone pad is shown in Fig. 4.25. The MATLAB program for the tones detection using the Goertzel algorithm is listed in Program 4.4.

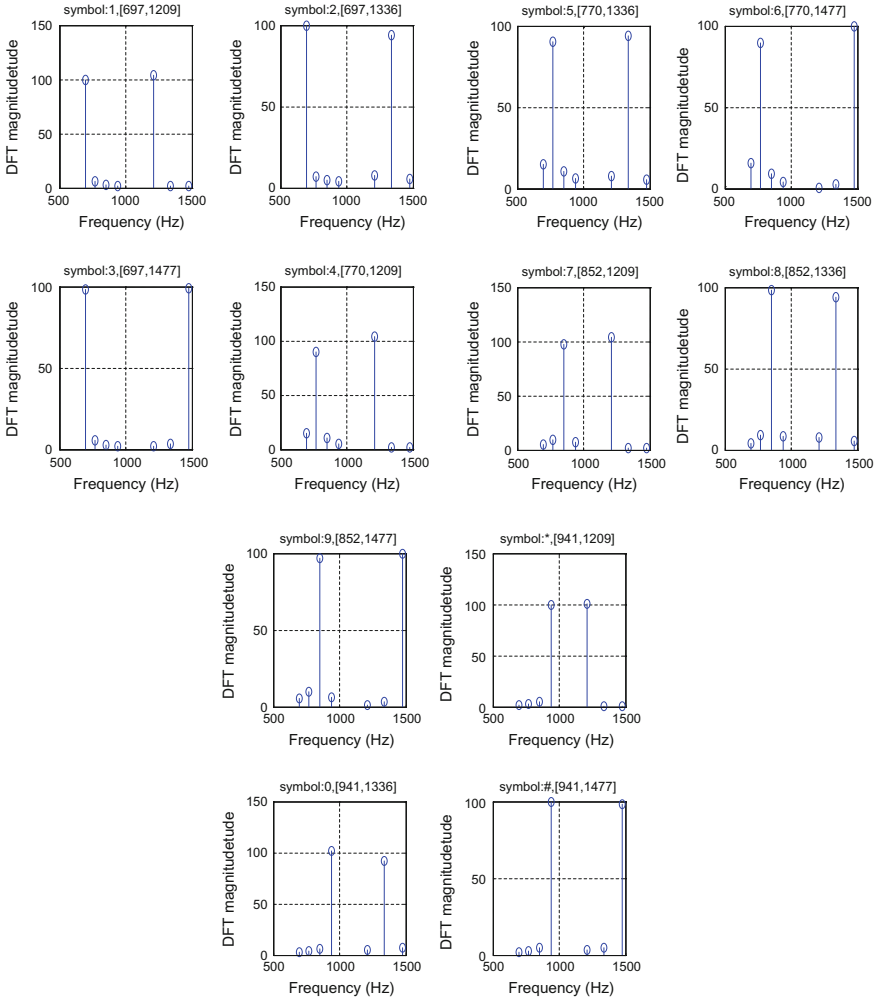


Fig. 4.25 DTMF tone detection using Goertzel algorithm

Program 4.4

```
clear all;clc;
Fs = 8000;N = 205;load DTMFdata
f = [697 770 852 941 1209 1336 1477];
freq_indices = round(f/Fs*N) + 1;
for tonechoice=1:12
tonedata=DTMFs(tonechoice,:);
dft_data(tonechoice,:) = goertzel(tonedata,freq_indices);
end
```

```
figure(1)
subplot(2,2,1);stem(f,abs(dft_data(1,:)));ax = gca;ax.XTick = f;
xlabel('Frequency(Hz')
ylabel('DFT magnitudetude');grid;title('symbol:1,[697,1209]');
subplot(2,2,2);stem(f,abs(dft_data(2,:)));ax = gca;ax.XTick = f;
xlabel('Frequency(Hz')
ylabel('DFT magnitudetude');grid;title('symbol:2,[697,1336]');
subplot(2,2,3);stem(f,abs(dft_data(3,:)));ax = gca;ax.XTick = f;
xlabel('Frequency(Hz')
ylabel('DFT magnitudetude');grid;title('symbol:3,[697,1477]');
subplot(2,2,4);stem(f,abs(dft_data(4,:)));ax = gca;ax.XTick = f;
xlabel('Frequency(Hz')
ylabel('DFT magnitudetude');grid;title('symbol:4,[770,1209]');
figure(2)
subplot(2,2,1);stem(f,abs(dft_data(5,:)));ax = gca;ax.XTick = f;
xlabel('Frequency(Hz')
ylabel('DFT magnitudetude');grid;title('symbol:5,[770,1336]');
subplot(2,2,2);stem(f,abs(dft_data(6,:)));ax = gca;ax.XTick = f;
xlabel('Frequency(Hz')
ylabel('DFT magnitudetude');grid;title('symbol:6,[770,1477]');
subplot(2,2,3);stem(f,abs(dft_data(7,:)));ax = gca;ax.XTick = f;
xlabel('Frequency(Hz')
ylabel('DFT magnitudetude');grid;title('symbol:7,[852,1209]');
subplot(2,2,4);stem(f,abs(dft_data(8,:)));ax = gca;ax.XTick = f;
xlabel('Frequency(Hz')
ylabel('DFT magnitudetude');grid;title('symbol:8,[852,1336]');
figure(3)
subplot(2,2,1);stem(f,abs(dft_data(9,:)));ax = gca;ax.XTick = f;
xlabel('Frequency(Hz')
ylabel('DFT magnitudetude');grid;title('symbol:9,[852,1477]');
subplot (2,2,2);stem(f,abs(dft_data(10,:)));ax = gca;ax.XTick = f;
xlabel('Frequency(Hz')
ylabel('DFT magnitudetude');grid;title('symbol:*,[941,1209]');
subplot (2,2,3);stem(f,abs(dft_data(11,:)));ax = gca;ax.XTick = f;
xlabel('Frequency(Hz')
ylabel('DFT magnitudetude');grid;title('symbol:0,[941,1336]');
subplot (2,2,4);stem(f,abs(dft_data(12,:)));ax = gca;ax.XTick = f;
xlabel('Frequency(Hz')
ylabel('DFT magnitudetude');grid;title('symbol:#,[941,1477]');
```

4.14 Problems

1. Determine the Fourier series representation for the following discrete-time signals:

   (a) $x(n) = 3\sin\left(\frac{\pi n}{4}\right)\sin\left(\frac{2\pi n}{5}\right)$
   (b) $x(n)$ is periodic of period 8, and $x(n) = n$ for $0 \le n \le 3$, and $x(n) = n$ for $4 \le n \le 7$

2. Compute the eight-point DFT of $(-1)^n$
3. Find the four-point DFT of the following sequences

   (i) $x(n) = \{1, 2, 1, 1\}$
   (ii) $x(n) = \sin(n+1)\pi/4$
   (iii) $x(n) = \{2, -1, 1, -2\}$.

4. Find eight-point DFT of the following sequences

   (i) $x(n) = \{1, 0, 1, 0, 0, 1, 1, 0\}$
   (ii) $x(n) = \cos(n+1)\pi/2$
   (iii) $x(n) = \{1, 1, 0, 0, 1, 0, 1, 1\}$

5. Compute the eight-point DFT of the square-wave sequence:

$$x(n) = \begin{cases} 2 & 0 \le n \le (N/2) \\ -2 & (N/2) \le n < N-1 \end{cases}$$

6. Find 16-point DFT of the following sequence:

$$x(n) = \begin{cases} 1 & 0 \le n \le 7 \\ 0 & 7 < n < 15 \end{cases}$$

7. Compute the eight-point circular convolution of

$$x_1(n) = \{1, 1, 0, 1, 0, 1, 1, 0\} \text{ and } x_2(n) = \sin(3\pi/4), 0 \le n \le 7.$$

8. Find the output $y(n)$ of a filter whose impulse response is $h(n) = \{0, 1, 1\}$ and the input signal is $x(n) = \{1, -2, 0, 1, 0, 2, 1, 2, 2, 1\}$ using the overlap-add method.
9. Using linear convolution, find $y(n) = x(n) * h(n)$ for the sequences $x(n) = (2, -3, 1, 2, 1, 1, -1, -3, 1, 2, 1, -1)$ and $h(n) = (2, 1)$. Compare the result by solving the problem using overlap-save method.
10. Compute the eight-point DFT of the following sequence using the radix-2 DIT algorithm for the following sequences:

(i) $x(n) = \{1, 1, -1, 0, 1, 0, 1, -1\}$
(ii) $x(n) = \{1, 2, 1, -1, 2, 1, -1, 1\}$
(iii) $x(n) = \{0.5, 0, 1, 0.5, 1, 0, 0.5, 0.5\}$

11. Compute the eight-point DFT of the sequence $x(n) = \{1, 1, -1, 0, 1, 0, 1, -1\}$ using the DIF algorithm
12. Find the 16-point DFT of the following sequence using radix-4 DIF algorithm.

$$x(n) = \{1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1\}$$

13. Compute DFT of the sequence $x(n) = \{1, 2, 3, 4\}$ using the Goertzel algorithm
14. Develop the FFT algorithm for the composite number 18, and show the flow graph.
15. Find the IDFT of $Y(k) = \{1, 0, 0, 1\}$.
16. Compute the IDFT of the sequence $X(k) = \{3, j, 1 + 2j, 1 - j, 1 + 2j, 1, 0, -j\}$ using (a) DIT algorithm and (b) DIF algorithm.

4.15 MATLAB Exercises

1. Verify the results of Problem 10 of Sect. 4.13 using MATLAB.
2. Verify the results of Problem 14 of Sect. 4.13 using MATLAB.
3. Write a MATLAB program using the command circshift to compute circular convolution of two sequences and verify the result of Problem 7 of Sect. 4.13.
4. Verify the results of Problem 8 of Sect. 4.13 using MATLAB.

References

1. J.W. Cooley, P.A.W. Lewis, P.D. Welch, Historical notes on the fast Fourier transform. IEEE Trans. Audio Electroacoust. **55**(10), 1675–1677 (1967)
2. J.W. Cooley, P.A.W. Lewis, P.D. Welch, Historical notes on the fast Fourier transform. Proc. IEEE **55**(10), 1675–1677 (1967)
3. G. Goertzel, An algorithm for the evaluation of finite trignometric series. Am. Math Monthly **65**, 34–35 (1958)
4. A.V. Oppenheim, R.W. Schafer, J.R. Buck, *Discrete-Time Signal Processing*, 2nd edn. (Prentice Hall, 1999)
5. L. Rabiner, R. Schafer, C. Rader, IEEE Trans. Audio Electroacoust. **17**(2) (1969)