Version:

TEX'd:
Oct. 23, 2013

11 The Discrete Fourier Transform and the FFT

PREVIEW

Classical numerical analysis techniques depend largely on polynomial approximation of functions for differentiation, integration, interpolation, and solution of differential equations. Fourier techniques, as presented in Chapter 8, use sinusoids and exponential functions to describe a function. Moreover, the Fourier techniques lead to the possibility of understanding physical phenomena in terms of the frequency components associated with a system or signal.

In Chapter 8, the Fourier series and Fourier transform were applied to signals and systems described by functions that are continuous functions of time. For computer analysis, which is in fact the modern approach to analyzing signals and systems, a signal or system is described by *samples* of the continuous function associated with the signal or system. Fourier

techniques are ideally suited to studying the effects of sampling such continuous functions.

This chapter introduces the Discrete Fourier transform (DFT) and an important algorithm to compute the DFT, called the Fast Fourier transform (FFT). Our emphasis is on the practical use of the FFT and the errors that can arise from sampling a signal and then computing its DFT. The final Section 11.6 explains the FFT algorithm.

11.1 FREQUENCY ANALYSIS OF SIGNALS

One of the most important uses of Fourier analysis, although not the only one by any means, is to analyze the frequency components of a *signal* derived from measurements of a physical variable of interest. These signals can be functions that represent quantities changing with time, such as voltage, force, or temperature. Light from a star can be analyzed to determine the spectrum of the light from the star. Similarly, the Fourier transform of radar echoes or audio (speech) signals is used to analyze the characteristics of the source of the signals. The study of the frequency content of such signals is called *spectral analysis*, and the Fourier transform is the primary method to compute the spectrum analytically.

Before attempting to apply Fourier analysis by computer to physical signals, it is necessary to thoroughly understand the frequency properties of such signals. This is one area where the motto, "The purpose of computing is insight, not numbers," mentioned in the preface, must be taken very seriously.[1]

First, suppose the physical signal of interest is a continuous function of time or another continuous variable. For computer analysis it is necessary to represent the signal by a finite number of values of the signal, usually called *samples* of the signal. Second, the analytical form of the Fourier methods, as presented in Chapter 8, assumes that the signal being analyzed is *infinitely* long in time. Obviously, any computer representation of the signal must be finite in length. Thus, representing a continuous signal for computer analysis leads to a finite-length vector of sampled points with the possibility of various errors in representation. Later in the chapter, we shall see that the spacing of the samples of a time-dependent signal and the length of time of observation of the signal will determine the information that can be derived from frequency analysis of the signal.

[1]The motto is from Richard Hamming's book, *Numerical Methods for Scientists and Engineers*.

As with any physical problem, some knowledge of the characteristics of the system being analyzed must be introduced into the Fourier analysis of the signals created by the system. For example, in human speech processing, the frequencies of interest generally range from 0 hertz (dc) to about 5000 hertz. White light, by contrast, has a frequency range from about 0.4×10^{15} hertz to 0.7×10^{15} hertz. The range of frequencies computed as the maximum frequency in the signal minus the minimum frequency is called the *bandwidth* of the signal. Thus, a typical speech signal would have a bandwidth of 5000 hertz. The bandwidth for white light is about 0.3×10^{15} hertz.

☐ EXAMPLE 11.1 ***Frequency Spectrum***

Comparing the frequency spectrum of single pulses gives important insight into the effect of changes in the characteristics of the pulse on the spectrum. Consider the pulse

$$P(t) = \begin{cases} A, & |t| < T/2, \\ 0, & \text{otherwise.} \end{cases} \tag{11.1}$$

According to the analysis in Chapter 8, the frequency spectrum of the pulse is determined by the Fourier transform of the pulse as

$$\mathcal{F}[P(t)] = F(f) = AT\frac{\sin(2\pi fT/2)}{(2\pi fT/2)}. \tag{11.2}$$

The effect on the spectrum $F(f)$ of changing the pulse width is shown clearly in Figure 11.1. The first zero of the 2-second pulse occurs at $f = 1/2$ hertz.

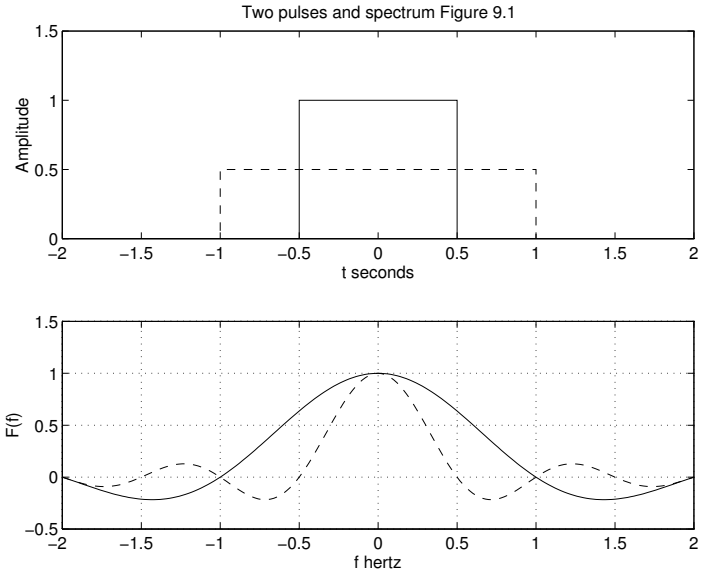

FIGURE 11.1 *Rectangular pulses and their spectra*

From Equation 11.2, the zeros of the frequency spectrum occur at the points

where $2\pi fT/2 = n\pi$ and n is an integer. Solving for f leads to the zero points as $f = n/T$, in which T is the total width of the pulse.

As the pulse width is decreased, the first zero crossing of the frequency axis moves up in frequency. The spectrum of the narrower 1-second pulse has its first zero at $f = 1$ hertz. If we assume that the frequencies of interest are primarily contained in the region defined from 0 hertz to the first zero, the bandwidth of the pulse for analysis could be considered to be approximately $1/T$ hertz. However, studying the analytical solution of Equation 11.2 shows that the actual bandwidth is infinite.

The magnitude of the frequency spectrum for a pulse of width T seconds is

$$|F(f)| = AT \left| \frac{\sin(2\pi fT/2)}{(2\pi fT/2)} \right|,$$

which has zeros at $f = n/T$ and maxima where

$$AT \frac{d}{df} \left| \frac{\sin(2\pi fT/2)}{(2\pi fT/2)} \right| = 0.$$

Differentiating leads to the equation

$$AT \left\{ -\frac{1}{f^2} \sin\left[2\pi f\left(\frac{T}{2}\right)\right] + \frac{2\pi T/2}{f} \cos\left[2\pi f\left(\frac{T}{2}\right)\right] \right\} = 0.$$

After dividing by the cosine term, the equation for the maxima is

$$\tan\left[2\pi f\left(\frac{T}{2}\right)\right] = \left[2\pi f\left(\frac{T}{2}\right)\right],$$

which becomes $x = \tan x$ by substituting $x = 2\pi fT/2$. The solutions are tabulated in the *Handbook of Mathematical Functions* by Abramowitz and Stegun listed in the Annotated Bibliography for this chapter. For a pulse with amplitude $A = 1$ and width $T = 1$ second, Table 11.1 shows the first four frequencies at which there are relative maxima of the magnitude of the spectrum. The values at these points show that the magnitudes of the frequency components diminish rapidly with increasing frequency. For practical purposes, the spectrum can be assumed to be zero above some frequency.

TABLE 11.1 *Maxima of* $F(f)$

f *hertz*	$\lvert F(f)\rvert$
0.0000	1.0000
1.4303	0.2172
2.4590	0.1284
3.4709	0.0913

The ideal pulses just discussed cannot exist in nature since every physical pulse has a finite rise time. The *rise time* is usually defined as the time for a

signal to change from 10% to 90% of its maximum value. A triangular pulse, as shown in Figure 11.2, will be used to demonstrate the effect of the rise time on its frequency spectrum.

*Triangular Pulse.*The triangular pulse of width T is defined by the equation

$$P_T(t) = \begin{cases} A\left(1 - 2\dfrac{|t|}{T}\right), & |t| < \dfrac{T}{2}, \\ 0, & \text{otherwise.} \end{cases} \tag{11.3}$$

This pulse has the Fourier transform spectrum

$$F(f) = \frac{AT}{2}\frac{\sin^2(2\pi fT/4)}{(2\pi fT/4)^2}, \tag{11.4}$$

with zeros at $f = 2n/T$. Comparing the spectra in Figure 11.1 and Figure 11.2 shows the frequency spectrum to the first zero for the triangular pulse is wider than that of a square pulse of the same width. However, the higher-frequency components in the spectrum of the triangular pulse go to zero more rapidly than those of a pulse with a zero rise time.

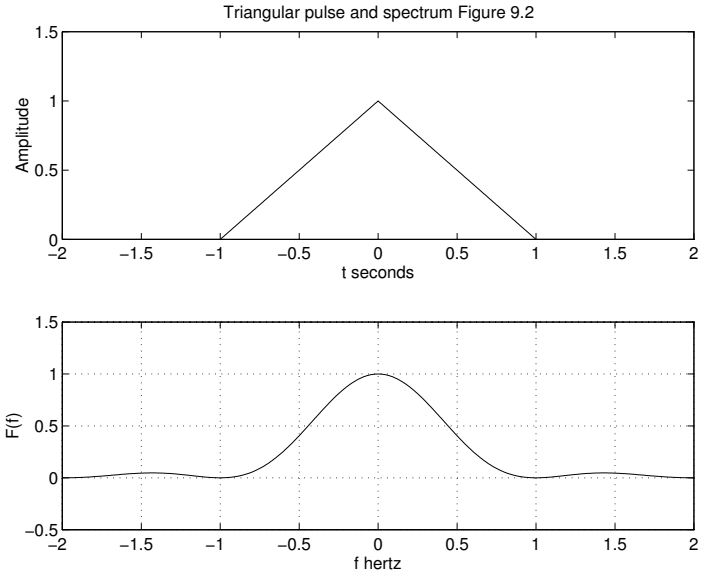

FIGURE 11.2 *Triangular pulse and its spectrum*

For the triangular pulse, the time taken for the signal to change from 0 to A is $T/2$ seconds, and the slope of the rising edge of the triangular pulse is $dP_T/dt = 2A/T$. Rewriting Equation 11.4 as

$$F(f) = \left(\frac{2A}{T}\right)\frac{\sin^2(2\pi fT/4)}{(\pi f)^2}$$

shows that the magnitude of the frequency components is proportional to the slope of the rising edge of the signal. In the case of typical pulses used in modern electronic equipment, the signal could change from 0 to 5 volts in 10

nanoseconds (10^{-8} seconds) or less. Thus, significant frequencies greater than $1/10^{-8}$ hertz, or 100 megahertz, could be present in the spectrum of the pulse.

□

11.2 DISCRETE AND FAST FOURIER TRANSFORMS

In this section, we treat the approximation of the exponential Fourier series and the integral Fourier transform of real-valued signals by sums of finite lengths and then present an algorithm for efficient computation. The discrete sum is called the *discrete Fourier transform*, or DFT, and the algorithm is the *fast Fourier transform*, or FFT. Table 11.2 summarizes the relationship between the time function and various Fourier techniques. In the table, $f(t)$ and $f(t_i)$ are functions of time. The parameter $\omega = 2\pi f$ radians per second is angular frequency, where f is frequency in hertz.

TABLE 11.2 *Table of Fourier techniques*

Name	*Characteristics*	*Typical use*
Fourier series	$f(t)$ continuous $F(\omega_i)$ discrete	Analysis of periodic functions and signals
Fourier transform	$f(t)$ continuous $F(i\omega)$ continuous	Frequency analysis of signals and systems
Discrete Fourier transform (DFT)	$f(t_i)$ discrete $F(\omega_i)$ discrete	Computation of other transforms Analysis of sampled signals
Fast Fourier transform (FFT)	$f(t_i)$ discrete $F(\omega_i)$ discrete	Algorithm to compute the DFT

Both the DFT and FFT deal with discrete functions in time and frequency. Thus, the DFT and the FFT transform a discrete sequence of values $f_n,\ n = 0, \ldots, N-1$, into another discrete sequence of values

$$F_k, \quad k = 0, \ldots, N-1.$$

In this section, we discuss the DFT and FFT in order to interpret the meaning of the F_k for practical signal processing.

DISCRETE FOURIER TRANSFORM (DFT)

Consider the portion of a continuous signal $f(t)$ shown in Figure 11.3. To determine the Fourier transform of $f(t)$ by computer analysis requires that the signal be sampled at a finite number of points. Typically, signals are sampled at equally spaced points in time. In Figure 11.3, assume the signal is sampled at intervals $\Delta t = T_s$ seconds to create a set of N points. The length in time of the sampled signal is $(N-1)T_s$ seconds. Thus, the original continuous signal is *sampled* and *truncated* for computer processing.

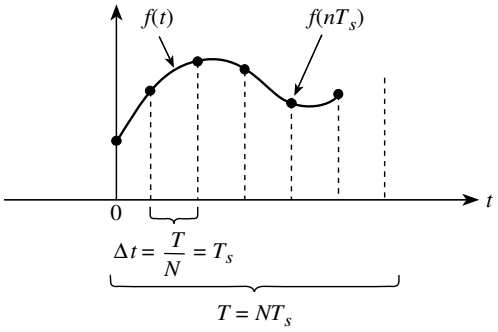

FIGURE 11.3 *Approximation of a signal by sampling*

Definition of DFT and IDFT Assume that a function $f(t)$ is defined at a set of N points, $f(nT_s)$ for $n = 0, \ldots, N-1$ values, as shown in Figure 11.3. The DFT yields the frequency spectrum at N points by the formula

$$F_k = F\left(\frac{k}{NT_s}\right) = \sum_{n=0}^{N-1} f(nT_s)e^{-i2\pi nk/N} \tag{11.5}$$

for $k = 0, \ldots, N-1$. Thus, N sample points of the signal in time lead to N frequency components in the discrete spectrum spaced at intervals $f_s = 1/(NT_s)$. The Inverse DFT (IDFT) is defined as

$$f_n = f(nT_s) = \frac{1}{N}\sum_{k=0}^{N-1} F\left(\frac{k}{NT_s}\right) e^{i2\pi nk/N} \tag{11.6}$$

for $n = 0, \ldots, N-1$. The IDFT is used to re-create the signal from its spectrum.

Real f(t) For a real function of time sampled at an even number of points N, the transform produces symmetry about the point $N/2$ because the real part of the transform is even and the imaginary part of the transform is odd. Table 11.3 summarizes the properties of the DFT applied to a real function of time. The DFT is assumed to have the form

$$F(f) = F_r(f) + iF_i(f), \tag{11.7}$$

where $F_r(f)$ is the real part of the transform and $F_i(f)$ is the imaginary part. A real and even signal has a transform that is a real and even function. Similarly, a real and odd signal produces an imaginary and odd function for the transform, as described in Table 11.3.

TABLE 11.3 *Properties of the DFT of* $f(t)$

$f(t)$	$F_r(f)$ *and* $F_i(f)$
Real	$F_r(f)$ even; $F_i(f)$ odd
Real and even	$F(f) = F_r(f)$; even
Real and odd	$F(f) = F_i(f)$; odd

In this chapter, we consider only functions $f(t)$ that are real functions of time.

Frequency Range The DFT frequencies described by Equation 11.5 appear to range from $f = 0$ to $f = (N-1)f_s$. However, this is *not correct* because of the symmetry of the transform results. Summarizing the relationship between sampling in time and the frequency components in the spectrum leads to the following conclusions:

1. The sample spacing in time T_s determines the highest positive frequency in the DFT as

$$F_{\max} = \frac{1}{2T_s} ;$$

2. The frequency spacing of the DFT components is

$$f_s = \frac{1}{T} = \frac{1}{NT_s}.$$

We stress these conclusions in discussing the DFT since it is necessary to understand that the frequency components computed by DFT analysis of a signal are determined by the choice of sample spacing and number of samples, *not* necessarily by the characteristics of the signal being analyzed. An important part of the analysis is to define the sampling parameters correctly so that the DFT represents the spectrum of the signal accurately. We shall reinforce these results in the chapter by numerous examples, discussions, and problems at the end of the chapter.

Figure 11.4 presents a comparison of the Fourier series, Fourier transform, and the DFT. The spectrum of a periodic signal computed as a Fourier series consists of a sequence of frequency components spaced at $\omega = 2\pi f_s = 2\pi/T$ hertz apart, where T is the period of the signal in

seconds. The Fourier transform is used to determine the spectrum of an infinite length signal which is continuous in time. The spectrum of the Fourier transform and Fourier series is defined on the frequency interval $(-\infty, \infty)$.

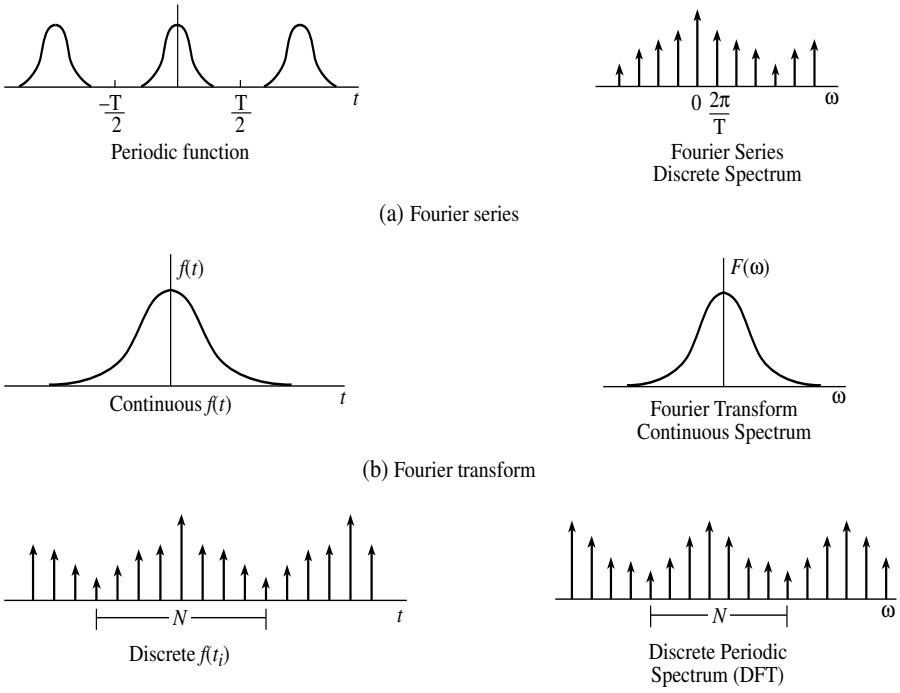

FIGURE 11.4 *Comparison of Fourier techniques*

We find that use of the discrete Fourier transform implies *periodicity* in both the time and frequency domain. This property of the DFT is proven in several of the texts listed in the Annotated Bibliography at the end of this chapter. In Problem 11.1, you are asked to show that the DFT as defined in Equation 11.5 is periodic. A similar result holds for the time function computed from the IDFT in Equation 11.6.

Thus, the discrete signal and its transform have the periodic properties

$$\begin{aligned} F(kf_s) &= F([k+N]f_s) = F(kf_s + F_p), \\ f(nT_s) &= f([n+N]T_s) = f(nT_s + T), \end{aligned}$$

where $F_p = 1/T_s$ is the period of the spectrum computed by the DFT. Thus, it is easily shown by direct substitution in the DFT that the result is periodic since

$$F(Nf_s) = F_0.$$

If the discrete spectrum from the DFT is replicated as a periodic function of period Nf_s defined over the entire real line, the result is called the *periodic extension* of the DFT. If the IDFT is used to compute the sampled function of time from its spectrum, the reconstructed signal has a periodic extension with period T seconds. This periodicity in time and frequency is characteristic of the DFT regardless of the true characteristics of the real signal being analyzed or its spectrum. It can be shown that the DFT and the IDFT give the exact results, neglecting numerical errors, for any discrete-time signal that is periodic with period N.

APPROXIMATION OF FOURIER TRANSFORMS

The DFT can be used to approximate the continuous Fourier transform. As defined in Chapter 8, the continuous Fourier transform is

$$\mathcal{F}[f(t)] = F(f) = \int_{-\infty}^{\infty} f(t)e^{-i2\pi ft}\, dt. \tag{11.8}$$

The frequency f in hertz is used as the parameter in this integral. The function $F(i\omega)$, where $\omega = 2\pi f$ is the frequency in radians per second, could be calculated as well. For the transform of a physical signal, we assume that $f(t) = 0$ for $t < 0$. Such signals are called *causal*. If the signal samples need to be shifted in time to meet this restriction, only the phase of the Fourier transform changes according to the shifting theorem presented in Chapter 8.

Using the sampled $f(t)$ with $t = nT_s$ and replacing f by the discrete frequencies $f_s = k/(NT_s)$ leads to the approximation of the Fourier transform as

$$F\left(\frac{k}{NT_s}\right) = T_s \sum_{n=0}^{N-1} f(nT_s)e^{-i2\pi nk/N} \tag{11.9}$$

for $k = 0, \ldots, N-1$. The factor $\Delta t = T_s$ replaced dt in the integral and is used as a multiplier of the DFT defined by Equation 11.5 in order to approximate the continuous Fourier transform. Problem 11.2 presents another derivation of the DFT approximation to the Fourier transform.

Various computer algorithms, called collectively *fast Fourier transforms* have been developed to compute the DFT. Few of them take the sampling time T_s in seconds into account since the typical FFT result, $F_0, F_1, \ldots, F_k, \ldots F_{N-1}$, is simply a function of the index k. Thus, the spectrum must be interpreted and scaled properly if the spectral components are to be displayed versus frequency in hertz.

FAST FOURIER TRANSFORM (FFT)

There are many algorithms that are generally called fast Fourier transform algorithms. In fact, developing techniques to improve the efficiency of calculation for the Fourier transform has been an active research area for many years. The algorithm is discussed in more detail in Section 11.6.

The FFT algorithms take advantage of the symmetry in the exponential functions $\exp(-i2\pi nk/N)$ to reduce the number of computations while computing the DFT. For example, a direct calculation of the DFT requires N^2 multiplications. The basic FFT requires approximately $N \log_2 N$ multiplications. If $N = 4096$ points, the FFT reduces the number of multiplications from more than 16 million to less than 50,000.

Figure 11.5 shows the important parameters for the DFT and the common FFT algorithms when applied to physical signals. In the figure, the magnitude of the spectrum computed by the FFT is plotted as

$$|F(f)| = \sqrt{[F_r(f)]^2 + [F_i(f)]^2}$$

using the notation of Equation 11.7. In the upper plot, the DFT components are plotted for the index $k = 0$ to $k = N - 1$ which emphasizes the symmetry about the index value $k = N/2$. The k-th positive digital frequency has the value $F = k/N$. The digital frequency, described in Section 10.9, corresponding to $F_{\max}$ is $F = 1/2$.

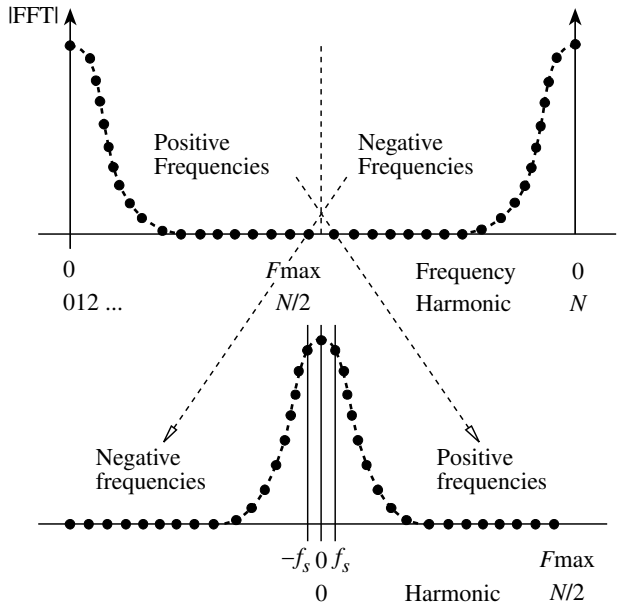

FIGURE 11.5 *Discrete Fourier transform spectrum*

The output of most FFT algorithms is folded in frequency, as shown in the upper plot of Figure 11.5 and the DFT spectrum is symmetrical around the frequency $F_{\max}$. This is called the *folding* frequency when the symmetry of the spectrum is being discussed.

Folding Frequency The theory of the folding shown in Figure 11.5 is that N points in time produce N points in the frequency domain. However, for a real signal, there are N complex numbers in the transform

with N real parts and N imaginary parts. The real part is even and the imaginary part is odd around the folding point as indicated in Table 11.3. Including the component at $f = 0$, there are really only $N/2 + 1$ unique points on the positive frequency axis.

Symmetry about the origin It is often convenient to plot the DFT spectrum showing the symmetry about the origin as in the lower plot of Figure 11.5. In terms of the components, the DFT of a real sequence possesses *conjugate symmetry* about the origin.

The procedure for plotting is to shift the upper half of the spectral components from $k > N/2$ to $k = N - 1$ to negative values $k - N$ so that the origin represents the zero of frequency. The results for $k > N/2$ are thus negative frequency results. The component at $N/2$ corresponds to the maximum positive frequency.

DFT summary Table 11.4 summarizes the DFT or FFT parameters when a real signal is sampled every T_s seconds for $(N-1)T_s$ seconds.

TABLE 11.4 *DFT parameters*

Parameter	*Notation*
Time domain:	
Sample interval	T_s (s)
Sample size	N points
Length	$(N-1)T_s$ (s)
Period (from IDFT)	$T = NT_s$ (s)
Frequency domain:	
Frequency Spacing	$f_s = \dfrac{1}{T} = \dfrac{1}{NT_s}$ (Hz)
Spectrum size	N components
Maximum frequency	$\dfrac{N}{2} f_s = F_{\max}$ (Hz)
Frequency period	$F_p = Nf_s = \dfrac{1}{T_s}$ (Hz)

The two important parameters in the time domain are the sampling interval T_s and the number of sample points N since other parameters can be derived in terms of these. Be careful to not confuse the period of the reconstructed signal from the IDFT, $T = NT_s$, with the total sampling time for the actual time signal. Also, the maximum frequency in the spectrum is $(N/2)f_s = 1/(2T_s)$ hertz, but the spectrum is periodic with period $Nf_s = F_p$ hertz.

DFT frequency in hertz In the frequency domain, the spectral lines of the DFT are spaced a distance f_s hertz or $\omega_s = 2\pi f_s$ radians/second

apart. The frequency *resolution* is thus $f_s = 1/(NT_s)$ hertz. As previously discussed, the spectral values for $k > N/2$ are simply negative frequency results. For example, $F[(N-1)f_s]$ in the FFT output corresponds to the frequency component $F(-f_s)$.

The practical consequences of the characteristics of the DFT and FFT are presented later in the section discussing practical signal analysis. First, we present computer examples to demonstrate the use of MATLAB commands to compute the DFT.

11.3 MATLAB FOURIER COMMANDS

MATLAB contains a number of commands to compute, manipulate and plot the DFT of a function. These are listed in Table 11.5. Except for the command **fourier**, the commands are used for numerical computation. The symbolic command **fourier** is part of the *Symbolic Math Toolbox*. If a function can be defined symbolically, **fourier** computes the Fourier integral transform. The *Signal Processing Toolbox* has additional commands for more advanced signal processing.

TABLE 11.5 *MATLAB commands for frequency analysis*

Command	*Result*
abs	Magnitude of FFT
angle	Phase angle of FFT in radians ($-\pi$ to π)
fft	FFT
fftshift	Moves zero frequency to center of spectrum
ifft	Inverse FFT
nextpow2	Returns next power of 2
unwrap	Unwraps phase angle beyond $-\pi$ to π
stem	Plots discrete sequence data
fourier	Symbolic Fourier transform

The MATLAB commands **fft** and **ifft** compute the DFT and inverse DFT, respectively, without regard for the actual sample spacing. The input to the functions is a given number of samples of the time function for **fft** or frequency function for **ifft**. Thus, the discrete functions that result from MATLAB Fourier computations functions must be scaled if an answer in physical units, such as hertz, is desired.

The result of **fft** is generally an array of complex numbers. The commands **abs** and **angle** can be used to compute the magnitude and angle

of the complex values, respectively. For plotting a two-sided transform, the command **fftshift** will rearrange the output of **fft** to move the zero component to the center of the spectrum.

INDEXING AND TRANSPOSE

MATLAB vectors are indexed from 1 to N, where N is the number of elements. However, the standard definition of the DFT, as in Equation 11.5, is indexed from 0 to $N-1$. Care must be taken in indexing when analyzing and plotting the results of MATLAB FFT calculations.

If the MATLAB transpose operator (') is used on a complex vector, the operation gives the complex conjugate transpose; that is, the sign of the imaginary part is changed as part of the transpose operation. This is not generally desired if the transpose of the DFT results are to be used since the sign of the phase angle would be changed. The operator (.') transposes a complex array but does not conjugate it.

□ EXAMPLE 11.2 *Function to Compute DFT*

The accompanying MATLAB script presents the function `clfftf`, which computes the approximate Fourier transform and the magnitude and phase of the two-sided spectrum of the function `ft`, defined at `N` sample points. The variable `Ts` defines the sample interval in seconds. Notice that the MATLAB command **fft** is multiplied by the sampling interval to approximate the actual Fourier transform.

MATLAB Script

```
Example 11.2
function [FT,FTmag,FTang] = clfftf(ft,N,Ts)
% CALL: [FT,FTmag,FTang] = clfftf(ft,N,Ts) Compute the DFT
%  approximation of the Fourier Transform
% Inputs:
% ft Sampled function of time f(nTs)
% N Number of sample points
% Ts Sample interval in seconds
% Outputs:
% FT Approximate Fourier transform using DFT
% FTmag Magnitude of spectrum
% FTang Phase in degrees
% Determine the two-sided spectrum
FT1=Ts*(fft(ft,N));            % Scale to approximate FT
FT=fftshift(FT1);              % Shift 0 to center
%
% Compute the magnitude and phase for the frequency values
%    in hertz fs=1/(N*Ts); fmax=1/(2*Ts)
%
FTmag=abs(FT);                 % Magnitude
FTang=(180/pi)*angle(FT);      % Phase in degrees
```

□

This function is used to compute the Fourier transform in other examples in this chapter. The file can also be modified to provide the spectrum in radians per second or to plot the results.

You will notice in this function and other scripts in this chapter that calculation of the FFT requires only a call to the MATLAB command **fft**. Most of the other executable statements scale or shift the result for plotting in physical units of frequency.

DFT OF THE EXPONENTIAL FUNCTION

The decaying exponential function is a good test function to use for exploring the accuracy and problems of the DFT applied to a continuous function.

☐ EXAMPLE 11.3 *FFT Computation of Spectrum*

Consider the function

$$f(t) = \begin{cases} e^{-t}, & t \geq 0, \\ 0, & t < 0, \end{cases}$$

with the Fourier amplitude spectrum

$$F(\omega) = \frac{1}{\sqrt{1 + (2\pi f)^2}},$$

as computed in Chapter 8. The phase of the spectrum is

$$\theta(f) = -\tan^{-1}\left(\frac{2\pi f}{1}\right).$$

The accompanying MATLAB script is used to compute the DFT of the function. The magnitude and phase of the result are plotted in Figure 11.6. After the number of points is input and the various parameters are computed, the FFT of the sampled signal is computed using the function `clfftf` described in Example 11.2. Even though the computed spectrum is discrete, the function **plot**, which interpolates through the points, is used when a large number of points are to be plotted. For a small number of plotted points, the MATLAB plotting command **stem** could be used.

A comparison of the DFT and exact Fourier transform plotted in the figure shows a close agreement in the amplitude spectrum. Experimenting with the number of points or the length of the period (`T`) in this example will show that taking a shorter interval or fewer sampling points leads to decreased accuracy.

MATLAB Script

```
Example 11.3
% EX11_3.M  Compute and plot the DFT of f(t)=exp(-t)
%  Creates f(t) sampled each Ts seconds for T seconds
%  Input:  N   -number of points input
%          T   -Period of signal
%      t0 =0   -start of time points
%  Calls clfftf to compute DFT
N=input('Number of points N= ')  % Sample N points
T=input('Period of signal T= ')
```

```
Ts=T/N;                          % Sampling interval
% Form the vector of time points and f(n*Ts)
t0=0;                            % Start of signal
ts=(t0:Ts:Ts*(N-1));             % Compute N points
ft=exp(-ts);
% Determine the spectrum
[Fft,Ffmag,Ffang]=clfftf(ft,N,Ts);
% Compute the frequency values in hertz fs=1/(N*Ts); fmax=1/(2*Ts)
%
fs=1/(N*Ts);                     % Frequency spacing
f=fs*linspace(-N/2,N/2-1,N);     % N points in frequency
% Plot Fexact and DFT result
w=2*pi*f;
Fexact=1./(sqrt(1+w.^2));        % Magnitude
Thetaex=-(180/pi)*atan(w);       % Angle in degrees
clf
subplot(2,1,1),plot(f,Fexact(1:N),'--',f,Ffmag(1:N));
title(['FT and DFT of exp(-t), N=',num2str(N), ' T= ',num2str(T),' sec'])
xlabel('Frequency in hertz')
ylabel('FT and DFT')
legend('FT','DFT')
subplot(2,1,2),plot(f,Thetaex(1:N),'--',f,Ffang(1:N));
xlabel('Frequency in hertz')
ylabel('Phase FT and DFT')
legend('FT','DFT')
```

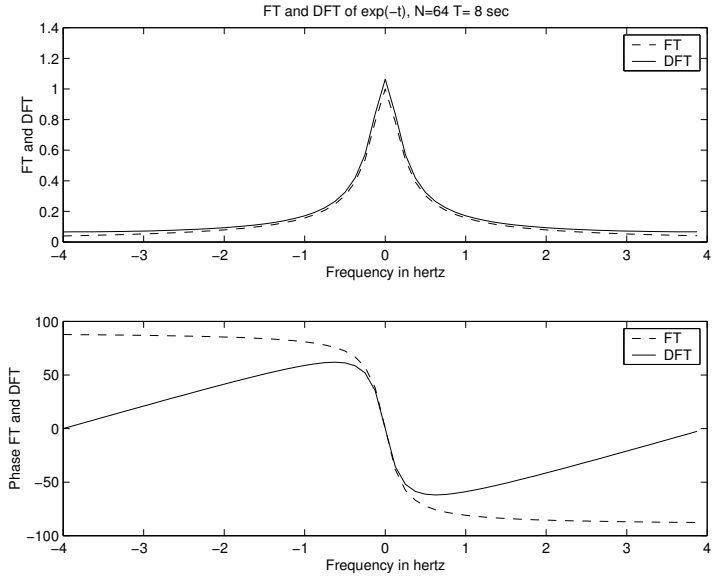

FIGURE 11.6 *Magnitude and phase of spectrum of* e^{-t}

□

DFT Phase Errors One potential problem with the DFT is that the signal being analyzed appears as a periodic function with period NT_s, where T_s is the sampling interval. Considering the decaying exponential in Example 11.3, the function is actually zero for all $t < 0$. However, applying the DFT to the time function can be viewed as taking the Fourier transform of a periodic waveform of exponential pulses.

The effect of periodicity of the analyzed signal was not obvious in computing the magnitude of the transform in Example 11.3. However, comparing the computed phase with the correct value ($-\arctan 2\pi f$) shows that the computed phase angle can be badly in error at the higher frequencies. The true phase changes from $\pi/2$ at negative frequencies to $-\pi/2$ at positive frequencies.

Caution. The function **angle** computes the arctangent of the ratio of the real to the imaginary part of a complex number. However, the result is very sensitive to the magnitude and sign of the elements. Values close to the minimum representable by the computer may cause unpredictable variations in the phase angle of the DFT when the command **angle** is used. These minimum values should be zero but are not due to roundoff errors. In some cases, it is best to plot the real and imaginary part of the DFT and set to zero any values that are below a predefined amount. A value such as $10\times$`eps` could be used as the minimum allowed value, where `eps` is the smallest MATLAB number for a particular computer.

11.4 PRACTICAL SIGNAL ANALYSIS

Assume that a physical signal is to be analyzed using the DFT to determine the spectral components. Figure 11.7 presents a simplified diagram of the input stage of a data acquisition system.

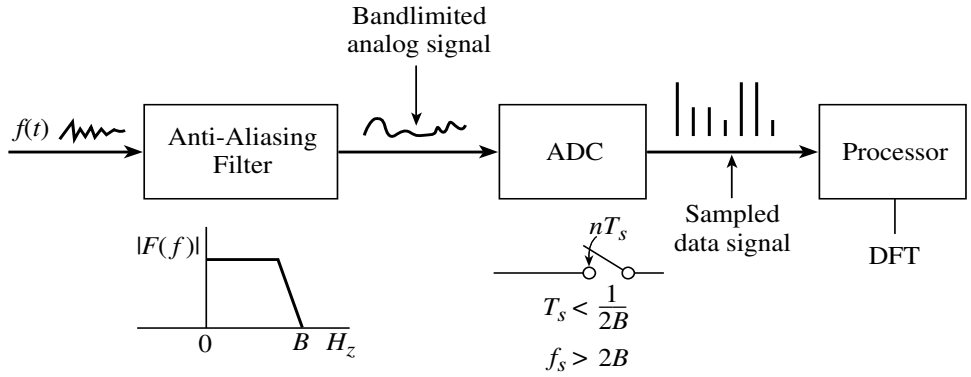

FIGURE 11.7 *Data acquisition system*

The signal will be assumed to be a *continuous-time* signal, which is often referred to as an *analog* signal. A system that converts the analog signal to a sequence of time samples suitable for computer processing is called a *data acquisition system*. The first component is a filter that eliminates unwanted frequency components of a signal. In this case, the filter limits the frequencies in the signal to the range 0 to B. A signal filtered in this way is said to be *bandlimited* to B hertz. In a data acquisition system, the filter is sometimes called a *presampling filter*. Such a filter is also called an *anti-aliasing* filter because it is intended to prevent an error called aliasing, as explained later after the sampling theorem is presented.

The analog-to-digital converter (ADC) samples the filtered signal each T_s seconds to create a *sampled-data* or *digital* signal, which is the discrete-time representation of the analog signal after sampling. Typically, the samples are stored in a computer's memory for processing.

SAMPLING

Two of the most important questions in the specification of a data acquisition system such as that shown in Figure 11.7 are the following:

1. How often should the analog signal be sampled?
2. How long should the signal be sampled?

If the highest frequency of interest in the signal is B hertz and the frequency spectrum of the signal is limited to B hertz by the anti-aliasing filter, the *sampling theorem* answers the question in Part 1. The theorem is the cornerstone of practical and theoretical studies in electronic communication.

Sampling Theorem Suppose the highest frequency contained in an analog signal $f(t)$ is $f_{\max} = B$ hertz. Then, if $f(t)$ is sampled periodically at a rate of

$$f_{\text{sample}} = 1/T_s > 2B \tag{11.10}$$

samples per second, the signal can be exactly recovered from the sample values.[2]

Considering the sampling interval T_s, the sampling theorem states that the analog signal must be sampled so that more than two samples per cycle of the highest frequency in the signal are taken. Since a cycle of the sinusoid at B hertz is $1/B$ seconds in length, the sampling interval must be less than $1/(2B)$ seconds, as indicated in the sampling theorem. The frequency $2B$ is called the *Nyquist frequency*, and the corresponding sampling rate is the *Nyquist rate*. Thus, an analog signal must be sampled at a rate greater than the Nyquist rate if errors due to sampling at too low a rate are to be avoided.

[2]The interpolation function to reconstruct the analog signal is discussed in most textbooks that cover digital signal processing. Several of these books are listed in the Annotated Bibliography at the end of this chapter.

For example, if the signal contains frequencies up to $B = 1000$ hertz, the Nyquist frequency is 2000 hertz and the sampling theorem requires sampling at a rate higher than 2000 samples per second. In terms of the parameters defined earlier for the DFT, the sampling interval corresponding to the Nyquist rate must be

$$T_s < \frac{1}{2B} = \frac{1}{2000} \quad \text{seconds},$$

or $T_s < 0.5$ milliseconds, between samples. Another way to view this is to realize that a 1000-hertz sinusoid repeats every millisecond. Sampling more than two samples per cycle requires a sampling interval of less than $1/(2 \times 10^3)$ seconds. The length of time the signal is sampled would be $(N-1)T_s$ seconds if N samples are taken.

Sampling and the DFT Suppose that an analog signal is sampled at a rate

$$f_{\text{sample}} = \frac{1}{T_s} > 2B \quad \text{samples/second}.$$

The highest possible frequency in the DFT spectrum would be $F_{\max} = 1/(2T_s)$ hertz. If the signal is bandlimited to B hertz and sampled properly, the component at the DFT maximum frequency should be zero since $F_{\max} > B$ hertz.

Although the sampling theorem gives an elegant solution to the sampling problem, a number of practical considerations intervene to complicate the selection of a sampling rate. First, the reader should review the examples in this chapter and in Chapter 8 and notice that none of the ideal example signals have a bandlimited spectrum. In practice, this is not a problem because any physical signal is in fact bandlimited. The physical system that created the signal cannot oscillate above some finite frequency, and the energy content of the signal is negligible beyond a certain frequency. However, the highest frequency may not be known, so an anti-aliasing filter can be used in a data acquisition system to limit the maximum frequency to a known value, say, B hertz. An ideal filter would leave frequency components of the signal below B hertz unaltered and eliminate all components above B hertz. Sampling the signal at a rate greater than $2B$ samples per second would thus satisfy the sampling theorem.

Because the physical filter in a data acquisition system is not ideal, frequencies above the desired bandlimited frequency B can be present in the signal even after filtering. To compensate for the nonideal characteristics of the anti-aliasing filter, the filtered signal is often sampled at a much higher sampling rate than that required by the theorem. In practice, a signal may be sampled at 3 to 10 or more times the rate dictated by the minimum ideal rate of $2B$ samples per second. This *oversampling* reduces sampling errors but increases the number of samples that must be stored and processed.

Another problem cannot be completely overcome. For the sampling theorem to be correct, the signal must theoretically be sampled in time over the interval $(-\infty, \infty)$. This is often stated as "a bandlimited signal cannot be limited in time." The ideal pulses in previous examples are time limited and thus cannot have bandlimited spectra. However, the desired frequency resolution in the spectrum of a signal can be used to define the length of time necessary to sample the signal. Since the DFT resolution is

$$\Delta f = f_s = 1/(NT_s) \quad \text{hertz},$$

where T_s is the sampling interval for the analog signal, if $NT_s = 2$ seconds, it is possible to resolve frequencies as low as $f = 1/2 = 0.5$ hertz.

If the signal is not sampled at a high-enough rate to satisfy the sampling theorem, the errors cannot be undone once the sampling occurs. Also, if the signal is not sampled long enough, the resolution in frequency cannot be improved.

☐ EXAMPLE 11.4 ***Sampling Example***

This example defines the relationship between sampling interval, frequency resolution, and number of samples for the DFT. In terms of previous notation, T_s is the sampling interval in seconds, f_s is the frequency resolution, and N is the number of sample points in time and in frequency.

Consider an analog signal with frequencies of interest up to 1200 hertz. The desired frequency resolution is 0.5 hertz. Thus, the signal should be filtered so that $B = 1200$ hertz. This filtering removes frequencies in the signal above 1200 hertz and *noise* above B hertz. The noise consists of unwanted signals added to the desired signal that are the result of environmental effects as the signal is transmitted to the data acquisition system.

By the sampling theorem, the sampling interval in time must be

$$T_s < \frac{1}{2B} = \frac{1}{2400} \quad \text{seconds},$$

so that at least 2400 samples per second are needed. For a resolution of 0.5 hertz, $T = 1/0.5 = 2$ seconds. The total number of points required is thus

$$N = \frac{T}{T_s} = \frac{2}{(2400)^{-1}} = 4800.$$

If N is to be a power of 2 for the FFT algorithm, $2^{13} = 8192$ samples would be taken. The sampling rate could be increased to 4096 samples per second, which is sampling at a rate corresponding to about 3.4 times the highest frequency of interest.

☐

Spectral Analysis The measurement and study of the frequency content of a signal is called *spectral analysis*. Many systems designed to measure the Fourier spectrum of an arbitrary signal $f(t)$ sample the signal and perform Fourier analysis using the DFT. If the sampled data points are stored in computer memory, the discrete signal can be analyzed using the DFT.

☐ EXAMPLE 11.5 ***MATLAB Spectral Analysis***

The time signal shown in Figure 11.8 was sampled with the parameters

$$\texttt{N} = 128 \quad \text{and} \quad \texttt{Ts} = 1/128.$$

The DFT analysis yields the spectrum of Figure 11.8, with a maximum frequency of 64 hertz and a resolution of 1 hertz. From the DFT analysis, the spectrum appears to have a strong component at 20 hertz. In fact, the MATLAB function that generated the signal was

```
>> N=128;
>> Ts=1/128;
>> t=0:Ts:Ts*(N-1);
>> ft=sin(2*pi*20*t);
>> ft=ft+randn(size(t));
>> save clex115.mat
```

where the MATLAB function **randn** generates random values (noise) superimposed on the sine wave. The MATLAB script analyzes the signal and plots the result.

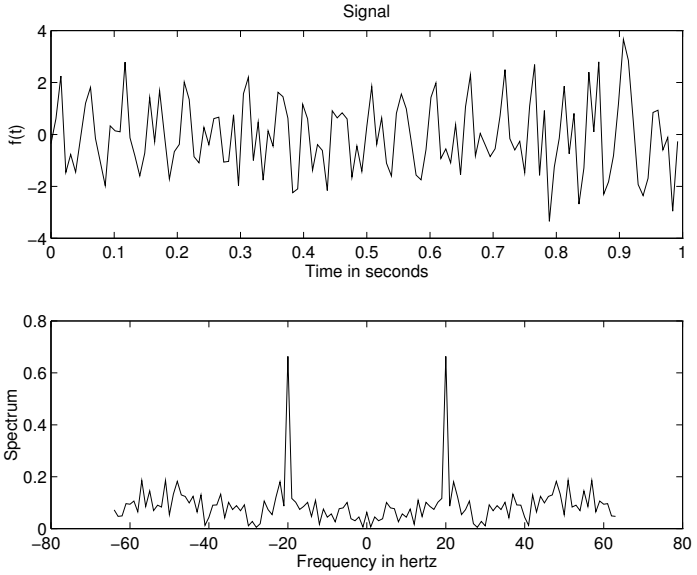

FIGURE 11.8 *Noisy signal and spectrum*

MATLAB Script

```
Example 11.5
% EX11_5.M Compute the spectrum of a signal saved
%   in file CLEX95.MAT. The data are:
%   N   samples of the |FFT| are plotted
%   Ts   sampling interval in seconds
%   t   time points
```

```
%   ft  function f(t)
% Calls clfftf to compute DFT
load clex95.mat                    % Load N,Ts,t,ft
fs=1/(N*Ts);                       % Frequency spacing
fhertz=fs*linspace(-N/2,N/2-1,N);  % Create frequency axis
[FT,FTmag,FTarg]=clfftf(ft,N,Ts);  % Compute DFT
% Plot f(t) and DFT
subplot(2,1,1), plot(t,ft)         % Time function
title('Signal')
xlabel('Time in seconds'), ylabel('f(t)')
subplot(2,1,2),plot(fhertz, FTmag) % Spectrum
xlabel('Frequency in hertz')
ylabel('Spectrum')
```

□

11.5 PRACTICAL SIGNAL SAMPLING AND DFT ERRORS

One way to approach an error analysis of the DFT is to compare the transform to the exponential (complex) Fourier series studied in Chapter 8. The Fourier series represents a periodic analog signal that exists for $-\infty < t < \infty$. The resulting line spectrum can cover the range $-\infty < f < \infty$ with a resolution of $1/T$ hertz, where T is the period of the analog signal in seconds. In contrast, the DFT analyzes the analog signal using a finite number of N samples in the range $0 \leq nT/N < T$. The useful frequency range is limited to $0 \leq k/T < N/2T$ and is periodic with a period of N frequency points. The limits imposed by the finite nature of the DFT lead to several possible errors that must be considered when the DFT is used to determine the spectrum of an analog signal.

ALIASING ERROR

If a signal is sampled at a rate that is equal to or less than the Nyquist rate defined by the sampling theorem, *aliasing errors* can occur. The name implies that one signal can be "impersonated" by another signal. The impersonation caused by too-coarse sampling (undersampling) describes high-frequency components of the true spectrum appearing as low-frequency components in the DFT spectrum.

The effect can be seen in several practical situations. Spoked wagon wheels on a stagecoach going forward occasionally appear to rotate backward when western films are shown on TV because the image of the rotating wheel is being sampled. The continuous motion is represented by a series of pictures (samples) taken at a time interval that is too slow to capture the true motion of the wheel. The effect can also be seen when a stroboscope is used to capture the motion of rotating equipment. If the

light from the stroboscope is flashed at times that are slightly less than the period of rotation, the equipment appears to be rotating a slower rate than the true rate. If the stroboscope flashes exactly at the period of rotation, the rotating portion of the machine appears to be stationary.

EXAMPLE 11.6 ***Aliasing***

A dramatic example of aliasing in signal processing is shown by the function

$$f(t) = \cos(2\pi f t),$$

which should be sampled at a rate greater than $2f$ samples per second. Suppose we sample at a rate corresponding to one-half the Nyquist frequency, or only f samples per second. The sampling interval is thus $T_s = 1/f$, and the times of sampling are

$$0, \frac{1}{f}, \frac{2}{f}, \ldots, \frac{N-1}{f} \quad \text{seconds}$$

if N points are taken. Notice that $f(nT_s) = \cos(2\pi f n T_s) = 1$ for every point $0 \leq n < N$. Now consider the constant function $f(t) = 1$ sampled at the same rate. Both the cosine and the constant yield the time series

$$f(nT_s) = [1, 1, 1 \ldots, 1],$$

so we conclude that the cosine sampled at one-half of the Nyquist rate impersonates a constant or dc value. The DFT of the cosine signal would indicate a spectral value at $f = 0$ but not at the frequency of the cosine, since the maximum frequency in the DFT spectrum is $f/2$. The only solution to eliminate the aliasing error is to sample at a rate that satisfies the sampling theorem. In practice, if the high-frequency components of the signal diminish rapidly, the effect of aliasing may not be significant. The anti-aliasing filter in Figure 11.7 ensures that the signal will be bandlimited in a practical data acquisition system.

□

LEAKAGE ERROR

The IDFT produces an N-sample periodic time-series from the spectrum created by the DFT. An error called *leakage* occurs when the actual signal being sampled is not perfectly periodic over an N-sample interval. The term implies that energy "leaks" from one frequency to another in the DFT spectrum. Hence, the signal appears to have frequency components that may not be present in the spectrum of the analog signal.

Truncating the time signal at $t = (N-1)T_s$ has the effect of introducing a discontinuity in the signal unless it is zero at the end point or the signal is periodic with period $T = NT_s$. Increasing the DFT period of the signal by increasing N tends to reduce the leakage effect unless the analog signal is actually periodic. Problem 11.9 allows you to investigate the effect.

PICKET FENCE EFFECT

The fact that only discrete frequencies k/T appear in the DFT spectrum leads to the conclusion that actual signal components at other frequencies will not be present in the spectrum. This is sometimes colorfully referred to as the *picket fence effect* since the scene behind this fence can be viewed only between the pickets. The pickets in this case correspond to the frequencies that cannot be resolved in the spectrum. Increasing the overall sampling time T while holding the sampling interval constant leads to greater resolution in the spectrum. This is equivalent to increasing the number of samples N.

SUMMARY OF ERRORS

Table 11.6 summarizes the major DFT errors that may arise from sampling at too low a rate or not sampling long enough. For the parameters listed in the table, $T = NT_s$, so there are really only two parameters to manipulate. Several problems at the end of this chapter are intended to emphasize the effects on the DFT spectrum of changing the sampling interval or the number of samples.

TABLE 11.6 *DFT errors*

Condition	*Cure*
Aliasing	Increase maximum frequency (decrease T_s)
Leakage	Increase frequency resolution (increase N)
Picket fence	Increase frequency resolution (increase N)

11.6 ANALYSIS OF DFT FOR COMPUTATION (OPTIONAL)

Using and interpreting the FFT results previously presented in this chapter does not require a complete understanding of the algorithm itself since the FFT is only one particular method of computing the discrete Fourier transform (DFT). In this section, we only describe particular cases of the FFT algorithm for simplicity. For example, we restrict the value of N to an *even* number, and furthermore the discussion often assumes that $N = 2^m$ for convenience in understanding and computation. References in the Annotated Bibliography for this chapter treat other cases with a more general FFT algorithm.

Consider the discrete Fourier transform of Equation 11.5 in the form

$$F_k = \sum_{n=0}^{N-1} f_n e^{-i2\pi nk/N} \qquad k = 0, 1, \ldots, N-1, \tag{11.11}$$

where nT_s is replaced with n and k/NT_s is replaced by k. This describes the components in terms of their index values. After the DFT is calculated, the values can be scaled to reflect the frequency resolution and range.

To explore the characteristics of the exponential factor in Equation 11.11, we first define

$$W_N = e^{-i2\pi/N}. \tag{11.12}$$

This is the complex number

$$W_N = e^{-i2\pi/N} = \cos\frac{2\pi}{N} - i\sin\frac{2\pi}{N}$$

with the property that $|W_N| = 1$ so that every one of these complex numbers lies on the unit circle in the complex plane. In Equation 11.11, the exponential factors have the form W_N^{nk} with the values

$$W_N^{nk} = (e^{-i2\pi/N})^{nk} = \cos 2\pi nk/N - i\sin 2\pi nk/N. \tag{11.13}$$

The term $W_N^0 = 1$. In cases where the argument is a multiple of π, the value is ± 1. If the argument is a multiple of $\pi/2$, the value is $\pm i$.

Since $W_N = z$ is a complex number, we can write the terms as the solutions of an equation of the form

$$z^N = a,$$

which is the formula for the Nth root of a. The solution uses the theorem of De Moivre

$$(\cos\theta + i\sin\theta)^N = \cos N\theta + i\sin N\theta, \tag{11.14}$$

which follows from the formula for multiplication of complex numbers written in polar form as presented in Chapter 2. Writing a in polar coordinates as $a = r(\cos\theta + i\sin\theta)$, the roots of a are

$$a^{1/N} = \sqrt[N]{r}\left[\cos\left(\frac{\theta}{N} + p\frac{2\pi}{N}\right) + i\sin\left(\frac{\theta}{N} + p\frac{2\pi}{N}\right)\right], \tag{11.15}$$

where p takes the values $0, 1, \ldots, n-1$. The term $\sqrt[N]{r}$ is the positive real Nth root of r, and for the specific case $|a| = 1$, this value is 1.

Consider the values p/N for integers $p = 0, 1, 2, 3$ with $N = 4$. Writing $W_4^p = e^{-\theta}$, where $\theta = 2\pi p/4$, these values correspond to the angles

$$\theta = 0\ (2\pi),\ \theta = \pi/2,\ \theta = \pi \text{ and } \theta = 3\pi/2$$

in the complex plane, respectively. The exponential W_4^p then takes the values

$$\begin{aligned} W_4^0 &= e^0 &&= 1 \\ W_4^1 &= e^{-i\pi/2} &&= -i \\ W_4^2 &= e^{-i\pi} &&= -1 \\ W_4^3 &= e^{-i3\pi/2} &&= i. \end{aligned} \tag{11.16}$$

Notice that when $N = 4$, these results correspond to the four roots of 1 in the equation $z^4 = 1$, where z is a complex number. The square roots of unity are ± 1, and the square roots of these roots are $1, -1, i, -i$ as is easily verified. The sum of the roots $(1 - i - 1 + i)$ is zero.

The repeated values in the relationship of Equation 11.13 can be written

$$W_N^{nk} = W_N^{nk \bmod N}, \tag{11.17}$$

where $nk \bmod N$ is the remainder upon division of nk by N. Thus, the term W_4^4, if formed, would be the same as W_4^0. Similarly, $W_4^5 = W_4^1$. The results thus far lead to observations about the periodicity and symmetry of the terms in the DFT.

Periodicity and Symmetry in the DFT Considering the exponential factor in the DFT of Equation 11.11, the periodic extension of the sequence of frequency components $F(k)$ is periodic with period N. This is shown by writing

$$\begin{aligned} F_{k+N} &= \sum_{n=0}^{N-1} f_n e^{-i2\pi n(k+N)/N} \\ &= \sum_{n=0}^{N-1} f_n e^{-i2\pi nk/N} e^{-i2\pi n} \\ &= \sum_{n=0}^{N-1} f_n e^{-i2\pi nk/N} = F_k \end{aligned} \tag{11.18}$$

because n is an integer.

When $f(t)$ is real, we expect that the real part of F_k is even and the imaginary part is odd, using $k = N/2$ as the origin. Remember from the previous discussions that the frequencies for $k > N/2$ are actually negative frequencies so that

$$\bar{F}_k = F_{N-k}$$

when $f(t)$ is real and N is an even number, where $\bar{F}_k$ is the complex conjugate of the term F_k. We say that the terms in the DFT exhibit *conjugate symmetry* about the point $N/2$ in this case. Thus, computation can be simplified if the DFT is to be computed directly from the definition.

☐ EXAMPLE 11.7 ***DFT Example***

Consider the sequence $f(n) = \{1, 1, 0, 0, 0, 0, 0, 0\}$, for which the DFT reduces to the sum

$$F(k) = \sum_{n=0}^{1} f(n)e^{-i2\pi nk/8} = 1 + \exp(-i\pi k/4) \qquad k = 0, 1, \ldots, 7. \quad \textbf{(11.19)}$$

The terms are thus

$$\begin{array}{rcccc} F_0 & = & 1+1 & = & 2 \\ F_1 & = & 1+\exp(-i\pi/4) & = & 1.7071 - 0.7071i \\ F_2 & = & 1+\exp(-i\pi/2) & = & 1 - i \\ F_3 & = & 1+\exp(-i\pi 3/4) & = & 0.2929 - 0.7071i \\ F_4 & = & 1+\exp(-i\pi) & = & 0 \\ F_5 & = & 1+\exp(-i\pi 5/4) & = & 0.2929 + 0.7071i \\ F_6 & = & 1+\exp(-i\pi 3/2) & = & 1 + i \\ F_7 & = & 1+\exp(-i\pi 7/4) & = & 1.7071 + 0.7071i. \end{array}$$

The argument of each exponential term is a multiple of $-\pi/4$. The numerical value can be computed from Equation 11.13, using $\pm 1/\sqrt{2} \approx \pm 0.7071$ as an approximation.

Notice that there is *conjugate symmetry* about $N/2 = 4$; that is, since $F_{N-k} = \bar{F}_k$,

$$F_5 = \bar{F}_3, \quad F_6 = \bar{F}_2, \quad F_7 = \bar{F}_1.$$

☐

WHAT IF? Compare the result of Example 11.7 with the results of the MATLAB script

```
>>ft=[1,1,0,0,0,0,0,0]
>> Fw=fft(ft)
```

What conclusions do you draw about the MATLAB **fft** command with respect to calculating the DFT?

Four-Point DFT It is convenient to write Equation 11.11 for the Fourier component F_k in matrix form to explore the characteristics of the DFT. We will consider the expansion of Equation 11.11 with $N = 4$ to show the pattern. Expanding Equation 11.11 leads to the result

$$\begin{array}{rcl} F_0 & = & f_0W_4^0 + f_1W_4^0 + f_2W_4^0 + f_3W_4^0 \\ F_1 & = & f_0W_4^0 + f_1W_4^1 + f_2W_4^2 + f_3W_4^3 \\ F_2 & = & f_0W_4^0 + f_1W_4^2 + f_2W_4^4 + f_3W_4^6 \\ F_3 & = & f_0W_4^0 + f_1W_4^3 + f_2W_4^6 + f_3W_4^9. \end{array} \quad \textbf{(11.20)}$$

In Equation 11.20, $N = 4 = 2^2$, so the exponential terms can be written as $W_4^{nk} = W_4^{nk \bmod 4}$ according to Equation 11.17. Thus, we compute

$$\begin{aligned} W_4^0 = W_4^4 &= 1 \\ W_4^1 = W_4^9 &= -i \\ W_4^2 = W_4^6 &= -1 \\ W_4^3 &= i. \end{aligned} \tag{11.21}$$

using the result of Equation 11.16. In Equation 11.20, the terms $W_4^5 = W_4^1$, $W_4^7 = W_4^3$, and $W_4^9 = (W_4^3)^3 = W_4^1$.

Rewriting Equation 11.20 in matrix terms, considering the results defined in Equations 11.21, yields

$$\begin{bmatrix} F_0 \\ F_1 \\ F_2 \\ F_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \end{bmatrix}. \tag{11.22}$$

***N*-Point DFT** Using the approach used for the four-point problem, the DFT for N points can be written

$$\begin{bmatrix} F_0 \\ F_1 \\ F_2 \\ \vdots \\ F_{N-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & W_N^1 & W_N^2 & \cdots & W_N^{N-1} \\ 1 & W_N^2 & W_N^4 & \cdots & W_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & W_N^{N-1} & W_N^{2(N-1)} & \cdots & W_N^{(N-1)^2} \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ \vdots \\ f_{N-1} \end{bmatrix}. \tag{11.23}$$

The Fourier equation (11.23), which can be written as

$$\mathbf{F} = \left[W_N^{nk}\right] \mathbf{f}, \tag{11.24}$$

performs the analysis of the signal represented by $\mathbf{f}$. The inverse relationship represents the synthesis of a signal from its Fourier components. As you are asked to show in Problem 11.10, the $N \times N$ Fourier matrix in Equation 11.23 has orthogonal columns. The inverse matrix is formed by taking the conjugate transpose and dividing by N.

Notice that N^2 multiplications and $N(N-1)$ additions are performed in the direct matrix computation of Equation 11.23, and many of these involve complex values. One complex multiplication requires four real multiplications and three real additions. The number of direct calculations can be reduced if we let $N = 2^m$, where m is an integer, since there are many repeated values among the real and imaginary parts in Equation 11.13.

☐ EXAMPLE 11.8 ***Four-point DFT***

Let $f(n) = [1, 2, 3, 4]$ so that the four-point DFT from Equation 11.22 is

$$\begin{bmatrix} F_0 \\ F_1 \\ F_2 \\ F_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} = \begin{bmatrix} 10 \\ -2+2i \\ -2 \\ -2-2i \end{bmatrix}. \tag{11.25}$$

Formulation of the DFT matrix from direct application of the DFT definition has a number of computational drawbacks. Obviously, it would not be very efficient to store the $N \times N$ matrix if N is a large number. In any case, the direct computation method is not considered computationally efficient, and it is desirable to find methods of speeding up the computer calculation by reducing the number of operations. When comparing computational methods, only the multiplications are usually counted since they are by far the most time-consuming compared to the time for addition, subtraction, and retrieval and storage of data values.

☐

FFT ALGORITHMS

As we have seen, the direct calculation of the N-point DFT requires N^2 complex multiplications. A number of researchers have developed algorithms called *fast Fourier transform algorithms* that more efficiently compute the DFT. These algorithms essentially reduce the problem of calculating an N-point DFT to that of calculating many DFTs for smaller values of N. Although there are numerous algorithms, the algorithms generally perform an efficient DFT by taking advantage of the following:

1. The symmetry and periodicity of $W_N^{nk} = \exp(-i2\pi nk/N)$.
2. Careful choice of N or the factors of N.
3. Separate calculation on samples of even index and odd index.
4. Performing calculations *in place*.

Radix-2 FFT The choice $N = 2^m$ for the number of points, where m is an integer, leads to the *radix*-2 FFT. We will present an intuitive development (without proof) of the fast Fourier transform (FFT) algorithm for the particular case that the number of points is a power of two.[3] The key to the efficiency of the FFT algorithm we discuss is the factorization of an N-point transform into two $N/2$-point transforms. Even with the computation necessary to assemble the two transforms into the N-point transform, considerable savings in computation time results, particularly when N is a large number. In fact, $N^2/2$ multiplications are required, which is half the number needed in the original problem. The reduction

[3]For a more rigorous approach and for a description of other algorithms, the textbook by Brigham listed in the Annotated Bibliography for this chapter is highly recommended.

procedure can be repeated until two-point transforms are computed. In Example 11.9, we will show a particular case of the FFT known as a *radix-2* transform for $N = 4$ points.[4]

EXAMPLE 11.9 ***Two-and Four-Point FFT***

Consider the case of the *two-point* DFT

$$F_k = \sum_{n=0}^{1} f_n e^{-i2\pi nk/2} = \sum_{n=0}^{1} f_n W_2^{nk} \qquad k = 0, 1.$$

Because $W_2^{0k} = 1$ and $W_2^{1k} = e^{-i\pi k} = (-1)^k$, the two-point result is

$$\begin{aligned} F_0 &= f_0 + f_1 \\ F_1 &= f_0 - f_1. \end{aligned}$$

In matrix form, the two-point transform is

$$\begin{bmatrix} F_0 \\ F_1 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \end{bmatrix}. \tag{11.26}$$

To factor the DFT for the FFT algorithm, the DFT is written in terms of the even and odd indices of $f(n)$. We assume that $N = 2^m$ and show the case for $N = 4$ written in terms of two, two-point transforms. This algorithm is described as a *decomposition-in-time* or *decimation-in-time*, *radix-2* FFT.

The kth term in the four-point transform is

$$\begin{aligned} F_k &= \sum_{n=0}^{3} f(n) e^{-i2\pi nk/4} \\ &= f_0 W_4^{0k} + f_1 W_4^{1k} + f_2 W_4^{2k} + f_3 W_4^{3k}, \end{aligned} \tag{11.27}$$

where $k = 0, 1, 2, 3$. Using the results in Equation 11.21 to simplify W_4^{nk} leads to the expression

$$F_k = [f_0 + f_2(-1)^k] + [f_1 + f_3(-1)^k] W_4^{1k} \tag{11.28}$$

for $k = 0, 1, 2, 3$. The idea is to write this expression in terms of two-point transforms. Define the even and odd points as

$$\begin{aligned} f_{en} &= f_{2n}, \quad n = 0, 1 \\ f_{on} &= f_{2n+1}, \quad n = 0, 1 \end{aligned}$$

so that

$$F_k = [f_{e0} + f_{e1}(-1)^k] + [f_{o0} + f_{o1}(-1)^k] W_4^{1k}, \tag{11.29}$$

where the factors in brackets are two-point DFTs. If we define

$$\begin{aligned} F_{eq} &= f_{e0} + f_{e1}(-1)^q \qquad q = 0, 1 \\ F_{oq} &= f_{o0} + f_{o1}(-1)^q \qquad q = 0, 1, \end{aligned} \tag{11.30}$$

[4] An illuminating article that covers the history and derivation of the FFT is "A Guided Tour of the Fast Fourier Transform," by G.D. Bergland in the *IEEE Spectrum*, Vol. 6, July 1969, pp41–52.

you are asked to show in Problem 11.13 that the four-point FFT can be written as

$$\begin{aligned} F_0 &= F_{e0} + F_{o0} \\ F_1 &= F_{e1} - F_{o1}(i) \\ F_2 &= F_{e0} - F_{o0} \\ F_3 &= F_{e1} + F_{o1}(i). \end{aligned} \tag{11.31}$$

In the final expression, the values $W_4^{1k} = (-i)^k$ for each k were used.

The conclusion is that the four-point DFT can be written as follows

$$\begin{aligned} [\text{Four-point DFT of } f(n)] &= [\text{Two-point DFT of } f_e(n)] \\ &+ W_4^{1k}[\text{Two-point DFT of } f_o(n)]. \end{aligned}$$

□

The result in Example 11.9 can be generalized for radix-2 FFT algorithms to show that an N-sample DFT can be written as the sum of two $N/2$-sample DFTs formed from the even- and odd-indexed samples of the original sequence. When $N = 2^m$, this process of decomposition or decimation can be continued until the N-point FFT can be computed by properly combining $N/2$ two-point transforms. The reader should see the references in the Annotated Bibliography for this chapter for more information about this important algorithm.

11.7 REINFORCEMENT EXERCISES AND EXPLORATION PROBLEMS

In these problems, do the computations by hand unless otherwise indicated and then check those that yield numerical or symbolic results with MATLAB.

REINFORCEMENT EXERCISES

P11.1. Periodicity of the DFT Show that the DFT defined in Equation 11.5 is periodic. Thus, for an arbitrary integer r,

$$F\left(\frac{r+N}{NT_s}\right) = F\left(\frac{r}{NT_s}\right).$$

P11.2. Approximation to Fourier transform Consider the Fourier transform of a function

$$F(\omega) = \int_{-\infty}^{\infty} f(t)e^{-i\omega t}\,dt$$

applied to a function sampled at intervals $t = nT_s$. Assume that the function is defined only for $t > 0$ and further that $f(nT_s)$ is almost zero for $n \geq N$ when N is a sufficiently large number. By

integrating over one sampling interval and summing the results, show that the discrete approximation to the Fourier transform is

$$F(k\Omega) \approx T_s \sum_{n=0}^{N-1} f(nT_s) e^{-ink\Omega T_s} \quad \Omega = \frac{2\pi}{NT_s},$$

which agrees with the definition of the DFT in this chapter.

Hint: Assume that the sampling interval is sufficiently small that exponentials not involving k or n can be expanded in a Taylor series with only two terms.

P11.3. DFT conditions Determine the functions $f(t)$ that have the properties that meet the criteria for exact DFT representation:

a. $f(t)$ is periodic and the spectrum is bandlimited.

b. $f(t)$ can be sampled at a rate that yields more than two samples per cycle of the highest frequency in $F(f)$.

c. $f(t)$ can be sampled exactly over one period or an integral number of periods.

Hint: Consider the answer in terms of the Fourier series expansion of the function.

P11.4. Direct calculation of DFT A good way to understand the DFT is to calculate the DFT for a simple function by hand. Determine the DFT for the discrete function

$$f_n = \begin{cases} 1, & 0 \le n \le 4, \\ 0, & 5 \le n \le 9. \end{cases}$$

Plot the amplitude and phase of the transform and describe the symmetries in the transform.

P11.5. Aliasing Given that the sampling rate is 10 samples per second for the following signals, are they aliased?

a. $f(t) = \cos(2\pi t)$

b. $f(t) = \cos(4\pi t)$

c. $f(t) = \cos(10\pi t)$

d. $f(t) = \cos(12\pi t)$

e. $f(t) = \cos(14\pi t)$

P11.6. MATLAB discrete signal Create a 256-point function with a 25% duty cycle of ones (64 points) and compute the Fourier spectrum and plot it.

a. Try using the MATLAB command **kron** to create the discrete pulse.

b. Compute the maximum error when compared with the Fourier transform of a pulse.

P11.7. MATLAB FFT Compare the Fourier spectrum for the signal

$$f(t) = \begin{cases} \sin\left(\frac{\pi t}{4}\right), & \text{for } 0 \le t \le 4, \\ 0, & \text{for } t < 0 \text{ or } t > 4, \end{cases}$$

sampled every T_s seconds at N points in the following cases:

a. $N = 16$ and $T_s = 1$ second;

b. $N = 64$ and $T_s = 1$ second;

c. $N = 128$ and $T_s = 0.5$ second.

P11.8. MATLAB FFT resolution Numerical techniques usually require some trial and error. For example, the proper number of sample points N and the sampling period of the FFT are not generally known for an arbitrary signal. Assume that a signal is given analytically and various values of sampling times and sampling period are to be tried.

One method is to choose a reasonable value for the number of samples N and compute the spectrum of the signal. Then, let $N1 = 2 * N$ and recompute. This is repeated until the spectrum of two subsequent calculations are very close. Start with 64 samples.

Assume the function

$$\sin(0.6\pi t) + 0.5\sin(0.64\pi t),$$

sampled every second for 512 points, is to be analyzed.

a. Compute the magnitude of the FFT for 64, 128, 256, and 512 points and plot the results.

b. Analyze each plot and compare the frequency resolution for each.

c. For the final plot, if there appear to be several distinct frequencies, pick off the peaks and print the frequencies that correspond.

Hint: Use MATLAB command **ginput** if you have a mouse.

P11.9. MATLAB FFT leakage Compare the FFT results for the functions:

$$f(t) = \sin(2\pi 20t) \text{ and } g(t) = \sin(2\pi 19t)$$

by sampling 64 points with a time resolution of 1/128 second.

a. Plot the magnitude of each FFT on a subplot to compare the results.

b. Explain the extra frequency components in each result.

P11.10. DFT matrix Show that the inverse of the $N \times N$ Fourier matrix of Equation 11.23 is the conjugate transpose divided by N. Therefore, the algorithm to compute the DFT can be used, with minor modifications, to compute the inverse DFT.

P11.11. DFT 8-point matrix Work out the 8-point DFT matrix from Equation 11.23.

P11.12. Four-point FFT matrix Show that the four-point FFT as in Example 11.9 can be written as the product of matrices as

$$\begin{bmatrix} F_0 \\ F_1 \\ F_2 \\ F_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & -i \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 0 & i \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \end{bmatrix}.$$

Explain the meaning of each matrix.

P11.13. FFT factorization Write out the terms in Equation 11.29 and show that the four-point FFT can be written in terms of 2 two-point FFTs.

P11.14. FFT four-point factorization Write out the terms in Equation 11.31 and show that the result is the same as Equation 11.22.

P11.15. Four point FFT Using the decomposition described in Example 11.9, compute the DFT of the function

$$f_n = [1, 2, 3, 4].$$

EXPLORATION PROBLEMS

P11.16. MATLAB FFT phase Given the function

$$f(t) = \begin{cases} e^{-t}, & t \geq 0, \\ 0, & t < 0, \end{cases}$$

as treated in Example 11.3, compute the phase of the function and compare the results with the actual value. Use the MATLAB command **fftshift** to plot the two-sided spectrum from $-F_{\max}$ to $F_{\max}$, where the frequency range will be determined by the sampling interval of $f(t)$ and the number of points taken.

Hint: Try to approach the Fourier transform result (at least near the origin) by adding zeros to increase the period of the time function as much as possible. This method is called *zero padding* to lengthen the sequence of sampled points. However, zero padding does not add any information about the signal being analyzed.

P11.17. MATLAB discrete spectrum plot Write an M-function to plot a discrete spectrum versus frequency in hertz or radians per second. The function call should be:

```
function plotdscf(f,F,xunit)
% CALL: plotdscf(f,F,xunit); plot a discrete spectrum [f F]
% Inputs to function are
%    f   - frequencies
%    F   - spectral values
%  xunit - units of frequency (Hz or rad/s)
```

P11.18. FFT Investigation (Optional) Consider the unit step function, $U(t) = 1$ for $f > 0$, defined in Chapter 6. Its ordinary Fourier transform does not exist (because the signal would have infinite energy) but a *generalized transform* can be defined. The transform is written

$$\mathcal{F}[U(t)] = \pi\delta(\omega) + \frac{1}{i\omega},$$

where $\delta(\omega)$ is the *unit impulse function*. Attempt to compute the DFT of the unit step function and explain the result. Use the inverse FFT (**ifft**) on the result and reconstruct the time function. Compare the results with the Laplace transform of the unit step function.

P11.19. MATLAB DFT matrix Write a MATLAB script with N as input to compute the N-point DFT matrix $\left[W_N^{nk}\right]$.

P11.20. MATLAB Script Write a MATLAB script that computes the DFT directly from the definition with the inputs

a. N, the number of sample points;

b. f, the vector of sample points.

Test the routine on the sequence $f = \{1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\}$.

P11.21. MATLAB FFT Consider the Fourier transform of the function

$$P(t) = \begin{cases} 1, & 0 \le t \le 2\text{s}, \\ 0, & \text{otherwise.} \end{cases} \tag{11.32}$$

Compare the FFT result with $|F(\omega)|$ and find the maximum error for various values of N, the number of samples; $N = 16, 128, 1024$. For each value of N, plot $|F(\omega)|$ and the FFT result on the same graph.

Remember you are comparing the Fourier transform of a time signal with the FFT results.

P11.22. MATLAB Script comparing DFT and FFT Write a MATLAB script to compare the time of the DFT routine from the previous problem and the MATLAB FFT routine. The result should be a table of the form

```
Comparison of DFT and FFT times
N     DFT     FFT     FFT/DFT     log2 N/N
8     Time1     etc.
64
128
1024
```

In the table, N is the number of sample points and the times are in seconds (or fractions of a second). Test the functions:

a. $f = \{1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ \ldots\}$.

b. A random sequence of N numbers.

P11.23. MATLAB Script comparing Fourier transform and FFT Compute the Fourier transform of the function

$$f(t) = e^{-t}\cos 100t.$$

Then, plot the following on the same figure:

a. Plot the function from $0 \le t \le 10$.

b. Plot the Fourier transform amplitude and phase versus radians/second over the range [0, 314];

c. Plot the amplitude and phase of the FFT result.

d. Comment on the error in the amplitude and phase as determined by the FFT.

11.8 ANNOTATED BIBLIOGRAPHY

1. Abramowitz, Milton, and Irene A. Stegun, *Handbook of Mathematical Functions*, Dover Publications, Inc., New York, 1972. *This handbook contains many formulas, graphs and tables of mathematical functions. In particular, the solutions to the equation* $x = \tan x$ *are tabulated.*

2. Bracewell, Ronald N., *The Fourier Transform and Its Applications*, 2nd edition, revised, McGraw-Hill Book Company, New York, 1986. *An excellent book that covers many applications of the Fourier transform as well as a number of other transforms.*

3. Brigham, E. Oran, *The Fast Fourier Transform and its Applications*, Prentice Hall, Englewood Cliffs, NJ, 1988. *A very readable treatment of Fourier techniques with emphasis on the FFT. There is also an extensive bibliography.*

4. Burrus, C. Sidney, J. H. McClellan, A. V. Oppenheim, T. W. Parks, R. W. Schafer, and H. W. Schuessler, *Computer-Based Exercises for Signal Processing Using MATLAB*, Prentice Hall, Englewood Cliffs, NJ, 1994. *The book contains a collection of computer exercises about signal processing using MATLAB.*

5. Cooley, James W., P. Lewis, and P. Welch, "Application of the Fast Fourier Transform to Computation on Fourier Integrals, Fourier Series, and Convolution Integrals," AU-15, No. 2, June 1967. *This paper discusses the use of the FFT and the errors involved in various approximations.*

6. Hamming, Richard W., *Numerical Methods for Scientists and Engineers*, McGraw-Hall Book Company, New York, 1973. *An interesting treatment of numerical techniques with a number of personal insights from the author. The book is particularly notable for dividing numerical analysis into classical analysis (polynomials) and modern analysis (Fourier methods).*

11.9 ANSWERS

11.1 Periodicity of the DFT For an arbitrary integer r, the DFT component at index $k = r + N$ is

$$F\left(\frac{r+N}{NT_s}\right) = \sum_{n=0}^{N-1} f(nT_s)e^{-i2\pi n(r+N)/N}.$$

Considering the exponential factor, we find

$$e^{-i2\pi n(r+N)/N} = e^{-i2\pi nr/N}e^{-i2\pi n} = e^{-i2\pi nr/N}$$

since

$$e^{-i2\pi n} = \cos 2\pi n - i \sin 2\pi n = 1$$

for n an integer. Thus,

$$F\left(\frac{r+N}{NT_s}\right) = F\left(\frac{r}{NT_s}\right),$$

which shows that the DFT defined in Equation 11.5 is periodic with a period N.

11.3 DFT conditions The only functions $f(t)$ for which all the ideal DFT conditions hold are those functions with finite Fourier series.

11.5 Aliasing Given that the sampling rate is 10 samples per second, the sampling theorem requires the maximum frequency in the signal to be less than $f_{\max} = 10/2 = 5$ hertz to avoid aliasing.

a. $\cos(2\pi t)$, $f = 1$ Hz, not aliased;

b. $\cos(4\pi t)$, $f = 2$ Hz, not aliased;

c. $\cos(10\pi t)$, $f = 5$ Hz, aliased;

d. $\cos(12\pi t)$, $f = 6$ Hz, aliased;

e. $\cos(14\pi t)$, $f = 7$ Hz, aliased.

11.9 FFT leakage The 20-hertz sine wave is sampled over a number of periods with a DFT spectrum resolution of 2 hertz. If a sufficient number of sampling points are taken, the spectrum appears as a "spike" at 20 hertz (with a component at negative frequencies). The 19-hertz signal is truncated by sampling but not over an integral number of periods. The energy at 19 hertz "leaks" into adjacent frequencies, and the spectrum near 19 hertz appears broadened. Another way to view this is to recognize that there is no component of this DFT spectrum at 19 hertz. Energy at 19 hertz will appear at all the frequencies in the spectrum, with the largest values at 18 and 20 hertz.

11.11 DFT 8-point matrix For $N = 8$, the terms of $\exp(i2\pi m/N)$, $m = 0, \ldots 7$ are as follows:

$$\begin{aligned}
W_8^0 &= 1 \\
W_8^1 &= (1-i)/\sqrt{2} \\
W_8^2 &= -i \\
W_8^3 &= -(1+i)/\sqrt{2} \\
W_8^4 &= -1 \\
W_8^5 &= -(1-i)/\sqrt{2} \\
W_8^6 &= i \\
W_8^7 &= (1+i)/\sqrt{2}.
\end{aligned}$$

Since $W_8^{nk} = W_8^{nk \bmod 8}$, relationships such as $W_8^0 = W_8^8 = W_8^{16} = W_8^{24}$, etc., reduce the matrix to the form

$$\left[W_8^{nk}\right] = \begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & W_8^1 & W_8^2 & W_8^3 & W_8^4 & W_8^5 & W_8^6 & W_8^7 \\
1 & W_8^2 & W_8^4 & W_8^6 & W_8^0 & W_8^2 & W_8^4 & W_8^6 \\
1 & W_8^3 & W_8^6 & W_8^1 & W_8^4 & W_8^7 & W_8^2 & W_8^5 \\
1 & W_8^4 & W_8^0 & W_8^4 & W_8^0 & W_8^4 & W_8^0 & W_8^4 \\
1 & W_8^5 & W_8^2 & W_8^7 & W_8^4 & W_8^1 & W_8^6 & W_8^3 \\
1 & W_6^6 & W_8^4 & W_8^2 & W_8^0 & W_8^6 & W_8^4 & W_8^2 \\
1 & W_8^7 & W_8^6 & W_8^5 & W_8^4 & W_8^3 & W_8^2 & W_8^1
\end{bmatrix}.$$

This can be further simplified by using the relationships

$$W_8^4 = -W_8^0 \quad W_8^5 = -W_8^1 \quad W_8^6 = -W_8^2 \quad W_8^7 = -W_8^3.$$

11.13 FFT factorization By direct expansion of Equation11.29, the four-point transform is

$$\begin{array}{rcll} F_0 & = & f_{e0} + f_{e1} + [f_{o0} + f_{o1}] & = & F_{e0} + F_{o0} \\ F_1 & = & f_{e0} - f_{e1} + [f_{o0} - f_{o1}](-i) & = & F_{e1} - F_{o0}(i) \\ F_2 & = & f_{e0} + f_{e1} + [f_{o0} + f_{o1}(-1)] & = & F_{e0} - F_{o0} \\ F_3 & = & f_{e0} - f_{e1} + [f_{o0} + f_{o1}](i) & = & F_{e1} + F_{o1}(i). \end{array}$$

11.15 Four point FFT Verify your answer with the result of Example 11.8.