# Chapter 4 - THE DISCRETE FOURIER TRANSFORM

## Introduction

The Fourier representation of signals that we studied in Chapter 3 is important for understanding how filters work and what a spectrum is, but it is not a practical tool because the DTFT is a continuous function of frequency and therefore its computation would in general require an infinite number of operations. The purpose of this chapter is to introduce another representation of discrete-time signals, the *discrete Fourier transform (DFT),* which is closely related to the discrete-time Fourier transform, and can be implemented either in digital hardware or in software. The DFT is of great importance as an efficient method for computing the discrete-time convolution of two signals, as a tool for filter design, and for measuring spectra of discrete-time signals. While computing the DFT of a signal is generally easy (requiring no more than the execution of a simple program) the *interpretation* of these computations can be difficult because the DFT only provides a complete representation of *finite-duration* signals.

## 4.1 Definition of the discrete Fourier transform

### 4.1.1 Sampling the Fourier transform

It is not in general possible to compute the discrete-time Fourier transform of a signal because this would require an infinite number of operations. However, it is always possible to compute a finite number of *frequency samples* of the DTFT in the hope that, if the spacing between samples is sufficiently small, this will provide a good representation of the spectrum. Simple results are obtained by sampling in frequency at regular intervals. We therefore define the *$N$-point discrete Fourier transform* $X[k]$ of a signal $x[n]$ as samples of its transform $X(f)$ taken at intervals of $1/N$:

$$X[k] \overset{\triangle}{=} X(k/N) = \sum_{n=-\infty}^{\infty} x[n] e^{-j2\pi kn/N} \qquad \text{for } 0 \leq k \leq N-1 \qquad (4.1)$$

Because $X(f)$ is periodic with period 1, $X[k]$ is periodic with period $N$, which justifies only considering the values of $X[k]$ over the interval $[0,\ N-1]$.

### 4.1.2 Condition for signal reconstruction from the DFT

An important question is whether the DFT provides a complete representation of the signal, that is, if the signal can be reconstituted from its DFT. From what we know about sampling, we expect that this will only be possible under certain conditions. Specifically, we have seen in Chapter 1 that, if we take $N$ samples *per period* of a continuous-time signal with period $T$, then the signal can be exactly reconstructed provided that $N > 2\ WT$, where $W$ is the largest frequency component in the signal. Similarly, if we take $N$ samples per period of the continuous-frequency, periodic signal $X(f)$ with period 1, we expect that the spectrum can be reconstructed if $N$ is greater than the duration of the time signal $x[n]$. The sampling operation is equivalent to a multiplication by a train of impulses (Fig. 4.1), which is covered in more detail in Chapter 5. Therefore, sampling at intervals of $1/N$ effectively forms the new spectrum

$$\tilde{X}(f) \overset{\triangle}{=} X(f) \sum_{k=-\infty}^{\infty} \delta(f - k/N) = \sum_{k=-\infty}^{\infty} X(k/N)\delta(f - k/N) = \sum_{k=-\infty}^{\infty} X[k]\delta(f - k/N) \qquad (4.2)$$

Remembering from (3.12m) that

$$N \sum_{r=-\infty}^{\infty} \delta[n - rN] \longleftrightarrow \sum_{k=-\infty}^{\infty} \delta(f - k/N)$$

and applying the convolution theorem shows that the inverse DTFT $\tilde{x}[n]$ of the sampled spectrum $\tilde{X}(f)$ is the convolution of the original signal $x[n]$ by a periodic train of unit samples:

$$\tilde{x}[n] = x[n] * N \sum_{r=-\infty}^{\infty} \delta[n - rN] = N \sum_{r=-\infty}^{\infty} x[n - rN] \qquad (4.3)$$

The relation between $x[n]$ and $\tilde{x}[n]$ is shown in Fig. 4.1. The signal $\tilde{x}[n]$ is periodic with period $N$. It is said to be a *time-aliased* version of $x[n]$ by analogy with the frequency-aliasing formula (1.30). In the important special case when the duration of $x[n]$ is smaller than $N$, specifically, if $x[n]$ is zero outside of the interval $[0, N - 1]$, one has:

$$\tilde{x}[n] = Nx[n] \qquad \text{for } 0 \le n \le N - 1 \qquad (4.4)$$

Only in this special case can the signal be exactly reconstructed from its DFT. Such reconstruction can be accomplished by multiplying with a rectangular window in time, which corresponds to convolving in frequency with the interpolation function illustrated in Figure 4.2.

### 4.1.3 Inverse discrete Fourier transform

So far, we have proven that the finite-duration signal $x[n]$ can in principle be reconstructed from its DFT $X[k]$, but we have not given an explicit formula for achieving this reconstruction. We will show by two different methods that the desired inverse DFT formula is:

$$x[n] \ = \ \frac{1}{N} \sum_{k=0}^{N-1} X[k]\ e^{j2\pi kn/N} \qquad (4.5a)$$

A first method for deriving this formula is to combine (4.4) with the definition of $\tilde{X}(f)$ in (4.2):

$$\tilde{X}(f) \ = \ \sum_{k=-\infty}^{\infty} X[k]\ \delta(f - k/N)$$

Taking the inverse DTFT, one obtains:

$$x[n] \;=\; \frac{1}{N}\,\tilde{x}[n] \;=\; \frac{1}{N}\int_0^1 \tilde{X}(f)\; e^{j2\pi fn}\; df \;=\; \frac{1}{N}\sum_{k=-\infty}^{\infty} X[k]\int_0^1 \delta(f-k/N)\; e^{j2\pi fn}\; df$$

where we have interchanged the orders of summation and integration to obtain the rightmost expression[1]. We note that

$$\int_0^1 \delta(f-k/N)e^{j2\pi fn}df = \begin{cases} e^{j2\pi kn/N} & \text{if } 0 \le k \le N-1 \\ 0 & \text{otherwise} \end{cases}$$

because $k/N$ is outside of the range of integration $[0,1[$ when $k$ is outside of the interval $[0, N-1]$. Therefore the inverse DFT formula is

$$x[n] \;=\; \frac{1}{N}\sum_{k=0}^{N-1} X[k]\; e^{j2\pi kn/N} \tag{4.5$a$}$$

Because the signal $x[n]$ is of finite duration, the definition of the DFT (4.1) becomes:

$$X[k] \;=\; \sum_{n=0}^{N-1} x[n]\; e^{-j2\pi kn/N} \tag{4.5$b$}$$

Formulas (4.5a) and (4.5b) constitute the DFT pair for finite-duration signals. Note the symmetry between (4.5a) and (4.5b), the only differences being the signs of the arguments of the complex exponentials and the $1/N$ factor in the inverse DFT formula (4.5a).

The DFT pair (4.5) can also be considered as a purely algebraic relation between the $N$ numbers $x[n], 0 \le n \le N-1$ and the $N$ numbers $X[k], 0 \le k \le N-1$, with the two sets of numbers being related by a set of $N$ linear equations. This point of view leads to an alternate proof of the inversion formula (4.5a). Specifically, assume that the $X[k]$ are defined from the $x[n]$ by (4.5b), and form the sum

$$y[m] \overset{\triangle}{=\!=} \sum_{k=0}^{N-1} X[k]e^{j2\pi km/N} = \sum_{k=0}^{N-1}\sum_{n=0}^{N-1} x[n]e^{j2\pi k(m-n)/N}$$

Interchanging the order of summations produces

$$y[m] \;=\; \sum_{n=0}^{N-1} x[n] \sum_{k=0}^{N-1} e^{j2\pi k(m-n)/N}$$

Using the result that discrete complex exponentials with period $N$ are orthogonal over the interval $[0,\; N-1]$:

$$\sum_{k=0}^{N-1} e^{j2\pi k(m-n)/N} = \begin{cases} N & \text{if } m-n \text{ is a multiple of } N \\ 0 & \text{otherwise} \end{cases}$$

we obtain

$$y[m] \;=\; \sum_{k=0}^{N-1} X[k]\; e^{j2\pi km/N} \;=\; N\; x[m]$$

[1]Although the range of integration $[0,\;1[$ differs from the usual range $[-\frac{1}{2},\;\frac{1}{2}[$ for inverse DTFTs, the two are equivalent because Fourier transforms are periodic.

which proves the inverse DFT formula (4.5a). This proof emphasizes that the DFT operation can be considered as a change of basis set (specifically a rotation) in an $N$-dimensional vector space. In the time domain, the signal is decomposed into a sum of orthogonal unit samples $\delta[n-k]$, while in the inverse DFT formula, the orthogonal basis vectors are the complex exponentials $e^{j2\pi kn/N}$.

### 4.1.4 Relation to discrete Fourier series

We have shown that taking $N$ samples of the DTFT $X(f)$ of a signal $x[n]$ is equivalent to forming a periodic signal $\tilde{x}[n]$ which is derived from $x[n]$ by time aliasing. If the duration of $x[n]$ is smaller than $N$, one period of $\tilde{x}[n]$ is identical to $x[n]$ within a factor of $N$. These results are the dual of those obtained in Section 1.3 for the sampling of periodic, continuous time signals. We showed that taking $N$ samples per period of the periodic signal $x(t)$ results in frequency aliasing of the Fourier series coefficients $X_k$. If the bandwidth of $x(t)$ is less than $N/2T$, the Fourier series coefficients of the discrete-time signal coincide with those of the original signal $x(t)$. Thus, there is a duality between sampling in frequency the DTFT of a discrete-time signal to form the DFT, and sampling in time a periodic signal to form the discrete Fourier series. In both cases, sampling produces signals that are discrete and periodic in both the time and the frequency domain. Therefore, the discrete Fourier transform and the discrete Fourier series are the same mathematical operation (within a factor of $N$).

This result is to be expected because there is a one-to-one correspondence between discrete time signals of duration $N$ and discrete, periodic signals with period $N$. Specifically, given a finite-duration signal $x[n]$, we can always generate a periodic signal $\tilde{x}_N[n]$ by repeating $x[n]$ indefinitely at intervals of $N$ samples:

$$\tilde{x}_N[n] \overset{\triangle}{=} \sum_{r=-\infty}^{\infty} x[n+rN] = x[n] * \sum_{r=-\infty}^{\infty} \delta[n-rN] \tag{4.6$a$}$$

Conversely, given a periodic signal $\tilde{x}_N[n]$, we can form a finite-duration signal $x[n]$ by multiplication with a rectangular pulse of length $N$:

$$x[n] = \tilde{x}_N[n] R_N[n] = \begin{cases} \tilde{x}_N[n] & \text{if } 0 \le n \le N-1 \\ 0 & \text{otherwise} \end{cases} \tag{4.6$b$}$$

The Fourier series coefficients $\tilde{X}_k$ of the periodic signal are related to the DFT of the finite-duration signal by the formula:

$$\tilde{X}_k = \frac{1}{N} X[k] \tag{4.7}$$

To summarize, computing the $N$-point DFT of a signal implicitly introduces a periodic signal with period $N$, so that all operations involving the DFT are really operations on periodic signals. These operations will give the same results as operations on finite-duration signals *providing that the durations of all the signals involved in these operations are less than $N$.*

### 4.1.5 Application to filter design

The DFT can be used to design filters that approximate arbitrary specifications. Specifically, suppose that the frequency response of the filter to be approximated is $H(f)$. The *frequency sampling* method of filter design consists in sampling $H(f)$ at intervals of $1/N$, then taking the inverse DFT, yielding an FIR filter of length $N$. As shown by Equation (4.3), the unit-sample response $h_N[n]$ of the FIR filter will be a time-aliased version of the desired unit-sample response $h[n]$:

$$h_N[n] \;=\; \left[\, h[n] * \sum_{r=-\infty}^{\infty} \delta[n-rN] \,\right] R_N[n] \;=\; \left[\, \sum_{r=-\infty}^{\infty} h[n+rN] \,\right] R_N[n]$$

If the desired frequency response is smooth enough that $h[n]$ decays to a negligible value for $n \geq N$, the frequency response $H_N(f)$ of this FIR filter will provide a good approximation to $H(f)$. On the other hand, if $H(f)$ has abrupt discontinuities, the unit-sample response $h[n]$ will decay very slowly, and $H_N(f)$ will always show ripples regardless of the value of $N$ (Gibbs' phenomenon). Such ripples are apparent in Figure 4.3, which shows the frequency response of two lowpass filters designed by frequency sampling with $N = 33$.

To be more specific, $H_N(f)$ is guaranteed to be exactly equal to $H(f)$ for the $N$ frequency samples $f_k = k/N$. In between the samples, $H_N(f)$ can deviate appreciably from $H(f)$, particularly if the latter shows abrupt discontinuities. An explicit formula for $H_N(f)$ can be derived by Fourier transforming the above expression for $h_N[n]$. Making use of the product and convolution theorems, and noting that

$$\frac{1}{N}\, R_N[n] \;\longleftrightarrow\; \Psi_N(f) \overset{\triangle}{=} \frac{1}{N}\, \frac{1 \,-\, e^{-j2\pi fN}}{1 \,-\, e^{-j2\pi f}} \;=\; \frac{\sin\, \pi fN}{N\, \sin\, \pi f}\, e^{-j\pi f(N-1)} \qquad (4.8a)$$

shows that $H_N(f)$ is the cyclic convolution of $\Psi_N(f)$ with a sampled version of $H(f)$:

$$H_N(f) = \left[ H(f) \sum_{k=-\infty}^{\infty} \delta(f-k/N) \right] \circledast \Psi_N(f) = \left[ \sum_{k=-\infty}^{\infty} H[k]\delta(f-k/N) \right] \circledast \Psi_N(f)$$

$$H_N(f) \;=\; \sum_{k=0}^{N-1} H[k]\; \Psi_N(f-k/N) \qquad (4.8b)$$

This gives an interpolation formula for $H_N(f)$ as a function of the frequency samples $H[k]$. Except for a phase delay, the interpolating function $\Psi_N(f)$ is the same as the periodic interpolating function $\Phi_N(t)$ defined in Chapter 1. This result is another example of the duality between Fourier series and DTFTs. As shown in Fig. 4.2, $\Psi_N(f)$ has $N/2 - 1$ side lobes in addition to the main lobe centered at $f = 0$. These side lobes will result in ripple in $H_N(f)$ when there are abrupt discontinuities between the frequency samples $H[k]$.

To summarize, the frequency sampling method of design is not a good method for designing filters with abrupt discontinuities. Despite this limitation, this method is very useful in practice because it can be used for arbitrary filter specifications (magnitude and phase), and because it does not require that the desired unit-sample response be available in closed form.

## 4.2 Properties of the discrete Fourier transform

Most properties of the discrete Fourier transform are easily derived from those of the discrete-time Fourier transform by making the substitution of variables $f = k/N$. However, some care is needed because, as we have just seen, the DFT is inherently an operation on periodic signals.

### 4.2.1 Linearity

The linearity of the DFT directly follows from that of the DTFT.

### 4.2.2 Symmetry

In deriving symmetry properties for the DFT, it is important to remember that both $x[n]$ and $X[k]$ must be zero outside of the interval $[0, N-1]$. As before, we use the following notation for a signal and its DFT:

$$x[n] \longleftrightarrow X[k]$$

With these conventions, the major symmetry properties for real signals are:

$$x[N-n] \longleftrightarrow X^*[k] = X[N-k] \tag{4.9$a$}$$

$$x_{e,N}[n] \triangleq \frac{x[n]+x[N-n]}{2} \longleftrightarrow X_R[k] \triangleq \frac{X[k]+X^*[k]}{2} = X_R[N-k] \tag{4.9$b$}$$

$$x_{o,N}[n] \triangleq \frac{x[n]-x[N-n]}{2} \longleftrightarrow jX_I[k] \triangleq \frac{X[k]-X^*[k]}{2} = -jX_I[N-k] \tag{4.9$c$}$$

### 4.2.3 Cyclic convolution modulo $N$

One property which requires some discussion is the convolution theorem. We know that the DTFT of the (linear) convolution of two signals $x[n] * h[n]$ is the product of their transforms $X(f)\ H(f)$. If a similar theorem could be demonstrated for the DFT, it would be of great practical importance because one could reduce filtering operations to simple multiplications of the DFTs. We will show that this is indeed possible under certain conditions. For this purpose, we need to introduce the *cyclic convolution modulo $N$* of two periodic signals $\tilde{x}_N[n]$ and $\tilde{h}_N[n]$:

$$\tilde{x}_N[n] \circledast_N \tilde{h}_N[n] \triangleq \sum_{m=0}^{N-1} \tilde{x}_N[m]\ \tilde{h}_N[n-m] \tag{4.10$a$}$$

The cyclic convolution of two periodic signals is shown in Fig. 4.4.

With this definition, the convolution theorem for the DFT becomes

$$\left[\ \tilde{x}_N[n] \circledast_N \tilde{h}_N[n]\ \right]\ R_N[n] \longleftrightarrow X[k]\ H[k] \tag{4.10$b$}$$

where the periodic signals $\tilde{x}_N[n]$ and $\tilde{h}_N[n]$ are formed from $x[n]$ and $h[n]$ as in (4.5a).

Cyclic convolution differs from the linear convolution $x[n] * h[n]$ in that the result of a cyclic convolution *modulo* $N$ is periodic with period $N$, whereas the linear convolution of two $N$-point signals is a finite-duration signal with length $2N-1$. A $2N-1$ point signal cannot be represented by its $N$ point DFT without introducing time aliasing. Therefore, in general, the inverse DFT of $X[k]\ H[k]$ is a time-aliased version of the linear convolution $x[n] * h[n]$. Only when the duration of $x[n] * h[n]$ is smaller than the DFT length $N$ do linear and cyclic convolution coincide.

To prove the cyclic convolution theorem (4.10), we form the product

$$X[k]\ H[k] \ = \ \sum_{m=0}^{N-1} x[m]\ e^{-j2\pi km/N} \sum_{l=0}^{N-1} h[l]\ e^{-j2\pi kl/N} \ = \ \sum_{m=0}^{N-1} \sum_{l=0}^{N-1} \tilde{x}_N[m]\ \tilde{h}_N[l]\ e^{-j2\pi k(l+m)/N}$$

The function that is being summed is periodic in both variables $l$ and $m$. Therefore, as Figure 4.5 shows, summing over the square array $0 \le m, l \le N-1$ is the same as summing over the parallelogram array defined by $0 \le m \le N-1$ and $0 \le l+m \le N-1$. Making the change of variable $n \triangleq l+m$, the product of the DFTs becomes

$$X[k]\ H[k] \ = \ \sum_{n=0}^{N-1} \left[ \sum_{m=0}^{N-1} \tilde{x}_N[m]\ \tilde{h}_N[n-m] \right] e^{-j2\pi kn/N}$$

in which we recognize the DFT of the cyclic convolution $\tilde{x}_N[n] \circledast_N \tilde{h}_N[n]$. This completes the proof of (4.10).

By symmetry with (4.10), it is clear that the DFT of a product of signals is the cyclic convolution of their DFTs (within a factor of $N$)

$$x[n]\ w[n] \ \longleftrightarrow \ \frac{1}{N} \left[ \tilde{X}_N[k] \circledast_N \tilde{W}_N[k] \right] R_N[k] \tag{4.11}$$

## 4.2.4 Parseval's theorem for the DFT

An application of the cyclic convolution theorem is a form of Parseval formula for the DFT. This is obtained by writing the cyclic convolution theorem for $x[n]$ and $x[-n]$:

$$[\ \tilde{x}_N[n] \circledast_N \tilde{x}_N[-n]\ ]\ R_N[n] \ \longleftrightarrow \ X[k]\ X[-k] \ = \ |X[k]|^2$$

Writing this formula for time zero gives Parseval's theorem:

$$\sum_{n=0}^{N-1} x[n]^2 \ = \ \frac{1}{N} \sum_{k=0}^{N-1} |X[k]|^2 \tag{4.12}$$

This formula can be interpreted as a conservation of norm in a vector space.

## 4.2.5 Cyclic convolution example

The difference between linear and cyclic convolutions is best illustrated by an example. Consider the rectangular signal of length $M$:

$$x[n] \triangleq \begin{cases} 1 & \text{if } -\frac{M-1}{2} \le n \le \frac{M-1}{2} \\ 0 & \text{otherwise} \end{cases}$$

The linear convolution of $x[n]$ with itself is a triangle of length $2M-1$:

$$x[n]*x[n] \;=\begin{cases} M+n & \text{if } \; -(M-1)\le n\le 0 \\ M-n & \text{if } \; 0\le n\le M-1 \\ 0 & \text{otherwise} \end{cases}$$

The signals $x[n]$ and $x[n]*x[n]$ are shown in Fig. 4.6a for $M=7$. Their DTFTs are shown in Fig. 4.6e.

We will study the effect of the length of the period used on the results obtained by cyclic convolution. We form the signal $\tilde{x}_N[n]$ according to (4.6a) and then perform cyclic convolution modulo $N$ of $\tilde{x}_N[n]$ with itself according to (4.10a). The result of the cyclic convolution is also periodic with period $N$, and one period of this signal matches the time domain half of the DFT convolution theorem (4.10b). In the frequency domain, this corresponds to taking the $N$-point DFT of one period of $\tilde{x}_N[n]$ and squaring.

Figure 4.6b shows the signals $\tilde{x}_N[n]$ and $\tilde{x}_N[n]\circledast_N\tilde{x}_N[n]$ for $N=M=7$, and Fig. 4.6f shows their DFTs. The periodic signal $\tilde{x}_N[n]$ is a constant signal with amplitude 1. Therefore, the cyclic convolution modulo $N$ of $\tilde{x}_N[n]$ with itself is also a constant signal with amplitude $N$. This can be interpreted as a completely time aliased version of the triangle seen in Fig. 4.6a. The darkened points in the figure represent one period of each signal on the interval $[0,\ N-1]$, corresponding to the interval used to determine the DFT. Time aliasing is also evident in the frequency domain, where the sampling in frequency is too coarse to give an adequate representation of the underlying DTFT.

Figure 4.6c shows the signals $\tilde{x}_N[n]$ and $\tilde{x}_N[n]\circledast_N\tilde{x}_N[n]$ for $M=7, N=10$, and Fig. 4.6g shows their DFTs. In this case, partial time aliasing occurs. Again, the darkened points in the figure represent one period of each signal on the interval $[0, N-1]$, corresponding to the interval used to determine the DFT. In the frequency domain, the sampling in frequency is less coarse than in Fig. 4.6f.

Finally, Fig. 4.6d shows the signals $\tilde{x}_N[n]$ and $\tilde{x}_N[n]\circledast_N\tilde{x}_N[n]$ for $M=7, N=2M-1=13$, and Fig. 4.6h shows their DFTs. The result of the cyclic convolution modulo $N$ is a sawtooth function with period $N$, corresponding to an unaliased version of the triangle resulting from linear convolution in Fig. 4.6a. Restricting this result to the range $[-(M-1),\ M-1]$ produces a signal identical to the linear convolution of $x[n]$ with itself. The representation of the spectrum by its samples in Fig. 4.6h is much clearer than in Fig. 4.6f. We have seen that linear convolution is the same of one period of cyclic convolution modulo $N$ only when $N$ is at least equal to the duration of the linear convolution.

### 4.2.6 Convolution of two finite-duration signals using the DFT

This result is quite general: if $x[n]$ and $h[n]$ are both of finite duration, it is always possible to compute the DFTs with a sufficient number of points to ensure that the cyclic convolution gives the same result as the linear convolution. Specifically, if $L$ is the length of $x[n]$, and $M$ is the length of $h[n]$, the length of the linear convolution is $M+L-1$. Therefore, it suffices to compute DFTs of length $N\ \ge\ M+L-1$ in order to make sure that the convolution will not be time-aliased. In this case, the following scheme for filtering the input $x[n]$ by the filter $h[n]$ becomes possible:

1. Compute the $N$-point DFT of $x[n]$
2. Compute the $N$-point DFT of $h[n]$
3. Form the product $Y[k] = X[k]\ H[k]$
4. Compute the inverse N-point DFT of $Y[k]$.

The interest of this method is that fast DFT algorithms allow this sequence of operations to be carried out in less time than a direct (time-domain) implementation of the convolution sum.

## 4.3 Frequency-domain implementation of FIR filters

The preceding results show that linear convolutions can be realized by means of the DFT provided that the two signals to be convolved have finite durations. In many applications, however, one needs to convolve a relatively short unit sample response (say a few tens to a few hundred points) with a signal of *indefinite* duration. In such cases, the method outlined above may no longer be applicable because the signal $x[n]$ might be too long to fit in the computer memory. Even if sufficient memory were available, computational times might be too long, and inaccuracies resulting from finite-precision arithmetic might be too great to make the DFT method useful. In real time applications, having to wait until the signal has ended before the filtering operation can be started would be clearly unacceptable. Fortunately, it is possible to modify the DFT method of convolution to make it applicable to signals of indefinite duration. The general idea is to first segment the signal $x[n]$ into chunks of manageable size, then compute the convolution of each segment with the unit-sample response $h[n]$ by the DFT method, and finally join the results of the convolutions for all segments. Some care is needed in joining the convolved segments, however, because the cyclic convolution of a signal segment with the unit-sample response will coincide with the linear convolution only for part of the segment. We describe here one particular implementation called the *overlap save method.* A slightly different method called *overlap add* is described in *Oppenheim and Schafer.*

### 4.3.1 The overlap-save method for convolution

The basis for the overlap-save method is that, if the unit-sample response $h[n]$ of the FIR filter is non-zero over the interval $[0,\ M-1]$, and if we use an $N$-point DFT to convolve $h[n]$ with a signal segment $x_k[n]$ of length $N$, then only the last $N-M+1$ points of the cyclic convolution coincide with the linear convolution. This is because, for $0 \leq n < M-1$, the result of the cyclic convolution is a weighted sum of samples from both the beginning and the end of the segment $x_k[n]$. The overlap-save method of convolution is shown in Fig. 4.7 and is described below.

1. Divide the signal $x[n]$ into *overlapping* segments $x_k[n]$, each of length $N$, with an overlap

of $M-1$ points between segments:

$$x_k[n] \triangleq \begin{cases} x[n + k\,(N-M+1)] & \text{if } 0 \le n \le N-1 \\ 0 & \text{otherwise} \end{cases}$$

2. Form the cyclic convolution *modulo* $N$

$$z_k[n] \triangleq x_k[n] \circledast_N h[n]$$

by multiplying the $N$-point DFTs of $x_k[n]$ and $h[n]$ and taking the inverse DFT of the result. The resulting signal has length $N$.

3. Form a new sequence $y_k[n]$ of length $N-M+1$ by discarding the first $M-1$ points of $z_k[n]$:

$$y_k[n] \triangleq \begin{cases} z_k[n] & \text{if } M-1 \le n \le N-1 \\ 0 & \text{otherwise} \end{cases}$$

4. Form the final result $y[n]$ by joining the $y_k[n]$ with no overlap:

$$y[n] = \sum_{k=0}^{\infty} y_k[n - k\,(N-M+1)]$$

## 4.4 Fast Fourier transforms

The term *fast Fourier transform (FFT)* refers to a family of efficient algorithms for implementing the discrete Fourier transform. While computation of an $N$-point DFT by the straightforward method implied by the definition (4.5) requires $N^2$ complex multiplications, FFT methods require only of the order of $N \log_2 N$ complex multiplications. The savings in computation are considerable when $N$ is large: for example, for $N = 4096$, an FFT requires 300 times fewer operations than a straightforward DFT. Perhaps more than any other factor, it is the invention of the FFT by Cooley and Tukey in the 1960's that has made possible many of the applications of digital signal processing.

We will only discuss one particular method called the *decimation-in-time algorithm* among the family of FFT algorithms. Alternative algorithms are described in *Oppenheim and Schafer.*

### 4.4.1 Decimation-in-time algorithm

The principle of decimation in time, which is illustrated in Fig. 4.8, consists in dividing an $N$-point DFT (where $N$ is a power of two) into a weighted sum of progressively shorter DFTs. Specifically, the DFT to be computed is

$$X[k] = \sum_{n=0}^{N-1} x[n]\, W_N^{nk} \qquad (4.13a)$$

where we have introduced the short-hand notation

$$W_N \triangleq e^{-j2\pi/N} \qquad (4.13b)$$

The summation over $n$ in (4.13a) can be decomposed into two separate sums, one over the even-numbered values of $n$ and the other one over the odd-numbered values. Specifically, if we write an even $n$ in the form $2r$, the sum over even-numbered terms is:

$$X_e[k] \;=\; \sum_{r=0}^{N/2-1} x[2r]\; W_N^{2rk} \tag{4.14a}$$

Similarly, by making the change of variable $n \;=\; 2r+1$, the sum over odd-numbered terms becomes

$$W_N^k\; X_o[k] \;=\; \sum_{r=0}^{N/2-1} x[2r+1]\; W_N^{(2r+1)k} \;=\; W_N^k \sum_{r=0}^{N/2-1} x[2r+1]\; W_N^{2rk} \tag{4.14b}$$

With these definitions, (4.13a) becomes:

$$X[k] \;=\; X_e[k] \;+\; X_o[k]\; W_N^k \tag{4.14c}$$

Assuming that $N$ is a multiple of 2, one has

$$W_N^{2rk} \;=\; e^{j4\pi kr/N} \;=\; W_{N/2}^{rk}$$

so that each of the sums in (4.14a) and (4.14b) represents a DFT of length $N/2$. Therefore, we have decomposed an $N$-point DFT into two $N/2$ point DFTs followed by $N$ complex multiplications by the $W_N^k$ factors in (4.14c). In fact, only $N/2$ of these multiplications are distinct because $W_N^{N-k}$ is the complex conjugate of $W_N^k$. If $N/2$ is a multiple of 2, each of the two $N/2$-point DFTs (4.14b) and (4.14c) can further be decomposed into two DFTs of length $N/4$, each followed by $N/4$ distinct complex multiplications. Clearly, if $N$ is a power of 2, these decompositions can be repeated $\log_2\, N$ times until we are left with $N/2$ two-point DFTs, which are simple operations requiring no multiplication. For each of these $\log_2\, N$ decompositions, there is a total of $N/2$ complex multiplications, so that the total number of multiplications is about of $N/2\; \log_2\; N$.

### 4.4.2 Computational advantage of the FFT in filtering applications

It is of interest to compare the amount of computation required in the direct implementation of a convolution with that for the overlap-save method based on the FFT. Specifically, assume that the input signal $x[n]$ has a very long duration compared to the length $M$ of the filter's unit sample response, so that we can ignore overhead such as end effects and computation of the DFT of the unit-sample response. The direct (time-domain) convolution requires $M$ multiplications for each input point. On the other hand, if we use $N$-point FFTs to implement the overlap-save method, we require:

- $N/2\log_2 N$ complex multiplications for computing the forward FFT of each signal segment,
- $N$ complex multiplications for forming the product of the DFT of the segment with the DFT of the filter's impulse response
- $N/2\log_2 N$ complex multiplications for the inverse DFT.

This makes a total requirement of $N(\log_2 N+1)$ complex multiplications for each signal segment. Among the $N$ points in each segment, only $N-M+1$ are retained in the output because of the $M-1$ point overlap between successive input segments. Thus, the number of multiplications per input point is $\frac{N}{N-M+1}\,(\log_2 N+1)$.

The number of multiplications per input point required to implement an FIR filter are shown as a function of $M$ in Fig. 4.9 for both direct convolution and overlap-save methods with different values of $N$. Clearly, for filters longer than about 10 points, the overlap-save method is advantageous. It is also apparent that, for each filter length $M$, there is a value of the FFT length $N$ that provides optimum efficiency. Because we have counted *complex* multiplications, these calculations apply to complex signals. For real signals and filters, direct convolution is more efficient because it can be computed entirely with real multiplications, while overlap-save still requires complex multiplications, each of which involving 4 real multiplications. Thus, a more realistic performance crossover between the two methods is $M \approx 40$ for real signals.

## 4.5 Summary of Fourier transforms

In these notes, we have studied four different kinds of Fourier transforms:

1. The continuous-time Fourier transform (CTFT)
2. The discrete-time Fourier transform (DTFT)
3. The continuous time Fourier series (CTFS)
4. The discrete Fourier transform (DFT) or discrete Fourier series (DFS)

The last three transforms can be considered as special cases of the CTFT obtained by sampling (multiplying by a periodic impulse train) in either the time domain or the frequency domain, or both. Sampling in one domain corresponds to an aliasing operation (convolution by a periodic impulse train) in the opposite domain. To each of these transforms correspond both a convolution theorem and a product theorem. Convolutions can be either discrete or continuous, and either cyclic or "linear". Which of these four types of convolutions should be used follows from the properties of the signals to be convolved: discrete convolution for discrete signals, cyclic convolutions for periodic signals, and convolution modulo $N$ for signals that are both discrete and periodic. Table 4.1 summarizes the type of signals to which each of the 4 transforms applies, and gives the appropriate convolution and product theorems.

## 4.6 Summary

The $N$-point discrete Fourier transform of a signal is obtained by sampling its DTFT at frequency intervals of $1/N$. If the duration of the signal is no more than $N$, the $N$-point DFT provides a complete representation of the signal, and is related to the signal by the finite formulas

$$X[k] \;=\; \sum_{n=0}^{N-1} x[n]\; e^{-j2\pi kn/N}$$

$$x[n] \;=\; \frac{1}{N} \sum_{k=0}^{N-1} X[k]\; e^{j2\pi kn/N}$$

These formulas are mathematically the same as the Fourier series for discrete, periodic signals. This is because sampling at intervals of $1/N$ in frequency inherently generates an $N$-periodic signal $\tilde{x}[n]$, which coincides with the finite-duration signal $x[n]$ over one of its periods.

A major application of the DFT is the efficient implementation of convolution (filtering) operations for either two signals of finite duration or one FIR filter and one signal of indefinite duration (e.g. using the overlap-save method). These methods have to be used with caution because the product of two $N$-point DFTs is not the transform of linear convolution of the two signals, but the transform of their cyclic convolution *modulo* $N$:

$$\Big[\; \tilde{x}_N[n] \circledast_N \tilde{h}_N[n] \;\Big]\; R_N[n] \;\longleftrightarrow\; X[k]\; H[k]$$

Cyclic convolution coincides with linear convolution only if the length of the convolved signal is less than $N$.

## 4.7 Further reading

*Oppenheim and Schafer:* Chapters 8 & 9.
*Siebert:* Chapter 18, Sections 3-4.

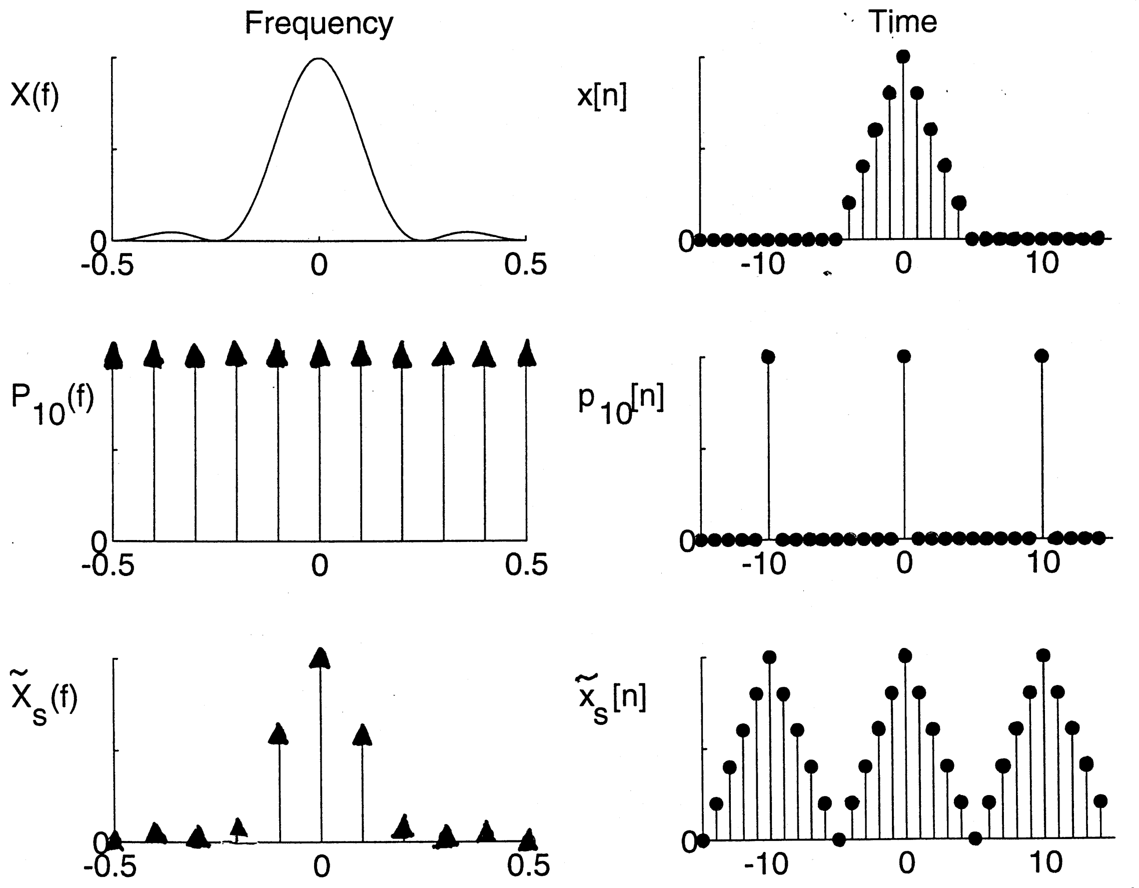


Figure 4.1: Representation of the DFT of a discrete-time signal, $x[n]$, by sampling its DTFT, $X(f)$. $\tilde{X}_s(f)P_N(f)$ and $\tilde{x}_s[n] = x[n] * p_N[n]$.

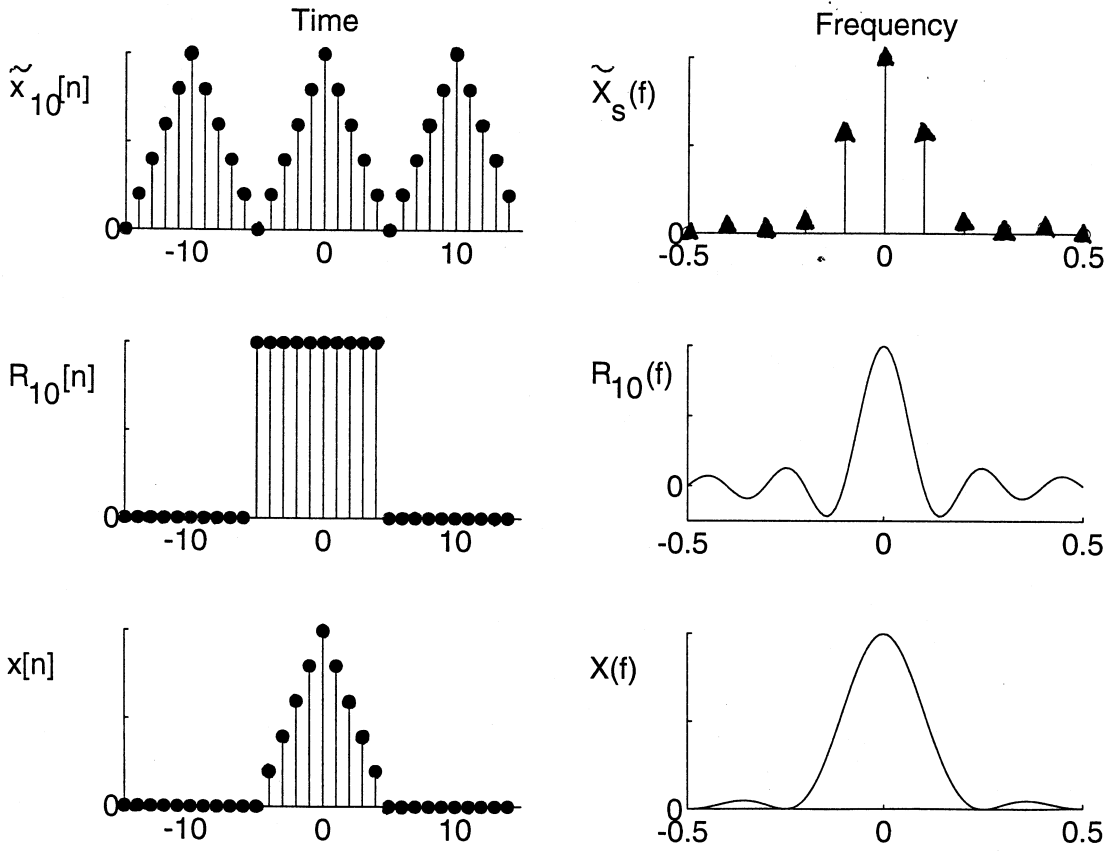


Figure 4.2: Reconstruction of the DTFT of $x[n]$ from DFT of the periodic signal $\tilde{x}_N[n]$. $x[n] = \tilde{x}_N[n]R_N[n]$ and $X(f) = \tilde{X}_N(f) \circledast R_N(f)$.

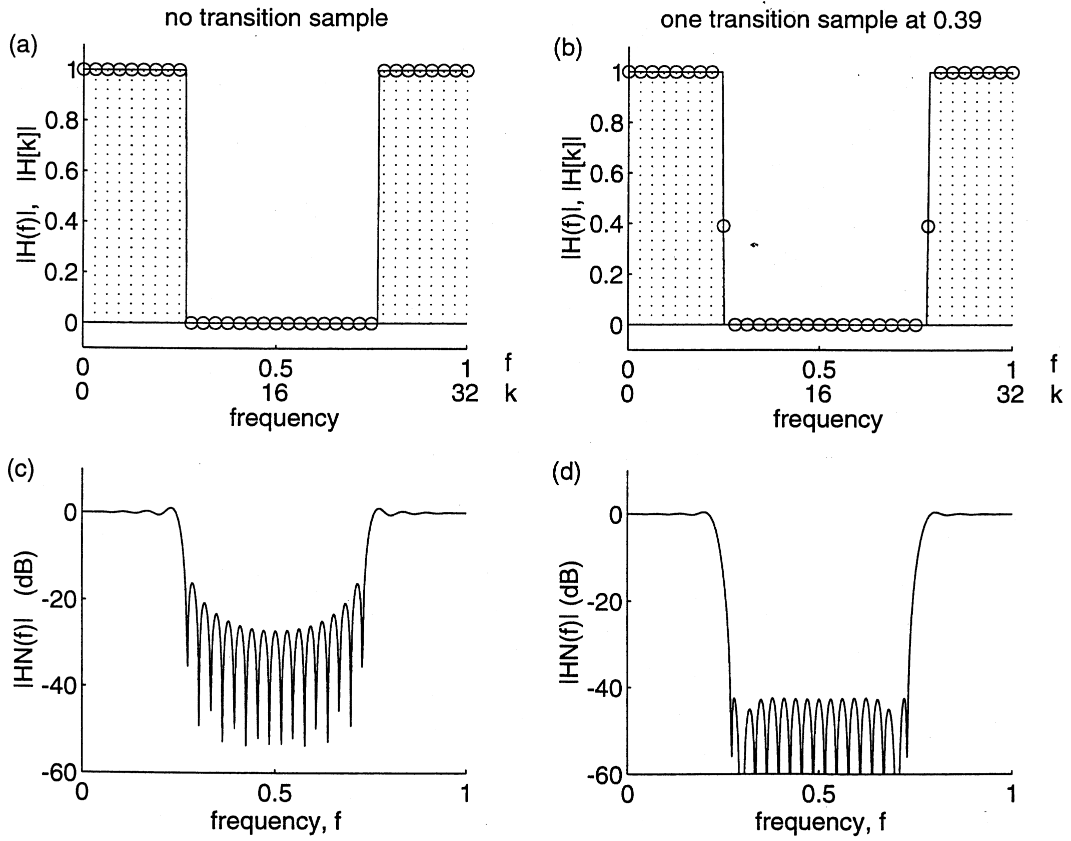


Figure 4.3: Filter design by frequency sampling. (a) Samples of ideal lowpass frequency response. (b) Addition of one transition sample. (c) Magnitude of frequency response for filter designed based on frequency samples in (a). (d) Magnitude of frequency response for filter with one transition sample as in (b).

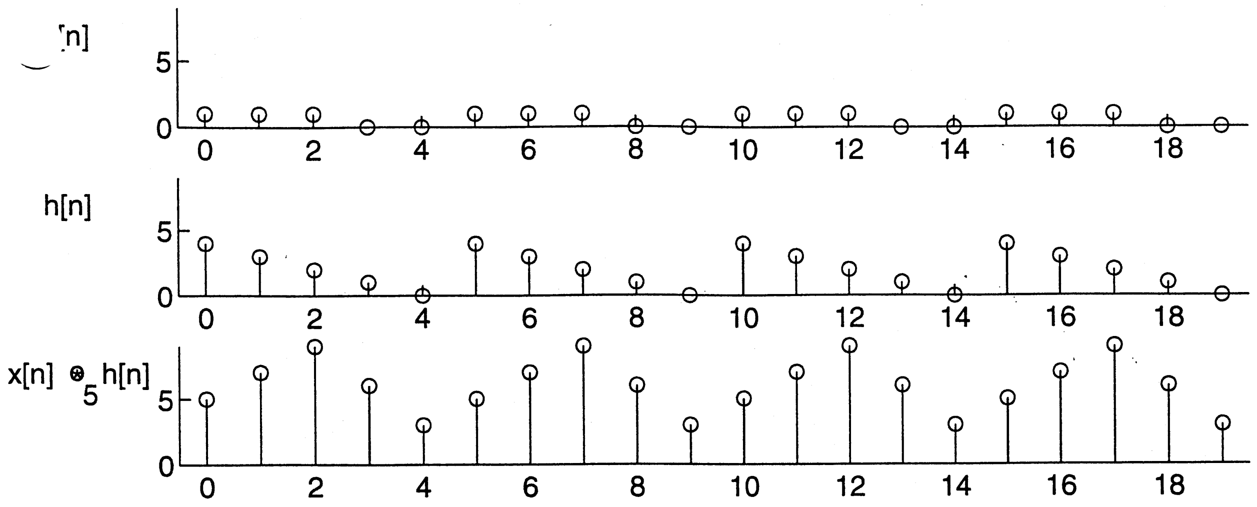


Figure 4.4: Cyclic convolution *modulo* 5 of periodic signals $x[n]$ and $h[n]$.

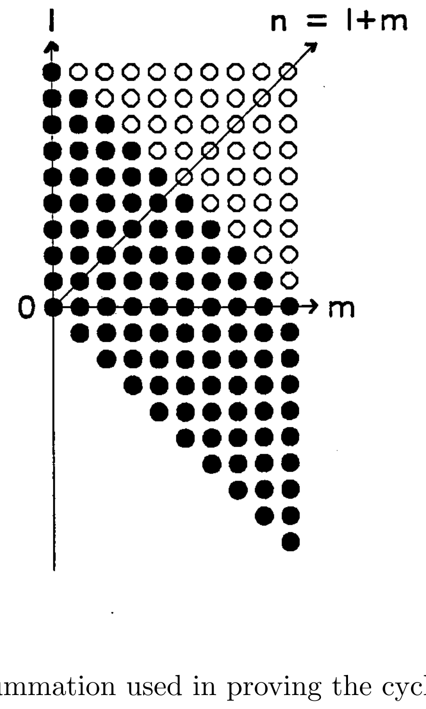


Figure 4.5: Regions of summation used in proving the cyclic convolution theorem.

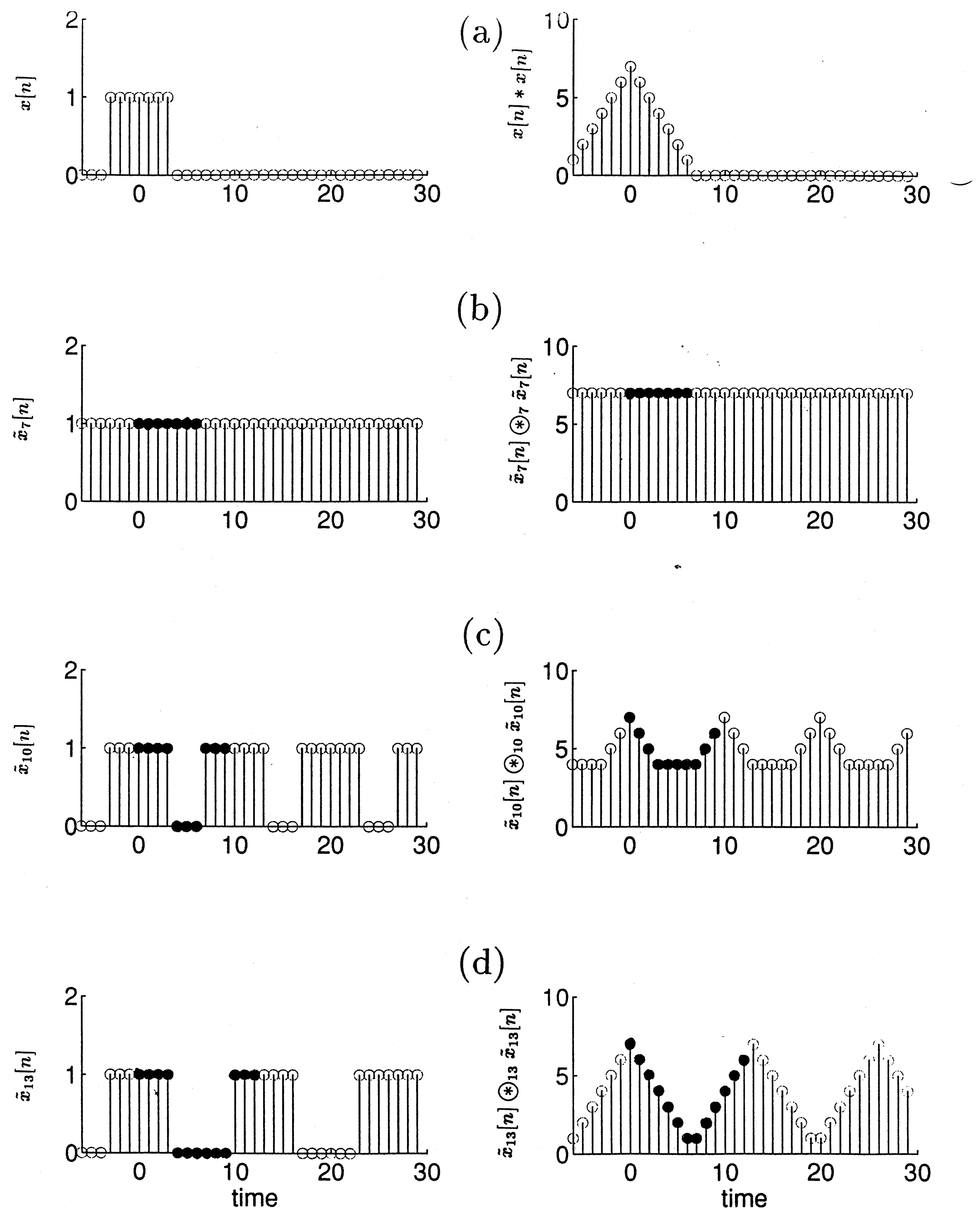


Figure 4.6: (a) The rectangular pulse $x[n]$ and the linear convolution of $x[n]$ with itself. (b) The periodic signal $\tilde{x}_7[n]$ obtained by repeating $x[n]$ with itself at intervals of 7 points and the cyclic convolution of $\tilde{x}_7[n]$ with itself. (c) The periodic signal $\tilde{x}_{10}[n]$ obtained by repeating $x[n]$ with itself at intervals of 10 points and the cyclic convolution $\tilde{x}_{10}[n]$ with itself. (d) The periodic signal $\tilde{x}_{13}[n]$ obtained by repeating $x[n]$ with itself at intervals of 13 points and the cyclic convolution of $\tilde{x}_{13}[n]$ with itself. (e) DTFT of the signals in (a). (f) DFT of one period of the signals in (b). (g) DFT of one period of the signals in (c). (h) DFT of one period of the signals in (d).

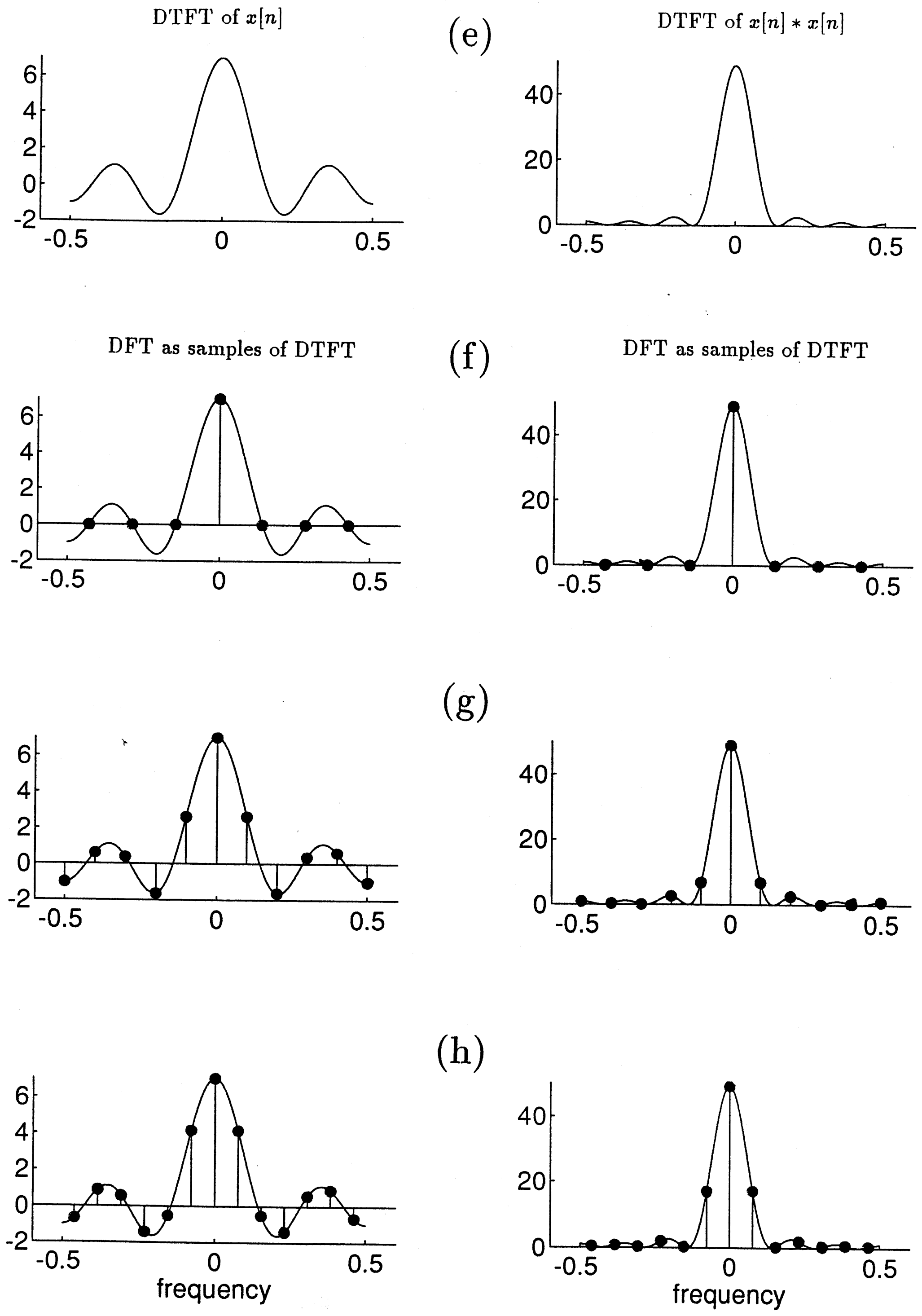
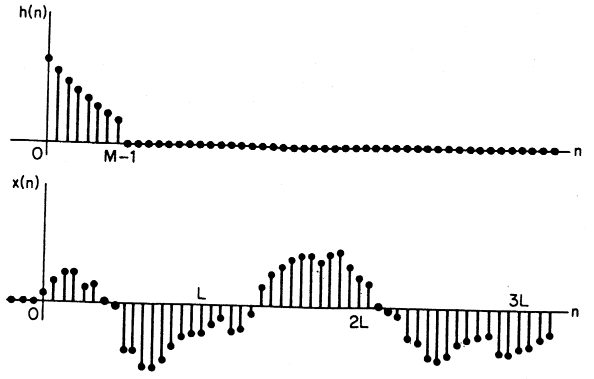
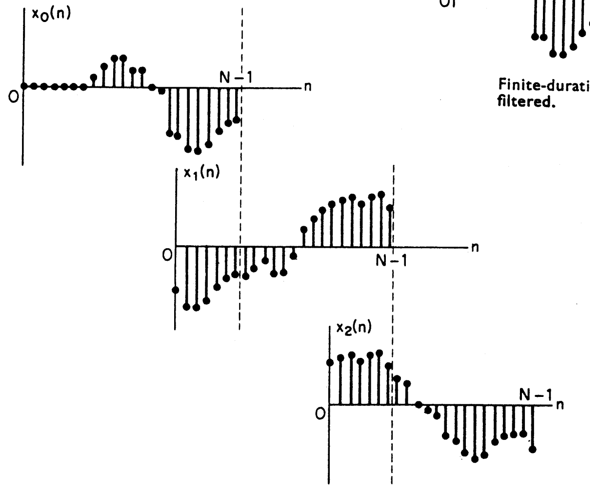


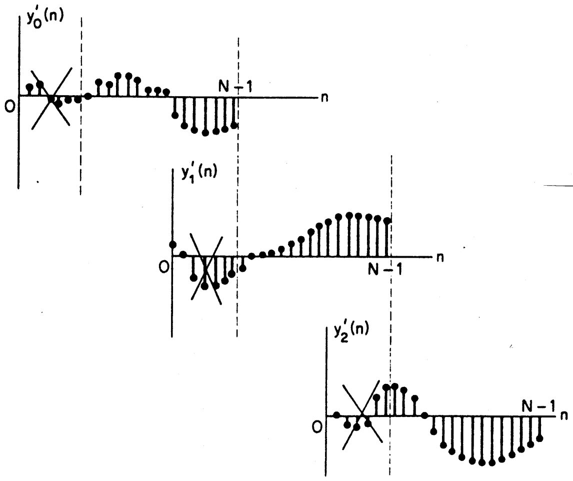
Figure 4.6 continued



Finite-duration unit-sample response $h(n)$ and signal $x(n)$ to be filtered.



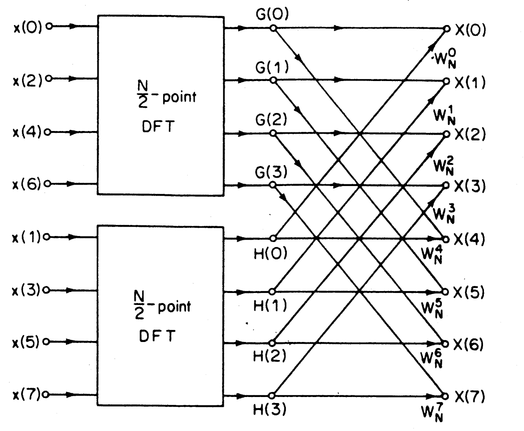
(a)



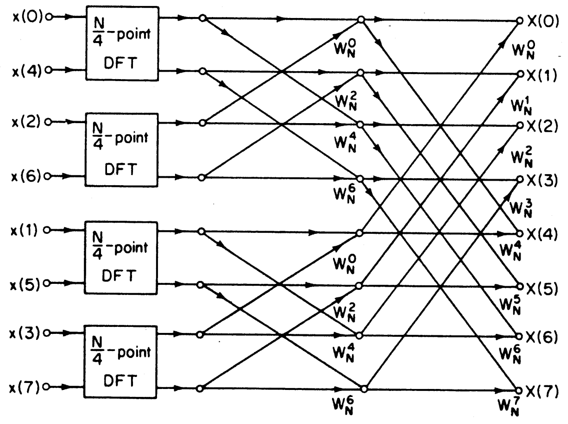
(b)

(a) Decomposition of $x(n)$ into overlapping sections of length $N$; (b) result of circular convolution of each section with $h(n)$. The portions of each filtered section to be discarded in forming the linear convolution are indicated.

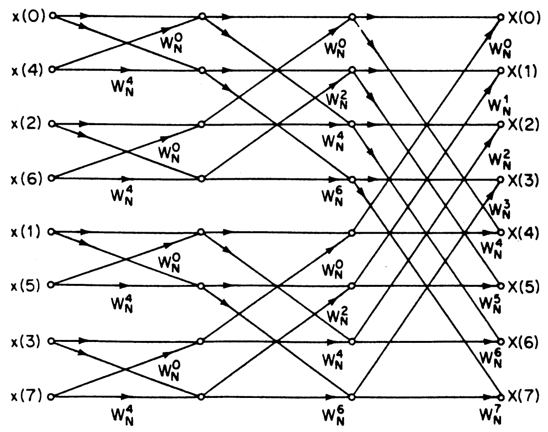
Figure 4.7: The overlap-save method of convolution (after *Oppenheim and Schafer*).



**Flow graph of the decimation-in-time decomposition of an *N*-point DFT computation into two *N*/2-point DFT computations (*N* = 8).**



8



**Flow graph of complete decimation-in-time decomposition of an eight-point DFT computation.**

Figure 4.8: The decimation-in-time FFT algorithm for $N = 8$ (after *Oppenheim* and *Schafer*).

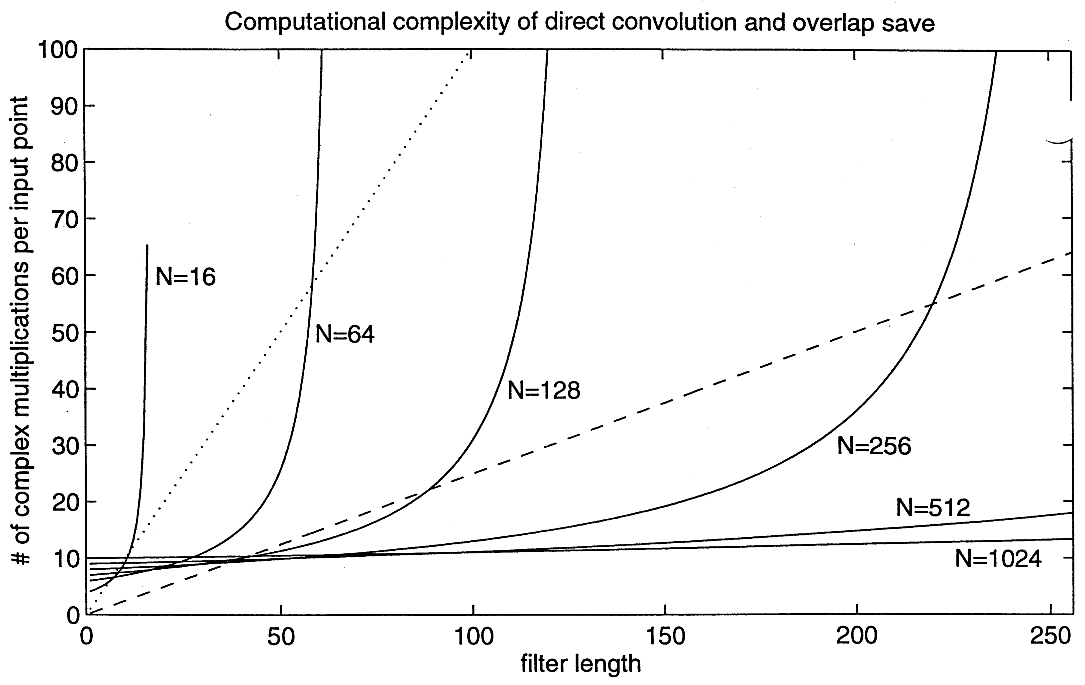
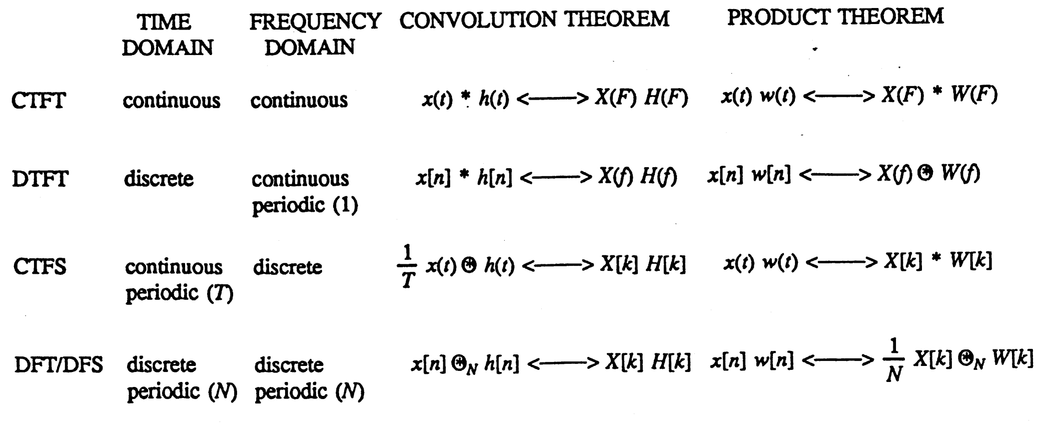


Figure 4.9: Number of complex multiplications per input point as a function of filter length for overlap-save method (solid lines: $N$ is FFT length), for direct convolution with complex data (dotted line), and for direct convolution with real data (dashed line).



| | TIME DOMAIN | FREQUENCY DOMAIN | CONVOLUTION THEOREM | PRODUCT THEOREM |
|---|---|---|---|---|
| CTFT | continuous | continuous | $x(t) * h(t) \longleftrightarrow X(F)\,H(F)$ | $x(t)\,w(t) \longleftrightarrow X(F) * W(F)$ |
| DTFT | discrete | continuous periodic (1) | $x[n] * h[n] \longleftrightarrow X(f)\,H(f)$ | $x[n]\,w[n] \longleftrightarrow X(f) \circledast W(f)$ |
| CTFS | continuous periodic ($T$) | discrete | $\frac{1}{T}\, x(t) \circledast h(t) \longleftrightarrow X[k]\,H[k]$ | $x(t)\,w(t) \longleftrightarrow X[k] * W[k]$ |
| DFT/DFS | discrete periodic ($N$) | discrete periodic ($N$) | $x[n] \circledast_N h[n] \longleftrightarrow X[k]\,H[k]$ | $x[n]\,w[n] \longleftrightarrow \frac{1}{N}\, X[k] \circledast_N W[k]$ |

Figure 4.10: Summary of Fourier transforms