SVM-3

Programming and Analysis for Digital Time Series Data

The Shock and Vibration Information Center
United States Department of Defense

SVM-3

Programming and Analysis for Digital Time Series Data

Loren D. Enochson
Robert K. Otnes
Measurement Analysis Corporation

1968

The Shock and Vibration Information Center
United States Department of Defense

The Shock and Vibration Information Center

This series is published by the Shock and Vibration Center, Naval Research Laboratory, Washington D.C.

William W. Mutch, Head

Henry C. Pusey

Rudolph H. Volin

Katherine G. Jahnel, Administrative Secretary

For sale by the Navy Publication and Printing Service Office, Naval District Washington, Bldg. 157-2, Washington Navy Yard, Washington D.C. 20390. Price $8.00 by check or money order made payable to the Treasurer of the United States.

The Shock and Vibration Monograph Series

SVM-1 Random Noise and Vibration in Space Vehicles – *Lyon*

SVM-2 Theory and Practice of Cushion Design – *Mustin*

SVM-3 Programing and Analysis for Digital Time Series Data – *Enochson and Otnes*

Now in preparation:

Dynamics of Rotating Shafts – *Loewy and Piarulli*

Principles and Techniques for Shock Data Analysis – *Kelly and and Richman*

Optimum Shock and Vibration Isolation and Absorption – *Sevin and Pilkey*

The Influence of Damping in Vibration Isolation – *Ruzicka*

Contract No. N0014-67-C-0073

FOREWORD

Time series data, also called random data or stochastic data, consist of any data sample, regardless of its origin, which cannot be represented by explicit mathematic relationships, but which can be effectively described in terms of probability statements and statistical averages. This type of data occurs routinely in such diverse application areas as vibration, acoustics, communications, control, oceanography, seismology, biomedical research, and structural dynamics.

There are three different problem areas associated with time series: *mathematics,* the mathematical theory of time series; *optimization,* the design of optimal systems involving time series; and *measurement,* the practical measurement and analysis of time series. During the last twenty-five years, a great deal of research and literature has been devoted to the areas of mathematics and optimization, while the area of measurement has received relatively little attention. Unfortunately, the theory of optimization has found few practical applications because the measurement of the required data was not practical.

Recently, the measurement problem has improved. It is now possible to acquire software systems and hardware systems that make the measurement of time series a practical tool for the engineer. In addition, there have appeared several very useful textbooks: J. S. Bendat and A. G. Piersol, *Measurement and Analysis of Random Data,* 1966; G. M. Jenkins and D. G. Watts, *Spectral Analysis and its Applications,* 1968; and E. A. Robinson, *Multichannel Time Series Analysis with Digital Computer Programs,* 1967.

This monograph makes a significant contribution to the measurement problem. It is a comprehensive study of the state of the art in digital computer programming for, and practical analysis of, digital time series data. The authors of this monograph have had a great deal of practical experience in this technology, experience that is valuable to others. This monograph also complements the text by Bendat and Piersol.

At the present time, there is much active research in the area of measurement and analysis of time series data, and much experience has still to be gained. It is also felt that the number of people using these techniques will increase very rapidly for quite a few years.

RUBERT B. STREETS, JR.

Electrical Engineering Department
The University of Calgary
Alberta, Canada
20 October 1968

PREFACE

This monograph has grown out of the experience of the authors in the field of digital data analysis and thus represents a viewpoint biased by that experience. On the other hand, the material presented here has been employed for the most part in practical data analysis environments and therefore has the advantages gained from trial by fire.

This monograph cannot be considered as an introductory text on the subject. Rather, the authors have attempted to include as much material as was available to them at the time of writing. Much of the resulting work may be found to be too advanced by many potential readers. To alleviate this difficulty, Chapter 1 has been included as a bridge between a knowledge of elementary statistics and some of the more difficult concepts that are required. Examples are introduced throughout which should clarify the text. The theoretical aspects of many topics are discussed more thoroughly by Hannan.*

Proofs of some important theorems are discussed, but the reader should not be deceived: The authors are not mathematically rigorous in their presentation, and there are many sticky mathematical points to which only a passing reference has been made.

Time and space (this monograph was originally viewed as about 100 pages) did not permit the inclusion of certain time series data analysis topics which are of great interest. Chief of these is prewhitening. It is felt that the application of prewhitening depends upon an understanding of the physics of the phenomenon being investigated, so that an adequate discussion of it would have required delving into meteorology, structural mechanics, seismology, etc., which are far afield from the intended direction of the monograph. Another major omission is a discussion of current procedures for analyzing transient or short duration data. However, this will be rectified with the issuance of another Shock and Vibration Information Center monograph in this series on that topic by R. D. Kelly and G. Richman.

The authors would like to thank W. R. Forlifer, H. Holtz, Merval Oleson, Prof. R. B. Streets, Jr. and Prof. A. R. Stubberud for their valuable comments and review of the monograph and Sally, Kathy, Gertrude, Helene, Kaz and Henry for their superb preparation of the manuscript.

LOREN D. ENOCHSON
ROBERT K. OTNES

Los Angeles, California

*E. J. Hannan, *Time Series Analysis*, John Wiley & Sons, Inc., New York, 1960.

CONTENTS

CHAPTER 1
PRELIMINARY CONCEPTS

1.1 Time Functions

Before the data reduction of time histories can be discussed in detail, a mathematical foundation must be laid for the procedures to be used. As such mathematical details are not the aim of this book, many proofs in this chapter will be sketched or completely omitted. If the reader is troubled by this, he should obtain fuller explanations in texts such as that by Hannan [1].

Data reduction involves the processing of data records, usually referred to as time histories. A time history is simply the output of a measuring instrument recorded on some medium such as graph paper, punched paper tape, or magnetic tape. A record made of the daily closing price of a particular stock would be an example of a time history. In terms of the mathematical definitions usually associated with time histories, the price of the stock would be the dependent variable, whereas the time at which the price was quoted would be an independent variable. In the work that follows, the most common independent variable is time, usually expressed in seconds. The next most common independent variable is frequency. There are numerous other possible independent variables, such as position or velocity coordinates, but they are relatively uncommon in practice. However, the use of the word time for the independent variable in a recording of data should not be considered as restrictive. The techniques discussed are perfectly general, and the substitution of distance or depth, for example, in place of time is often done and only requires a consistent rearrangement of terms.

The precise definition of a function is not required for the ensuing discussion. Most of the functions employed may be thought of simply as processes that assign a single value to the dependent variable for each value of the independent variable. There are exceptions; functions will be used that have multiple dependent or independent variables.

The time histories to be discussed have a number of constraints placed on them by the nature of their origin. The four main limitations are

1. Record length is finite,
2. Range of the data is restricted,
3. Data are sampled, and
4. Data are discrete.

Example 1.1 A recording digital voltmeter is used to monitor a certain experiment. A transducer is used to convert the physical quantity being observed into an electrical voltage. This voltage is the input to a voltmeter which records ±100 V in one-digit steps. It samples once per second, and the experiment

consists of taking 100 samples–thus, the record length is limited to 100 sec. Values above 100 V or below −100 V may not be recorded, occurrences between the 1-sec samples are not observed, and fluctuations smaller than 1 V may be missed.

These values set definite limitations on what can be observed with such a system. These limitations, as will be seen in succeeding sections, carry over to analyses done in a frequency sense.

Time functions will be denoted in the following manner:

$x(t)$ = continuous function,

and

x_i = sampled function.

Usually the sampling increment will be uniform and denoted by Δt. In this case, we could write that

$$x_i = x(i\Delta t). \tag{1.1}$$

The function x could be any physical quantity measured during the course of a test and may be thought of as any time in history being examined at some given point in the data reduction process. If two time histories must be considered at the same time, the first will be denoted by x and the second by y.

Note that the following convention is used when discussing time histories: a single letter x refers to the function or process in general. The symbol $x(t)$, on the other hand, is the particular numerical value of the function x at time t.

1.2 Mean Value

A host of problems arise when the time histories under consideration are random processes. To avoid continual rediscussion of these difficulties, an assumption will be made: unless explicitly stated otherwise, the time histories, when they are random processes, are stationary and ergodic. This topic is deferred to Chapters 8 and 9, where the definition of these terms is more appropriate.

Under the above assumptions, the time average (mean value) of a function is equivalent to its ensemble average and may be used in place of it. The time average $\bar{x}$ is defined for continuous data by

$$\bar{x} = \lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} x(t)\,dt. \tag{1.2}$$

Of course, only a finite portion of a record is available. Suppose the record starts at time $t = a$ and ends at time $t = b$. Then the time average over this piece of data would be

$$\bar{x} = \frac{1}{b-a}\int_a^b x(t)\,dt. \tag{1.3}$$

If the data are sampled, then Eq. (1.3) will take the form

$$\bar{x} = \frac{1}{N}\sum_{i=k}^{N-1+k} x_i, \tag{1.4}$$

where $k\Delta t = a$, and $(N-1+k)\Delta t = b$.

For convenience, k is usually taken to be zero, so Eq. (1.4) is usually written as

$$\bar{x} = \frac{1}{N}\sum_{i=0}^{N-1} x_i. \tag{1.5}$$

Subscripts could have been added to the last three definitions of $\bar{x}$ to indicate that the average was taken over the interval (a, b). However, $\bar{x}$ will always be a time average over the total record being discussed at the point at which it is used. Deviations from this convention will be duly noted when they occur.

1.3 Variance

The variance of a time history under the assumptions discussed in the previous section is s_x^2 where

$$s_x^2 = \lim_{T\to\infty} \frac{1}{2T}\int_{-T}^{T} [x(t)-\bar{x}]^2\,dt. \tag{1.6}$$

For finite record lengths, this is

$$s_x^2 = \frac{1}{t_2-t_1}\int_{t_1}^{t_2} [x(t)-\bar{x}]^2\,dt. \tag{1.7}$$

The standard deviation of x is the positive square root of the variance, s_x^2. Devices that compute s_x using a mechanization of Eq. (1.7) are called true root-mean-square (rms) meters. Other types of meters exist that give rms readings only for certain specific periodic functions, usually sine waves.

For sampled data, the definition of s_x^2 is

$$s_x^2 = \frac{1}{N-1} \sum_{i=0}^{N-1} (x_i - \bar{x})^2. \tag{1.8}$$

The mean square value of x in the continuous case, written Ψ_x^2, is

$$\Psi_x^2 = \lim_{T \to \infty} \frac{1}{2T} \int_{-T}^{T} x^2(t)\, dt. \tag{1.9}$$

As with the variance, both a record of finite length and discrete versions will be employed where necessary.

Example 1.2 Suppose that x is a sinusoid

$$x(t) = A \sin (2\pi f_c t + \phi), \quad -\infty < t < \infty. \tag{1.10}$$

Then

$$s_x^2 = \lim_{T \to \infty} \frac{1}{T} \int_{-T/2}^{T/2} A^2 \sin^2 (2\pi f_c t + \phi)\, dt$$

$$= A^2 \lim_{T \to \infty} \frac{1}{T} \int_{-T/2}^{T/2} \left[\frac{1 - \cos (4\pi f_c t + 2\phi)}{2} \right] dt$$

$$= A^2 \lim_{T \to \infty} \frac{1}{T} \left[\frac{T}{2} + \{\text{sine term} \leqslant 1\} \right] = \frac{A^2}{2}. \tag{1.11}$$

1.4 Fourier Transform

A number of definitions of the Fourier transform are currently in use. They differ in their multiplicative constants. However, the most common form for vibration analysis (and for communication theory) is

$$X(f) = \lim_{T \to \infty} \int_{-T}^{T} x(t)\, e^{-j2\pi ft}\, dt. \tag{1.12}$$

X may be shown to exist if the square of x is integrable in the Lebesgue sense.* As usual, this restriction actually is far broader than it need be for practical

*See Ref. 2 for details.

applications. Also, it is only sufficient rather than necessary. Two notable exceptions to the requirement are the sine and cosine functions, which when squared and integrated, are unbounded. However, as will be seen, it is still possible to talk of their Fourier transforms. It may also be shown that there is a function x_1 such that

$$x_1(t) = \lim_{F\to\infty} \int_{-F}^{F} X(f)\, e^{j2\pi ft}\, df. \tag{1.13}$$

The function x_1 is very much like x. In fact, the equation

$$x(t) - x_1(t) = 0 \tag{1.14}$$

holds for most values of t. Technically, Eq. (1.14) is true almost everywhere. Thus, applying the operation defined by Eq. (1.13) is almost like reversing the procedure given in Eq. (1.12). Because the discrepancies are slight, and even nonexistent in most practical cases, the transformations defined by Eq. (1.13) will be referred to as the inverse Fourier transform.

These operations may be denoted by $\mathcal{F}[x(t)]$ rather than the full integrals. That is,

$$X(f) = \mathcal{F}[x(t)] = \int_{-\infty}^{\infty} x(t)\, e^{-j2\pi ft}\, dt$$

$$x(t) = \mathcal{F}^{-1}[X(f)] = \int_{-\infty}^{\infty} X(f)\, e^{j2\pi ft}\, df. \tag{1.15}$$

The independent variable f will always be expressed in units of hertz (Hz). Occasionally, it will be convenient to use two other related independent variables, ω and λ, where

$$\omega = 2\pi f$$

and

$$\lambda = 2\pi f \Delta t. \tag{1.16}$$

The delta or impulse function δ constantly occurs when dealing with Fourier transforms. It is defined by the following integral equation:

$$y(a_0) = \int_{-\infty}^{\infty} y(a)\, \delta(a - a_0)\, da, \tag{1.17}$$

where y is any bounded function. This function δ may be viewed as one which is arbitrarily tall and narrow but of unit area. Serious problems arise from the use of delta functions. However, it has been shown (Ref. 3) that the calculus of delta functions is consistent. Therefore, they will be used with no exploration of the extensive theory required to discuss them rigorously.

Applying Eq. (1.17) to a few simple expressions rapidly yields some very useful results:

$$\mathcal{F}^{-1}\left[\frac{\delta(f-f_0)+\delta(f+f_0)}{2}\right]=\cos 2\pi f_0 t, \tag{1.18}$$

$$\mathcal{F}^{-1}\left[\frac{\delta(f-f_0)-\delta(f+f_0)}{2j}\right]=\sin 2\pi f_0 t, \tag{1.19}$$

$$\mathcal{F}[\delta(t)]=\mathcal{F}^{-1}[\delta(f)]=1. \tag{1.20}$$

Thus, the Fourier transform of a sine or cosine consists of two delta functions. Also, suppose that the time function x is composed solely of sinusoids with varying amplitudes and phases,

$$x(t)=\sum_{n=1}^{N} A_n \cos(2\pi f_n t+\phi_n). \tag{1.21}$$

Then X consists of a set of $2N$ delta functions, as follows:

$$X(f)=\sum_{n=1}^{N}\frac{A_n}{2}\left[\delta(f-f_n)\,e^{j\phi_n}+\delta(f+f_n)\,e^{-j\phi_n}\right]. \tag{1.22}$$

Example 1.3 Suppose x is defined by

$$x(t)=\begin{cases}1 & 0\leqslant t<\frac{1}{2}\\ 0 & \frac{1}{2}+k\leqslant t<\frac{3}{2}+k\\ 1 & \frac{3}{2}+k\leqslant t<\frac{5}{2}+k \qquad k=0,2,4,\ldots,\end{cases} \tag{1.23}$$

and $x(t)=x(-t)$. This is a square wave of unit height. Another equivalent definition of x would be the trigonometric series obtained from the Fourier analysis of the above,

$$x(t) = \frac{1}{2} + \frac{2}{\pi} \sum_{i=0}^{\infty} \frac{(-1)^i}{1+2i} \cos\left[\pi(2i+1)t\right]. \tag{1.24}$$

The Fourier transform of x is

$$X(f) = \frac{1}{2}\delta(f) + \frac{1}{\pi} \sum_{i=0}^{\infty} \frac{(-1)^i}{1+2i} \left\{ \delta\left[f - \frac{1}{2}(2i+1)\right] + \delta\left[f + \frac{1}{2}(2i+1)\right] \right\}. \tag{1.25}$$

This is shown graphically in Fig. 1.1. The height of the arrow is equal to the coefficient of the corresponding δ function. Because of the symmetry of $x(t)$, $X(f)$ is a real function.

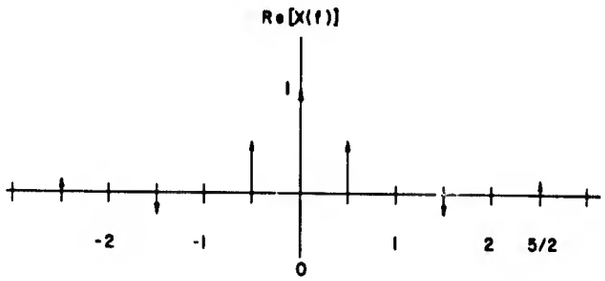


Fig. 1.1. Fourier transform of a square wave with period one unit.

Example 1.4 Suppose x, y, X, and Y are all known. What is the relation between the product of x and y and the functions X and Y? The answer is that

$$\mathcal{F}[x(t)\,y(t)] = \int_{-\infty}^{\infty} X(f')\,Y(f-f')\,df'$$

$$= \int_{-\infty}^{\infty} X(f-f')\,Y(f')\,df'. \tag{1.26}$$

The relation may be derived in the following manner:

$$\mathcal{F}[x(t)\,y(t)] = \mathcal{F}\left\{\mathcal{F}^{-1}[X(f_1)]\mathcal{F}^{-1}[Y(f_2)]\right\}$$

$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} X(f_1)\,Y(f_2)\exp\left[-j2\pi(ft - f_1 t - f_2 t)\right]\,df_1\,df_2\,dt$$

$$\mathcal{F}[x(t)\,y(t)] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} X(f_1)\,Y(f_2)\int_{-\infty}^{\infty} \exp\left[-j2\pi(f-f_1-f_2)t\right]\,dt\,df_1\,df_2$$

$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} X(f_1)\,Y(f_2)\,\delta(f-f_1-f_2)\,df_1\,df_2$$

$$= \int_{-\infty}^{\infty} Y(f_2)\left\{\int_{-\infty}^{\infty} X(f_1)\,\delta(f-f_1-f_2)\,df_1\right\} df_2$$

$$= \int_{-\infty}^{\infty} Y(f_2)\,X(f-f_2)\,df_2. \tag{1.27}$$

The right-hand side of Eq. (1.26) is known as the convolution of X and Y.

Example 1.5 Suppose the function u_T is defined by

$$u_T(t) = \begin{cases} 0 & t < -T \\ 1 & -T \leqslant t \leqslant T \\ 0 & T < t. \end{cases} \tag{1.28}$$

Then the transform is

$$U_T(f) = \int_{-\infty}^{\infty} u_T(f)\,e^{-j\omega t}\,dt = \int_{-T}^{T} e^{-j\omega t}\,dt = -\frac{1}{j\omega}e^{-j\omega t}\Big|_{-T}^{T}$$

$$= -\frac{1}{j\omega}\left[\cos\omega T - j\sin\omega T - \cos(-\omega T) + j\sin(-\omega T)\right] = \frac{2\sin\omega T}{\omega}. \tag{1.29}$$

The transform is shown in Fig. 1.2. This important function will occur repeatedly in subsequent analyses. Again, because of symmetry, $U_T(t)$ is a real function.

Example 1.5 and previous equations can be extended to yield Table 1.1, which shows the relationships of some of the basic functions. Apparently there is a natural correspondence in the Fourier transform; a very narrow function has a very broad transform. When such a narrow function is widened, its Fourier transform becomes narrower.

An interesting result related to the above discussion shows that there are functions which are transformed into themselves. For example,

$$\mathcal{F}\left[\exp -\frac{t^2}{2}\right] = \sqrt{2\pi}\ \exp -\frac{\omega^2}{2}. \tag{1.30}$$

Also

$$\mathcal{F}\left[\sqrt{a}\ \exp -\frac{(at)^2}{2}\right] = \sqrt{\frac{2\pi}{a}}\ \exp -\frac{\omega^2}{2a^2}. \tag{1.31}$$

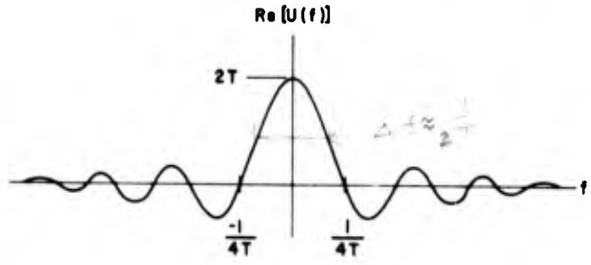
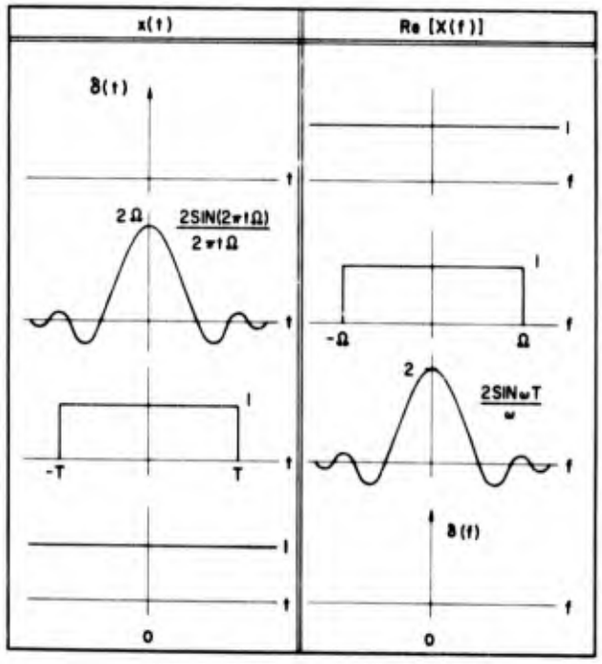


Fig. 1.2. Sine of x divided by x function.

Table 1.1. Comparison of Various Time Functions and Their Fourier Transforms



A large a will cause the time function to be narrow and the corresponding frequency function to be broad. Thus, a function of short time duration requires a Fourier transform that is defined over a wide range of frequencies. Conversely, a function that requires only a short range of frequencies to define it will have a large range in the time domain.

In actual practice, one cannot record the results of an experiment that is infinite in length. The question naturally arises as to what effect that has on the Fourier transform. One way of analyzing the problem is to regard the truncated record as being the function y, which is the product of two other functions,

$$\begin{aligned} &x(t) && \text{as usual} \\ &u_T(t) = \begin{cases} 1 & -T \leqslant t \leqslant T \\ 0 & \text{otherwise,} \end{cases} && (1.32) \\ &y(t) = x(t)\, u_T(t). \end{aligned}$$

Thus, the Fourier transform of y is

$$Y(f) = \int_{-\infty}^{\infty} x(t)\, u_T(t)\, e^{-j\omega t}\, dt. \tag{1.33}$$

It reduces to (using the convolution technique as discussed in Example 1.4):

$$Y(f) = \int_{-\infty}^{\infty} X(f')\, U_T(f' - f)\, df'. \tag{1.34}$$

The Fourier transform of $U_T(t)$ is given by

$$U_T(f) = \frac{2 \sin \omega T}{\omega}. \tag{1.35}$$

It is the same function discussed in Example 1.5. In the limit as T goes to infinity, the function takes on the same characteristics as $\delta(f)$, so that the right-hand side of Eq. (1.34) reduces to $X(f)$. On the other hand, for finite T, the observed functions will be affected by the truncation. If x were a cosine wave, then

$$x(t) = A \cos 2\pi f_0 t,$$

$$X(f) = \frac{A}{2} [\delta(f - f_0) + \delta(f + f_0)],$$

and

$$Y(f) = \frac{A}{2\pi}\left[\frac{\sin\,[2\pi T(f-f_0)]}{f-f_0} + \frac{\sin\,[2\pi T(f+f_0)]}{f+f_0}\right]. \tag{1.36}$$

1.5 Power Spectrum

In Section 1.3 the concept of the mean square is discussed, and an expression for the mean square is given:

$$\Psi_x^2 = \frac{1}{t_2 - t_1}\int_{t_1}^{t_2} x^2(t)\,dt. \tag{1.7}$$

Dimensionally, Ψ_x^2 is proportional to the mean square energy per unit time, which is, of course, power.

The power spectral density of the function x, written $G_x(f)$, is an extension of this concept. The interpretation of $G_x(f)$ is that the integral

$$\Psi^2(f_1, f_2) = \int_{f_1}^{f_2} G_x(f)\,df, \qquad 0 \leqslant f_1 \leqslant f_2 \tag{1.37}$$

is the power between the frequencies f_1 and f_2. Thus,

$$\Psi^2 = \int_0^{\infty} G_x(f)\,df. \tag{1.38}$$

It may be shown that an appropriate expression for G_x is given by

$$G_x(f) = 2\lim_{T\to\infty}\frac{1}{T}\left|\int_{-T/2}^{T/2} x(t)\,e^{-j\omega t}\,dt\right|^2, \qquad f \geqslant 0. \tag{1.39}$$

Statistical difficulties may arise if Eq. (1.39) is used without care. Statistically inconsistent estimates result when no frequency smoothing is performed. Also, mathematical precautions are necessary as pointed out in Ref. 4 (p. 580). These two considerations, along with the fact that only inefficient computational procedures were available in the past, have led to the dismissal of Eq. (1.39) as a suitable spectrum estimation procedure. However, the fast Fourier transform (see Chapter 4) eliminates the computation objection, proper frequency domain smoothing provides statistically consistent estimates, and finite record lengths eliminate the mathematical problems.

An equivalent definition, under appropriate circumstances, is

$$G_x(f) = 2\int_{-\infty}^{\infty} R_x(\tau)\, e^{-j\omega\tau}\, dt = 4\int_0^{\infty} R_x(\tau)\cos(2\pi f\tau)\, d\tau, \quad f \geqslant 0, \tag{1.40}$$

where the term $R_x(\tau)$, the time autocorrelation function of x, is defined as

$$R_x(\tau) = \lim_{T\to\infty} \frac{1}{T}\int_{-T/2}^{T/2} x(t)\, x(t+\tau)\, dt. \tag{1.41}$$

Demonstration of the equivalence of these two methods of computing G_x is beyond the scope of this work but is discussed in several references [5, 6]. A third method of defining G_x is available. However, it is based on the concept of filtering, and discussion must be postponed until after filtering has been introduced.

The function G_x is usually called the one-sided power spectral density (PSD). Thus, the PSD of x means G_x as defined by either Eq. (1.39) or (1.40). The function R_x requires further discussion. Under the condition of ergodicity assumed earlier, it is equal to

$$R_x(\tau) = E[x(t)\, x(t+\tau)] = \int_{-\infty}^{\infty} x(t)\, x(t+\tau)\, p(x, t, \tau)\, dx, \tag{1.42}$$

where $p(x, t, \tau)$ is an appropriate joint probability density function.

Example 1.6 Suppose $n(t)$ is uncorrelated random noise with a zero mean. Then its autocorrelation is given by

$$R_n(\tau) = \frac{N}{2}\delta(\tau),$$

and

$$G_n(f) = N. \tag{1.43}$$

That is, the PSD of uncorrelated random noise is a constant. That type of data is known as white noise because the power is uniform (flat) through any given band and, therefore, is like white light which is more or less uniform in the visible portion of the optical spectrum.

While useful as a mathematical tool for clarifying the concept of the PSD, it must be noted that true white noise is not physically realizable because its variance is infinite. Devices which supposedly generate white noise in actuality produce noise whose PSD is flat out to some frequency and then drops off

for higher frequencies. That may only mean that the output of the device has a flat PSD over some frequency band of interest.

Example 1.7 Suppose that x is a sine wave with frequency f_c, amplitude A, and phase ϕ. That is,

$$x(t) = A \sin (2\pi f_c t + \phi). \tag{1.44}$$

The autocorrelation function is computed as follows:

$$R_x(\tau) = \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} A^2 \sin (2\pi f_c t + \phi) \sin [2\pi f_c (t+\tau) + \phi]\, dt$$

$$= \lim_{T\to\infty} \frac{1}{T} \left(\frac{A^2}{2}\right) \int_{-T/2}^{T/2} [\cos (2\pi f_c \tau) - \cos (2\pi f_c \tau + 4\pi f_c t + 2\phi)]\, dt$$

$$= \lim_{T\to\infty} \frac{A^2}{2T} \left[t \cos (2\pi f_c \tau) + \frac{1}{4\pi f_c} \sin (2\pi f_c \tau + 4\pi f_c t + 2\phi) \right]_{-T/2}^{T/2}$$

$$= \lim_{T\to\infty} \left\{ \frac{A^2}{2} \cos (2\pi f_c \tau) + \left(\frac{A^2}{8\pi f_c}\right)\left(\frac{1}{T}\right) \left[\sin (2\pi f_c \tau + 2\pi f_c T + 2\phi) \right.\right.$$

$$\left.\left. - \sin (2\pi f_c - 2\pi f_c T + 2\phi) \right] \right\} = \frac{A^2}{2} \cos (2\pi f_c \tau). \tag{1.45}$$

This is true because the term within the brackets is always less than 2. In the limit, this term must go to zero, as it is divided by T.

Note that the arbitrary phase angle ϕ has disappeared in the final result. As the phase angle was arbitrary, we could just as well have been dealing with cosine waves. Thus, the autocorrelation of a sinusoid, regardless of its phase, is always a cosine with zero phase. In general, phase information is lost in the autocorrelation process.

Applying Eq. (1.40),

$$G_x(f) = 4 \int_0^\infty \left[\frac{A^2}{2} \cos (2\pi f_c \tau) \right] \cos (2\pi f \tau)\, d\tau = \frac{A^2}{2} \delta(f - f_c), \qquad f > 0. \tag{1.46}$$

This is a major result. It says that the PSD of a sinusoid of frequency f_c is a single δ function at frequency f_c. The multiplier of the δ function is equal to the power in the sinusoid. Recalling Eq. (1.38),

$$s_x^2 = \int_0^\infty G_x(f)\,df = \frac{A^2}{2}\int_0^\infty \delta(f-f_c)\,df = \frac{A^2}{2}, \tag{1.47}$$

which is exactly the result obtained in Example 1.2.

Example 1.8 As a last example of power spectral density for this section, consider the case where the autocorrelation is a truncated cosine wave

$$R_x^*(\tau) = \begin{cases} \dfrac{A^2}{2}\cos(2\pi f_c\tau) & \tau \leqslant T \\ 0 & \tau > T. \end{cases} \tag{1.48}$$

This is precisely the type of function that might be obtained in some practical situation. It is truncated because the experiment cannot be continued indefinitely.

R_x^* could be interpreted in the following way:

$$R_x^*(\tau) = R_x(\tau)\,u_T(\tau), \tag{1.49}$$

where R_x is as defined in the previous case and $u_T(\tau)$ is as defined in Example 1.5. With this interpretation, we could find $G_x^*(f)$ using the convolution

$$\begin{aligned} G_x^*(f) &= 2\int_{-\infty}^{\infty} u_T(\tau)\,R_x(\tau)\,e^{-j2\pi f\tau}\,d\tau \\ &= \int_{-\infty}^{\infty} U_T(f-f')\,G_x(f')\,df' \\ &= \int_{-\infty}^{\infty} \frac{2\sin[2\pi T(f-f')]}{2\pi(f-f')}\,\frac{A^2}{2}\,\delta(f'-f_c)\,df' \\ &= \frac{A^2}{2}\left\{\frac{2\sin[T(\omega-\omega_c)]}{\omega-\omega_c}\right\} \qquad \text{for } f>0. \end{aligned} \tag{1.50}$$

This function is shown in Fig. 1.3.

1.6 Digital Data Functions

Digital data collected during a test may be thought of as related to a hypothetical continuous function $x(t)$, defined for all t, by the equation

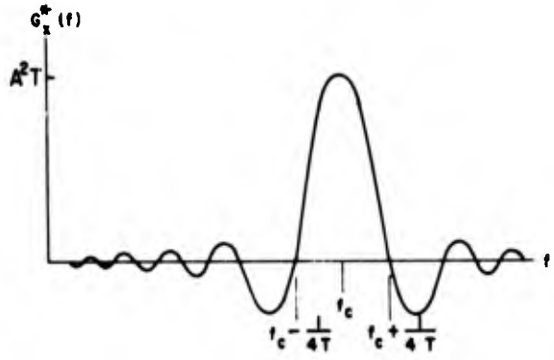


Fig. 1.3. PSD of a truncated sinusoidal function.

$$x_i = \int_{-\infty}^{\infty} \delta(t - i\Delta t)\, x(t)\, dt, \qquad i = 0, \cdots, N. \tag{1.51}$$

There are a number of indexing conventions that could be used besides the above. The other possibilities, such as $i = 1, \cdots, N$ or $i = 0, \cdots, (N-1)$, will be employed whenever they are convenient.

The question naturally arises as to how well the sequence $\{x_i\}$ represents the function x. Put another way, does the series of x_i values tell everything about x? The asymptotic answer to this question is given in the sampling theorem,* which will be discussed below. The word *asymptotic* is used because two hypotheses of the sampling theorem can be met only in limiting cases.

Sampling Theorem

Hypotheses

1. x is a random variable defined for $-\infty < t < \infty$.
2. The power spectral density of x, $G_x(f)$, exists.
3. $G_x(f) = 0$ for $f \geqslant B$.

Conclusion

Let Δt be any sampling interval such that $\Delta t \leqslant 1/(2B)$. Given $\{x_i\}$ sampled at Δt intervals, $i = -\infty, \cdots, -1, 0, 1, \cdots, \infty$, then $x(t)$ can be reconstructed uniquely (almost everywhere). Put another way, if the power in x is limited to a band less than $1/(2\Delta t)$ Hz, then sampling at interval Δt enables one to reconstruct the function x uniquely. There are some mathematical cases which must be excepted. One of these will be discussed in an example.

*See Ref. 7.

Proof of the Sampling Theorem

A precise (rigorous) proof will not be given, as it would necessitate more mathematics than is appropriate to this work. A heuristic (plausible but incomplete) proof may be given the following way. Hypotheses 2 and 3 of the theorem are sufficient to ensure that X, the Fourier transform of x, exists and that $X(f) = 0$ for $|f| > B$. That is, X is a function defined for $-B \leqslant f \leqslant B$. Thus, X can be represented by a Fourier series. In particular, one could use the representation

$$X(f) = \frac{1}{2B} \sum_{k=-\infty}^{\infty} c_k \exp\left[-j2\pi kf/(2B)\right], \qquad -B \leqslant f \leqslant B, \tag{1.52}$$

where

$$c_k = \int_{-B}^{B} X(f) \exp\left[j2\pi fk/(2B)\right] df, \qquad \text{for } k = -\infty, \cdots, -1, 0, 1, \cdots, \infty. \tag{1.53}$$

This is only one of several methods of defining the Fourier coefficients. It may be shown to be equivalent to the usual formulation, which involves sines and cosines. The c_k term as defined above absorbs some of the coefficients encountered in other definitions.

Let

$$\Delta t = \frac{1}{2B}. \tag{1.54}$$

Also, note that the limits of integration of Eq. (1.53) could be extended from $(-B, B)$ to $(-\infty, \infty)$, as $X(f)$ is equal to zero on the extended interval. Thus, Eq. (1.53) could be rewritten as

$$c_k = \int_{-\infty}^{\infty} X(f) \exp\left(j2\pi fk\Delta t\right) df. \tag{1.55}$$

But the right-hand side of Eq. (1.55) is exactly equal to x_k. That is, the set $\{x_k\}$ uniquely represents $X(t)$.

The last question discussed is how to obtain x for values of t *not* equal to $k\Delta t$. There are two answers. If X is available, then $k\Delta t$ may be replaced with t, so that

$$x(t) = \int_{-B}^{B} X(f) \exp\left(j2\pi ft\right) df. \tag{1.56}$$

Alternatively, given only the x_k terms, one could write that

$$x(t) = \int_{-B}^{B} X(f) \exp(j2\pi ft)\, df$$

$$= \int_{-B}^{B} \frac{1}{2B} \sum_{k=-\infty}^{\infty} x_k \exp(-j2\pi fk\Delta t) \exp(2\pi ft)\, df$$

$$= \frac{1}{2B} \sum_{k=-\infty}^{\infty} x_k \int_{-B}^{B} \exp[j2\pi f(t - k\Delta t)]\, df. \tag{1.57}$$

Using the results of Section 1.4, the integral term in Eq. (1.57) may be evaluated and found to be

$$\int_{-B}^{B} \exp[j2\pi f(t - k\Delta t)]\, df = \frac{2 \sin[2\pi B(t - k\Delta t)]}{2\pi(t - k\Delta t)}. \tag{1.58}$$

Thus, for $t = k\Delta t$, Eq. (1.57) reduces to

$$x(k\Delta t) = \frac{1}{2B}(x_k 2B) = x_k. \tag{1.59}$$

Note that Eq. (1.59) is valid only for values of k which are integers. To recover $x(t)$ for $(t/\Delta t)$ not an integer, the general expression in Eq. (1.57) must be used. The integral within that equation may be replaced with the $(\sin x)/x$ type of term given in Eq. (1.58), so that Eq. (1.57) reduces to

$$x(t) = \sum_{k=-\infty}^{\infty} x_k \left\{ \frac{\sin[2\pi B(t - k\Delta t)]}{2\pi B(t - k\Delta t)} \right\}. \tag{1.60}$$

Given x_k, Eq. (1.60) enables one to reconstruct $x(t)$ for any value of t.

The theorem holds only for infinite record lengths and for bandlimited functions. These requirements cannot be met in actual practice. However, long record lengths and the use of high-quality "low-pass" filters (see Chapter 3) before the functions are sampled will yield satisfactory results. The theorem breaks down at the frequency $B = 1/(2\Delta t)$ when applied there. That is shown in the following example.

Example 1.9 Let x be given by

$$x(t) = A \sin (2\pi Bt + \phi), \tag{1.61}$$

where $B = 1/(2\Delta t)$, and $0 \leqslant \phi < 2\pi$. That is, x is a sine wave of arbitrary phase whose frequency is commensurate with the sampling rate. Then

$$\begin{aligned} x_k = x(k\Delta t) &= A \sin (\pi k + \phi) = A \ [\sin \pi k \cos \phi + \cos \pi k \sin \phi] \\ &= A (-1)^k \sin \phi. \end{aligned} \tag{1.62}$$

The last step is true because $\sin \pi k = 0$ for all k, and $\cos \pi k = (-1)^k$. Given only the sequence $\{x_k\}$, the observer sees a term of magnitude $(A \sin \phi)$, which alternates. If ϕ is unknown, there is no way of reconstructing the function. In fact, if $\phi = 0$, the observer will see nothing! In any case, as $0 < |\sin \phi| < 1$, the observed sequence will have amplitudes which are lower in value than A. Calculations made using these data will be biased. An example of this is seen in Fig. 1.4.

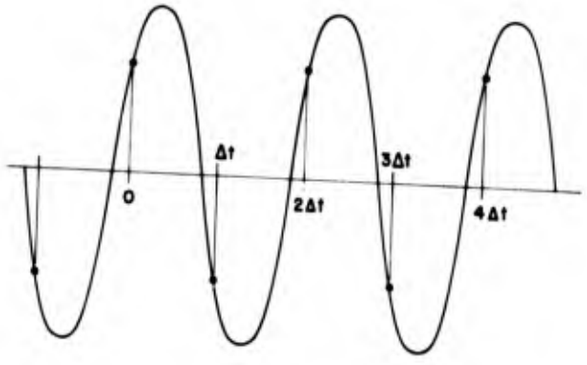


Fig. 1.4. $x(t) = A \sin (2\pi Bt + \phi)$. Data indicate when sampling takes place.

There is a folktale current in the engineering profession that a sampling rate of five times the highest frequency to be observed is necessary for good, unbiased results. This is pessimistic for most applications. Furthermore, use of the 5-sample/cycle criteria is wasteful, since a 2.5-sample/cycle rule has been found to give excellent results. This rule says that if f_c is the highest frequency to be observed, then set

$$B = \frac{5}{4} f_c$$

and

$$\Delta t = \frac{2}{5 f_c}. \tag{1.63}$$

Example 1.10 If $f = 1/(2\Delta t)$ is the highest frequency that can be observed in any case, what is the lowest frequency which can be observed? An answer to this question may be found by examining the PSD of sine waves. Suppose two sinusoid functions compose a data function, and that each has amplitude A. That is,

$$x(t) = A\left[\sin(2\pi f_1 t + \phi_1) + \sin(2\pi f_2 t + \phi_2)\right], \qquad -\infty < t < \infty. \tag{1.64}$$

The phases ϕ_1 and ϕ_2 are constant but random. The autocorrelation of this may be shown to be

$$R_x(\tau) = \frac{A^2}{2}(\cos 2\pi f_1 \tau + \cos 2\pi f_2 \tau), \tag{1.65}$$

and the PSD is then

$$G_x(f) = \frac{A^2}{2}\left[\delta(f - f_1) + \delta(f - f_2)\right], \qquad f \geqslant 0.$$

Suppose that $R_x(\tau)$ is then limited to the range $-T \leqslant \tau \leqslant T$. Call this new function $R_x^*(\tau)$. That is,

$$R_x^*(\tau) = R_x(\tau)\, u_T(\tau),$$

and

$$u_T(\tau) = \begin{cases} 1 & -T < \tau < T \\ 0 & \text{otherwise.} \end{cases} \tag{1.66}$$

The Fourier transform of $u_T(\tau)$ is discussed in Section 1.4. $G^*(f)$, the PSD corresponding to $R^*(\tau)$, may be obtained in a manner similar to that of Example 1.8:

$$G_x^*(f) = 2\int_{-\infty}^{\infty} R_x^*(\tau)\, e^{-j2\pi f\tau}\, d\tau$$

$$= 2\int_{-\infty}^{\infty} R_x(\tau)\, u_T(\tau)\, e^{-j2\pi f\tau}\, dt. \tag{1.67}$$

The convolution theorem may now be used:

$$G_x^*(f) = \int_{-\infty}^{\infty} G_x(f')\, U_T(f-f')\, df'$$

$$= \frac{A^2}{2} \int_{\infty}^{\infty} [\delta(f'-f_1) + \delta(f'-f_2)] \cdot \frac{2 \sin [2\pi T(f-f')]}{2\pi(f-f')}\, df'$$

$$= \frac{A^2}{2} \left\{ \frac{\sin [2\pi T(f-f_1)]}{2\pi(f-f_1)} + \frac{2 \sin [2\pi T(f-f_2)]}{2\pi(f-f_2)} \right\}. \tag{1.68}$$

Thus, the PSD $G_x^*(f)$ consists of two $(\sin x)/x$ terms, one centered at f_1 and the other at f_2. When these sinusoids are close together, being able to resolve them (being able to tell if there are one or two sinusoids) parallels a similar problem in optics. Physicists use Rayleigh's criterion in these circumstances, which also applies to the case given in this example. Rayleigh's criterion [8] says that the two sinusoids can be resolved if $\Delta f = |f_1 - f_2|$ is as large as the following:

$$\Delta f = \frac{1}{2T} \quad \text{well resolved}$$

$$\Delta f = \frac{1}{4T} \quad \text{just resolved.} \tag{1.69}$$

That is, if f_1 and f_2 are separated by $1/2T$ or more hertz, then according to Rayleigh's criterion they may be resolved easily.†

Finally, to return to the original question, suppose there is only one sinusoid and it has a low frequency. Then in the range $0 \leqslant f \leqslant 1/2T$, it will be very difficult to distinguish the frequency of the sinusoid. Thus, if it is desired to examine a sine wave with a frequency of approximately f, then the length of record must be at least $1/(2f)$ to be able to say anything about its frequency.

Another problem arising from sampling data is that of aliasing. If a sinusoid of frequency higher than $1/(2\Delta t)$ Hz is sampled at a rate of $1/\Delta t$ samples per second, then the sinusoid will appear as a lower frequency. The frequency $f = 1/(2\Delta t)$ is called the folding frequency. To illustrate, suppose that $x(t) = \sin 2\pi f t$, and that $f = [1/(2\Delta t)]\,(n+p)$ where n is an integer and $0 \leqslant p < 1$. Then

$$x(k\Delta t) = \sin (2\pi f k \Delta t) = \sin 2\pi k \Delta t \left(\frac{n+p}{2\Delta t}\right)$$

$$= \sin [\pi k(n+p)] = \sin \pi k n \cos \pi k p + \cos \pi k n \sin \pi k p. \tag{1.70}$$

†Readers familiar with quantum mechanics will recognize this resolution problem as a form of the uncertainty principle.

Since $\sin \pi kn = 0$, Eq. (1.70) reduces to

$$x(k\Delta t) = [\cos \pi kn] \sin \pi kp = [(-1)^n]^k \sin \pi kp = [(-1)^n]^k \sin (2\pi k \Delta t f'), \tag{1.71}$$

where

$$f' = p/2\Delta t.$$

By definition,

$$0 \leqslant f' < B. \tag{1.72}$$

If n is even, then the expression is simply

$$x(t) = \sin (2\pi t f') \tag{1.73}$$

for $t = k\Delta t$. On the other hand, if n is odd, then

$$x(t) = \cos \pi k \sin \pi kp = \sin (\pi k - \pi kp) = \sin [2\pi t(B - f')] \tag{1.74}$$

for $t = k\Delta t$. This is a many-to-one mapping. In summary, if $f > B$, to find the frequency of the aliased sinusoid:

1. Divide f by B, and obtain n and p;
2. If n is even, the aliased frequency is simply pB;
3. If n is odd, the aliased frequency is $(1 - p)B$.

At least one example of aliasing is well known to all moviegoers; namely, the phenomenon of stagecoach wheels seeming to go backwards. In this case, one period would not be a complete revolution of the coach wheel, but rather a function of the spoke separation. If there are 12 spokes, one period is 1/12 of a revolution.

After the data function has been sampled, it may be necessary to obtain its PSD or to examine its Fourier transform. The Fourier transform is the most basic of these two frequency concepts, so its definition is examined first. Given $\{x_k\}$, $(k = 0, \cdots, N)$, the Fourier transform is

$$X(f) = \Delta t \sum_{k=0}^{N} x_k \exp (-j2\pi f k \Delta t). \tag{1.75}$$

Note that $X(f)$ is also what would be obtained if the transform of $x_s(t)$ were taken, where

$$x_s(t) = \sum_{k=0}^{N} x(t)\, \delta(t - k\Delta t) = x(t) \left[\sum_{k=0}^{N} \delta(t - k\Delta t) \right]. \tag{1.76}$$

The expression in brackets is referred to as the sampling function. Interesting properties may be derived for it.

Equation (1.75) is valid for all f in the range $(0, B)$. However, not all of these values need be considered. In fact, if the sequence $\{X_\ell\}$ is defined by

$$X_\ell = \Delta t \sum_{\kappa=0}^{N} x_k \exp\left(\frac{-j\pi\ell k}{N}\right), \tag{1.77}$$

then $\{x_k\}$ can be obtained from $\{X_\ell\}(\ell = 0, \cdots, N)$ using the following equation

$$x_k = \Delta f \sum_{\ell=0}^{N} X_\ell \exp\left(\frac{j\pi\ell k}{N}\right), \qquad k = 0, \cdots, N. \tag{1.78}$$

The equivalence of Eq. (1.77) and Eq. (1.78) may be shown by inserting one of the relations into the other and applying the formulas given in Appendix B to reduce the resulting equation to a tautology.

The digital power spectral density may also be defined in several ways. The classical method, as popularized by Blackman and Tukey [9], obtains the PSD in two steps. First, the sample autocorrelation function is computed:

$$R_r = \frac{1}{N-r} \sum_{k=0}^{N-r} x_k x_{k+r}, \qquad r = 0, \cdots, m. \tag{1.79}$$

The subscript r corresponds to the variable τ and is sometimes referred to as the lag value, whereas m, the largest lag value, is called, simply, the number of lags (calculated). The second step consists of taking the cosine transformation of the autocorrelation function, which yields

$$G_p = \Delta t \left[2R_0 + 4 \sum_{r=1}^{m} R_r \cos\left(\frac{\pi p r}{m}\right)\right]. \tag{1.80}$$

The alternative is to use the Fourier transform and obtain

$$G'_q = \frac{1}{N\Delta t} X_q^* X_q = \frac{1}{N\Delta t} |X_q|^2, \qquad q = 0, \cdots, N, \tag{1.81}$$

where X_q^* is the complex conjugate of X_q. This version of the PSD may be shown to be the equivalent of using the following autocorrelation:

$$R'_r = \frac{1}{N} \sum_{k=0}^{N-r} x_k x_{k+r}, \qquad r = 0, \cdots, m. \tag{1.82}$$

As $m \to N$ and $N \to \infty$, G_p and G_p' will approach each other. However, for any given finite sequence, there may be differences. The Blackman-Tukey PSD is currently a standard, so people using Eq. (1.81) attempt to rearrange the method so that the result it yields is equivalent to the Blackman-Tukey PSD. That is difficult to do. The direct method using Eq. (1.81) has important computational advantages. Thus, it is quite possible that the direct method with adjustments may become the standard usage.

CHAPTER 2
PREPROCESSING OF DATA

2.1 Data Acquisition Systems

A data acquisition system must exist between the random process that generates data and the digital computer that processes the data. It must be an organized mechanism for collecting the data and transforming it into a state compatible with the computer. Such a device or devices will be referred to as a data acquisition system (DAS). Systems of this sort might be simple or extremely complex. A simple system might consist of an individual filling out keypunch forms with stock market information obtained from the evening paper. That would be followed by conversion to punched cards through the use of a keypunch machine. The original data are already in digital form. The only additional requirement is that the data be put into a form that may be readily put into a computer, such as punched cards.

A general and somewhat simplified type of DAS will be discussed, and some of the more basic problems that affect data processing will be examined in detail. Most of the data processing under consideration is not done in real time. That is, the data of interest are recorded during a test and analyzed at a later time (frequently there is much reprocessing of data). Therefore, the DAS may be conveniently divided into two parts. The first part, as shown in Fig. 2.1a, will be referred to as the recording portion of the DAS, or simply the recording system. It contains the following pieces of equipment:

1. Transducer: This converts the physical phenomena being observed into either a continuous or a discrete electrical analog of the phenomena.

2. Multiplexer, filter and/or converter: This device reduces the bandwidth of the signal to some allowable bandwidth. Then the signal is either frequency multiplexed or converted to a discrete form and time multiplexed with other data simultaneously being observed. If the transducer is digital, the filtering precedes the digitizing.

3. Transmission link: This can vary from a simple cable to a transmitter-receiver combination of extreme complexity.

4. Recording system: This is usually some type of magnetic tape machine.

Figure 2.1b shows the playback and conversion portion of the data system. It performs the following functions:

1. Playback system: This is essentially the reverse of the magnetic recording system.

2. Conversion, etc.: If the data are not already in digital form, this stage of the system does the analog-to-digital conversion (ADC). Also included in the

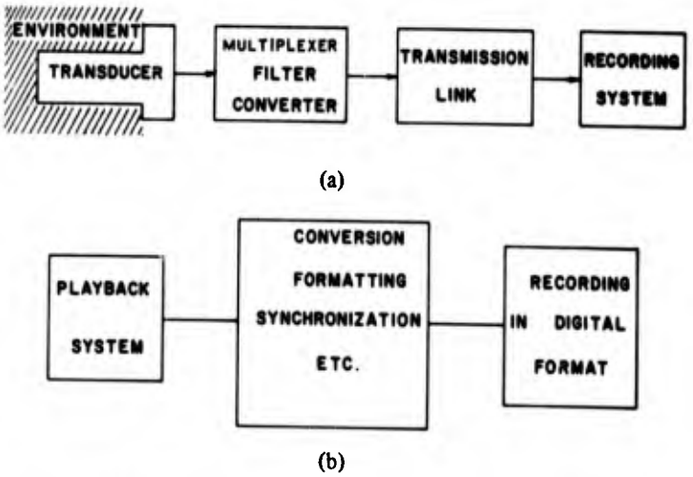


Fig. 2.1. Data acquisition system: (a) recording portion, (b) playback and conversion portion.

hardware at this point might be discriminators, filters, and a multiplexer. If the data were already in binary form, the hardware might take the form of bit, word, and frame synchronizing devices.

3. The final recording device of the DAS usually would be a digital tape drive whose format is compatible with the digital computer to be used in reducing the data.

The large variety of DAS's currently in use prohibits a detailed discussion and in any event that topic would be more appropriately covered in a work on telemetry. A discussion of the errors introduced by a typical DAS is important and relevant, however, since such errors must at least be known if not accounted for in the final presentation of the data.

Figure 2.2 shows a hypothetical DAS. The transducer, assumed to be perfect, produces a voltage proportional to the function being observed. This electrical

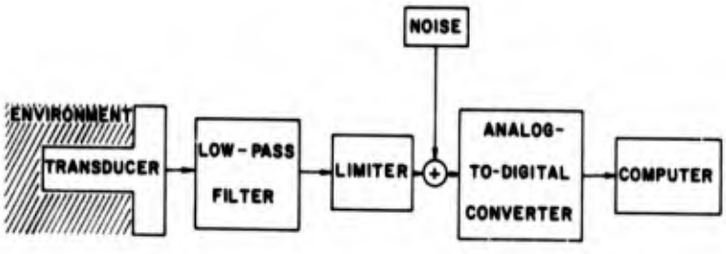


Fig. 2.2. Hypothetical DAS.

signal is then put into a low-pass filter. The filter is designed so that it will not pass any power corresponding to a frequency higher than one-half of the sampling rate of the digitizer for that function. From the filter the data pass through the limiter. The limiter in the figure symbolizes the fact that there are definite limits to the magnitude of the voltage which the DAS will pass.

Actually, there are a number of components in any system that tend to act as limiters, including the transducer itself and the ADC. All such devices have been lumped together, and the resulting worst case assumed for the system. At the next stage, additive noise is introduced. The noise may be either correlated or uncorrelated. Next, the data are digitized. In a later section, it will be shown how the ADC also introduces noise into the data. Finally, the data are shown entering the computer to be processed. The digital recording and playback techniques have now reached a point where they need not be considered as error sources relative to the errors introduced from other devices.

The device shown in Fig. 2.2 may be used as an error model for a wide variety of DAS's. In fact, it makes good sense to describe the errors and inaccuracies in the system in terms of such a simplified model because it is difficult to make a realistic theoretical model for such a real system if all possible errors from all possible components are considered. Some sources of error might be overlooked in the analysis, resulting in an error figure that is too optimistic. On the other hand, some error figures from individual pieces of equipment may be estimated to be too large, resulting in a pessimistic evaluation. The best way would seem to be to actually measure the total system error. Because of the difficulty of exciting the transducers adequately on site, the DAS would be disconnected between the transducer and the input to the next stage here taken to be a filter. Appropriate voltages of quality test signals are then put into the remainder of the system. The same techniques employed in analyzing the data may then be applied to the analysis of the test signals. Properly performed, such a test will give the user the required information on overall system performance.

The noise introduced into the data by the ADC and other sources tends to be additive. The expected power spectral density (PSD) of a time history, G_x^*, as finally stored in the computer of the system in Fig. 2.2 would be the sum of components as follows:

$$G_x^*(f) = G_x(f) + G_n(f) + G_{ADC}(f). \tag{2.1}$$

G_x is the PSD of data before it is introduced into the system. G_n is the spectrum of the noise generated within the system exclusive of that produced by the ADC. Finally, the G_{ADC} term is the component produced exclusively by the ADC. All of this assumes that both x and n are within the limits of the limiter. When one or the other exceeds these limits, the problem becomes considerably more complicated, and such a simple expression as Eq. (2.1) may not be correct.

Example 2.1 Suppose that x is a cosine wave with frequency f_c and that the record examined is long enough so that the power spectrum of x will be approximately

$$G_x(f) = \frac{A^2}{2}\,\delta(f - f_c), \qquad 0 \leqslant f \leqslant \infty. \tag{2.2}$$

When x is passed through the limiter, the output has the form

$$y(t) = \begin{cases} x(t) & |x(t)| < C \\ C & x(t) \geqslant C \\ -C & x(t) \leqslant -C. \end{cases} \tag{2.3}$$

When A is much greater than C, then y looks very much like a square wave with amplitude equal to C. In this case $y(t)$ will have the form

$$y(t) = \frac{4C}{\pi}\left[\cos(2\pi f_c t) - \frac{1}{3}\cos(6\pi f_c t) + \frac{1}{5}\cos(10\,\pi f_c t) - \cdots\right]. \tag{2.4}$$

The PSD of $y(t)$ will be

$$G_y(f) = \frac{8C^2}{\pi^2}\sum_{i=1}^{n}\frac{1}{(2i-1)^2}\,\delta[f - (2i-1)f_c]. \tag{2.5}$$

Partially truncated sinusoids will have PSD's somewhere between the above and Eq. (2.2). The exact form of such a PSD can be found by using describing function techniques as discussed by Truxal [10].

Equipment malfunction and operator error are both major problems with DAS's. A properly designed system should include self-checking features. If properly implemented, such automatic checks will perform an invaluable function. In much data analysis, several days of processing through a complex system may be wasted if obviously meaningless results are found in the final plots or printouts. Proper self-checking procedures can eliminate a large percentage of these wasted results.

2.2 Analog-To-Digital Conversion

The ADC is the potential source of a number of types of errors. Many of these are unimportant if PSD's are the only output. Where gain factors or phase information are required, the various degradations introduced by the ADC become important and cannot be overlooked.

ADC's are usually either binary or binary-coded decimal (BCD). A binary ADC maps a continuous voltage into any one of 2^N numbers, while the BCD maps the voltage onto a set of 10^N numbers. Binary ADC's are simpler to build but normally require machine language programming to process their data output. If properly formatted onto magnetic tape, the output of a BCD ADC may be read directly by a computer program written in the FORTRAN language. That can be a tremendous convenience. However, BCD digitizers and formats are relatively inefficient. When large volumes of data are to be processed, loss of efficiency cannot be ignored.

Example 2.2 Suppose that a scale of 1,000 is required. That is, the data signal is to be mapped onto the numbers 000 through 999. Suppose that 000 corresponds to −5.00 V and that 999 corresponds to +4.99 V. This conversion is assumed to be linear, so that one count, i.e., each digit, is worth 0.01 V, and 525 corresponds to any reading between +0.245 and +0.255 V. Ten binary digits (bits) could adequately be used to express such numbers, as $2^{10} = 1024$. In binary notation, 0000000000 would correspond to −5 V ±0.005 V, 0111110100 would correspond to 0.00 V, etc. On the other hand, while only three decimal digits are required to express the same range, the decimal digits so used must be coded into BCD. One common method of coding maps each decimal digit into a block of six binary digits as follows:

$$\begin{aligned} 0 &\rightarrow 000000 \\ 1 &\rightarrow 000001 \\ 2 &\rightarrow 000010 \\ 3 &\rightarrow 000011 \\ 4 &\rightarrow 000100 \\ 5 &\rightarrow 000101 \\ 6 &\rightarrow 000110 \\ 7 &\rightarrow 000111 \\ 8 &\rightarrow 001000 \\ 9 &\rightarrow 001001 \end{aligned} \tag{2.6}$$

In this system, six binary digits are used for each decimal digit. That means that 18 binary digits will be required to express the same number which required only 10 digits in straight binary encoding. In addition, the tape formats directly readable by FORTRAN routine are limited to short blocks of BCD information. Such blocks (called records) are read at a slower rate than longer blocks because many digital tape units must stop completely between each record. The shorter the record length, the more stops are necessary to pass the same amount of data. The greater the number of stops, the slower the average speed.

As discussed in the example, one common encoding of decimal digits uses a six-bit code. Six bits may be used to represent 64 different numbers and characters. If letters of the alphabet and other such symbols are not required, four-bit

encoding may be used; in the encoding given in the example, only the last four binary digits are used. On the other hand, some computer systems have gone to eight-bit encoding, which is very inefficient for data unless two numeric characters are packed into a single eight-bit character.

In addition to variations in the number of bits used when coding decimals, some systems use entirely different coding schemes. For example, the complement of the BCD code discussed above may be used. This consists of replacing ones with zeroes and vice-versa.

Many digital transducers, such as shaft encoders, generate a coded form of binary numbers known as grey code. In this code, only a single digit changes between any two successive numbers. One three-bit grey code is as follows:

$$\begin{aligned} 0 &\rightarrow 000 \\ 1 &\rightarrow 001 \\ 2 &\rightarrow 011 \\ 3 &\rightarrow 111 \\ 4 &\rightarrow 101 \\ 5 &\rightarrow 100 \\ 6 &\rightarrow 110 \end{aligned} \tag{2.7}$$

The advantage of this code is that if the encoder is read while it is changing values, the uncertainty is only one count. Contrast this with the following: Suppose that an uncoded 10-bit transducer is changing from 511_{10} to 512_{10} counts, i.e., from 0011111111_2 to 0100000000_2. If the reading is made during the change, an incorrect number such as 0101101101_2 might be the output. This would be a significant error.

The principal disadvantage of grey code is that it must be converted into binary before it can be used in a computer, and this conversion adds extra steps to the computer time needed to calibrate the data.

All ADC's introduce a type of error known as quantization noise, shown by Fig. 2.3. The relationship between the input quantity, here taken to be voltage, and the output in counts is

$$e = a + n_d. \tag{2.8}$$

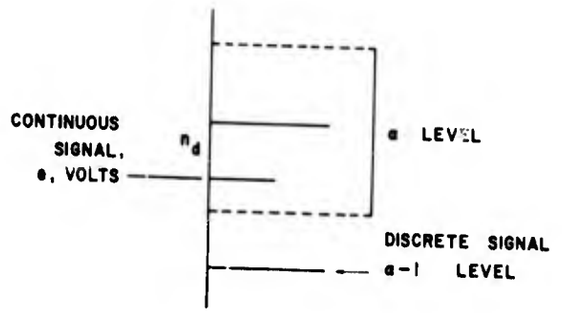


Fig. 2.3. Noise introduced by quantization.

The a term is always an integer. The quantity n_d is the quantization error and is such that

$$-\frac{1}{2} \leqslant n_d < \frac{1}{2}. \tag{2.9}$$

It is reasonable to assume that n_d is uniformly distributed:

$$f(n_d) = \begin{cases} 1 & \left(-\frac{1}{2} \leqslant n_d \leqslant \frac{1}{2}\right) \\ 0 & \text{otherwise.} \end{cases} \tag{2.10}$$

The mean and the variance of n_d may be found by taking expectations, as described in Chapter 8:

$$\mu_{n_d} = \int_{-1/2}^{1/2} n_d \, dn_d = 0,$$

$$\sigma_{n_d}^2 = \mathrm{E}\left[(n_d - \mu_{n_d})^2\right] = \int_{-1/2}^{1/2} n_d^2 \, dn_d = \frac{1}{12}. \tag{2.11}$$

The errors for any two readings are uncorrelated for most digitizers. Thus, the digital autocorrelation of the noise is expected to be

$$R_{n_d}(i\Delta t) = \begin{cases} \frac{1}{12} & i = 0 \\ 0 & \text{otherwise.} \end{cases} \tag{2.12}$$

The PSD is therefore

$$G_{n_d}(f) = 2\Delta t \, \frac{1}{12} = \frac{\Delta t}{6}, \qquad 0 \leqslant f \leqslant \frac{1}{2\Delta t}. \tag{2.13}$$

The units of Eq. (2.13) are (counts)^2/Hz. If a range of e volts goes into 2^N counts, then G_{n_d} can be given in terms of volts:

$$G_{n_d}(f) = \frac{\Delta t}{6} e^2 \, 2^{-2N}. \tag{2.14}$$

Thus, for a fixed voltage range, the PSD of the quantization noise decreases exponentially with the number of bits in the quantization word. For large N, say N greater than 7, the quantization noise tends to be negligible.

There are other digitizing errors. Their importance will vary with the particular application. In summary they are

1. Aperture error. This error arises from the fact that the sample is taken over a finite period of time rather than instantaneously. The aperture, which is the length of time over which the data are averaged, should be small with respect to the sampling interval.

2. Jitter. If the length of the interval between samples varies slightly in some random manner, the resulting uncertainty is called jitter. Jitter can affect PSD's and may cause degradation of phase information at high frequencies.

3. Nonlinearities. This category covers a number of possible problems, usually traceable to the ADC being out of adjustment or to some parts actually being inoperative. Among the problems which can occur are (a) bit dropout—a portion of the circuitry corresponding to one of the bits is inoperative on an intermittent or regular basis, (b) spacing—the spacing between the levels may become nonuniform, resulting in some levels being more likely than others, and (c) zero discontinuity—if a digitizer has a range which includes both plus and minus, it is possible that misalignment of the ADC will result in a large discontinuity at the zero count value.

2.3 Calibration

This section is concerned with the problem of converting the numbers of the ADC output into a set of numbers that represent actual physical units. To clarify the process, a single time history of data is discussed, and a particular type of transducer is assumed. It is not difficult to generalize to other kinds of transducers.

Figure 2.4 shows a transducer of the type employed in subsequent discussions. A constant voltage s, for example 5 V, is applied to the electrical input. The transducer consists of a potentiometer whose movable, or wiper, arm is activated in some manner by the physical quantity to be measured. Thus, as the process x moves the arm of the potentiometer, the output voltage varies between zero and 5 V. This output will be referred to as v_x. After v_x has been digitized, it becomes yet another process which will be called c_x. $c_x(t)$ will be a whole number between zero and $(2^N - 1)$ if the ADC is binary.

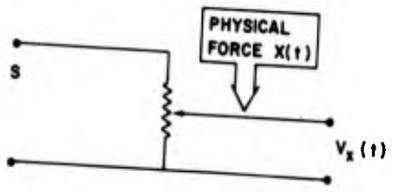


Fig. 2.4. Typical transducer.

Now suppose that the excitation voltage s varies with time about a nominal value s_n. The data value of $x(t)$, here labeled $\hat{x}(t)$, would be the output of the process shown in Fig. 2.5. The computing of $\hat{x}(t)$ is visualized most easily if the necessary steps are taken one at a time.

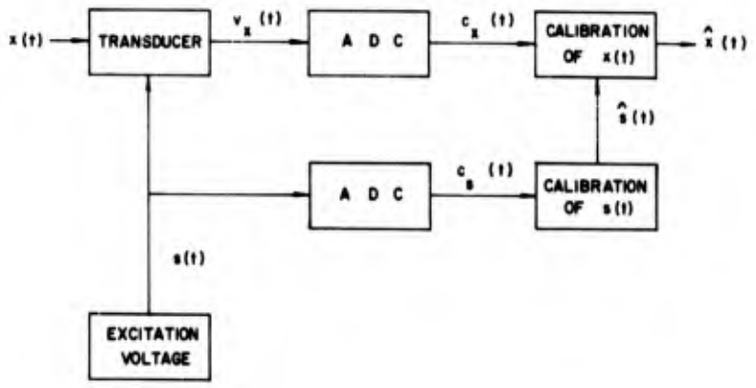


Fig. 2.5. Process that yields $\hat{x}(t)$, the data sequence.

1. For all functions, there is one parameter, v_c, that converts counts to volts. Thus, the standardizing voltage $\hat{s}$ is

$$\hat{s}(t) = v_c c_s(t). \tag{2.15}$$

This is simply the time history of the power supply used to excite the transducers. This function may be used with all channels excited by a common voltage source.

2. Therefore, the standardized voltage recorded for the transducer corresponding to the function x is

$$\hat{v}_s(t) = v_c c_x(t) \left[\frac{s_n}{\hat{s}(t)}\right]. \tag{2.16}$$

3. If the calibration is linear, then $x(t)$ may be obtained from

$$\hat{x}(t) = \left\{ v_c c_x(t) \left[\frac{s_n}{\hat{s}(t)}\right] - m_x \right\} e_x. \tag{2.17}$$

The constants m_x and e_x are, respectively, the offset (which may be zero) and the scale factor. These parameters are usually supplied with the transducer or measured at an installation's standards laboratory. The above equation could be rewritten in the form

$$\hat{x}(t) = a_1 \frac{c_x(t)}{\hat{s}(t)} - a_2,$$

where

$$a_1 = v_c s_n e_x$$
$$a_2 = m_x e_x. \tag{2.18}$$

4. If the calibration table is nonlinear, it is frequently approximated by a polynomial of the form

$$\hat{x}(t) = \sum_{k=0}^{K_x} c_k v_x^k(t). \tag{2.19}$$

K_x is usually less than or equal to 3. If K_x is 1, the equation reduces to the previous one.

The term *calibration* has two distinct meanings when applied to the above process. First, of course, is the actual laboratory calibration made of the instrument. A standards laboratory is required for that type of work. Each transducer must be subjected to known forces of the kind that the instrument was designed to measure, and the output of the device recorded accurately. The result of this calibration will be a table of numbers, physical units versus their electrical counterpart, usually volts. When plotted, such data take the shape of a hysteresis curve. A typical procedure is to fit a single curve (polynomial, exponential, or even a fairing with a french curve) to the plot and use only the resulting one-to-one functional relationship.

The second use of the word *calibration* is a misnomer. In that sense, it refers to the complete conversion process from digital counts to physical units, and is thus a calibration of data. As there is no other suitable nomenclature commonly in use, that usage of the word will be retained.

Another term to be used with care is *standardization.* Standardization may be thought of as an electrical-electronic calibration. One method of accomplishing it is through the use of a special standardizing voltage, recorded as a separate function. That procedure was described above. Sometimes it is not practical, and a second technique is used which involves pre- and post-experiment standardizing. In this case, the transducer is disconnected. Known resistances are then substituted for its input, and readings are taken of the result. As few as one or two resistances may be used, although five is a common number. The resistances would be chosen to correspond to the range of data. For example, suppose that a pressure transducer is to be used on a range of 0 to 20 psia, and that the instrument reads 0 V at 0 psia and 5 V at 20 psia. Then it would be reasonable

to substitute resistances that would generate outputs of 0, 1.24, 2.5, 3.75, and 5.0 V. These would correspond, if the instrument were linear, to applied pressures of 0, 5, 10, 15, and 20 psia. Switching such resistances can be made (and usually is) into an automatic process. It is carried out for all instruments by a single command, whereas excitation of the transducers themselves might be impossible because of inaccessibility or the difficulty of applying the correct forces to them. That type of standardization can only be made before or after the experiment; interruption of the experiment to generate standardization might result in the loss of critical portions of the data.

As an example of how such data would be used, suppose that both a pretest and a post-test standardization are conducted, and the substitutions for the transducer corresponding to the function $x(t)$ were, respectively, zero ohms and a value corresponding to the maximum internal impedance of the instrument. Before any data are reduced, these readings are digitized. These data are averaged so as to reduce the extraneous noise. In the absence of any other knowledge, the experimenter assumes that any changes between the start and end are occurring in a linear manner. In this case $\hat{s}(t)$ and $\hat{z}(t)$, the sample maximum and zero voltage, take the form

$$\begin{aligned} \hat{s}(t) &= \frac{(t-T_0)}{T_1-T_0}(s_1-s_0)+s_0 \\ \hat{z}(t) &= \frac{(t-T_0)}{T_1-T_0}(z_1-z_0)+z_0, \end{aligned} \tag{2.20}$$

where

T_0 = time of pretest standardization,
T_1 = time of post-test standardization,
s_0 = volts of pretest,
s_1 = volts of post-test,
z_0 = zero of pretest, and
z_1 = zero of post-test

There is a basic inconsistency in using this arrangement. What is actually being measured is the relative change of the system power supply and the ADC. The redundancy of doing it for every channel would not seem to be of much benefit. Of course, if an FM telemetry system is being used, then the procedure is required. In this case, the voltage-controlled oscillator (VCO) and discriminator are the components being standardized.

The post-test standardization information may not be available in many tests. In that event, only the pretest standardization is used, and $\hat{s}$ and $\hat{z}$ are constants. They are applied to the data in a manner similar to that of the standardizing procedure previously discussed.

One general problem with calibration is the large amount of information required. In a typical test, there may be hundreds of functions being analyzed. Each has its own set of calibration information, usually stored on punched cards. The total number of such cards may reach over two thousand. The assemblage of such a calibration check and the verification of its correctness is no small task. Mispunched cards, incorrectly completed load sheets, technician error, etc., have led very often to costly reruns. For this reason, the person in charge of the calibration should be both thorough and painstaking.

If funds permit, the calibration decks should be checked a few days prior to a test to make sure that they are in satisfactory condition. This can be done either with a special program or with the actual calibration program and a dummy input.

2.4 Phase and Numerical Integration and Differentiation

The functions discussed in the previous section were assumed to be simple functions; that is, functions generated by a transducer with only one output. Many functions require either combinations of transducers or multi-output transducers. Strain-gage bridges, for example, have four outputs. Impact and static air pressure, measured separately, may be combined to give airspeed. Accelerometer outputs may be integrated, combined, and transformed to yield position. These are only some of the many possibilities.

There are two distinct types of problems that arise when functions are combined. The first is usually called phase error. Phase error occurs when two or more of the functions to be combined are sampled at slightly different intervals.

Example 2.3 Consider the situation shown in Fig. 2.6. Suppose it is desired to record y as a time history. Then from a recording of x and z, y could be reconstructed as follows:

$$y(t) = z(t) - x(t). \tag{2.21}$$

If x is consistently recorded with a phase error τ, then the computed quantity will have the form

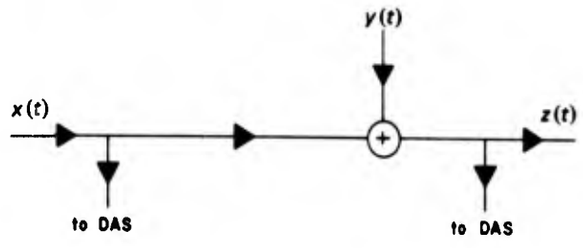


Fig. 2.6. Generation of a phase error as described in Example 2.3.

$$z(t) - x(t-\tau) = y(t) + [x(t) - x(t-\tau)]. \tag{2.22}$$

The quantity within the brackets is an error term. As an example, suppose x were a sinusoid, say $A \sin \omega t$. Then the error would have the form

$$x(t) - x(t-\tau) = A\,[\sin \omega t - \sin(\omega t - \omega\tau)]$$

$$= 2A \sin \frac{\omega\tau}{2} \cos\left(\omega t - \frac{\omega\tau}{2}\right). \tag{2.23}$$

As the size of τ increased, the error term would correspondingly become larger until it reached a relative maximum at τ equal to $\pi/(2\omega)$.

Phase errors such as the one described in the example do occur in tests. They may result from the type of multiplexer used, from long lines leading from the transducer to the multiplexer, or from attenuation in the system which varies with frequency. Correction for phase error can be made. One method uses Eq. (1.60) from Chapter 1. In other cases, the final result, such as cross spectra, can be compensated as will be shown in Chapter 6. In many cases, no correction is required. That would be true if PSD's, single probability density functions, etc., of uncombined functions were the only items required.

The best place to eliminate the phase error is at its source in the DAS rather than in the data reduction after the data have been collected. Equipment corrections are frequently difficult to make. The DAS usually was designed long before the testing began, making major revisions impossible. It is an unfortunate fact that many DAS's have been built without a proper analysis of the test requirements.

The second problem encountered when combining data is that one or more of the functions may require integration or differentiation. Both operations may be examined using Fourier transforms. If X is the Fourier transform of x, then

$$\mathcal{F}[\dot{x}(t)] = j2\pi f X(f)$$

$$\mathcal{F}\int_{-\infty}^{t} x(t')\,dt' = \frac{X(f)}{j2\pi f}. \tag{2.24}$$

That is, differentiating or integrating $x(t)$ corresponds respectively to multiplying or dividing $X(f)$ by $j2\pi f$.

It is possible to analyze digital integration and differentiation procedures using a similar technique as, for example, in Salzer [11]. Two simple, numerical expressions for the derivative v_i and the integral u_i of x_i are

$$u_i = u_{i-1} + x_i \Delta t$$
$$v_i = \frac{(x_i - x_{i-1})}{\Delta t}. \tag{2.25}$$

Taking the Fourier transform of these yields

$$\mathcal{F}[u_i] = \frac{\Delta t}{1 - e^{-j2\pi f \Delta t}} \mathcal{F}[x_i]$$
$$\mathcal{F}[v_i] = \frac{1 - e^{-j2\pi f \Delta t}}{\Delta t} \mathcal{F}[x_i]. \tag{2.26}$$

In the limit as Δt goes to zero, these expressions approach those given in Eq. (2.24). There is, however, a built-in error for finite Δt. For example, at the folding frequency, $f = 1/(2\Delta t)$, the ratio of the actual function to the desired one is

Integration	$j\frac{\pi}{2}$
Differentiation	$\frac{2}{j\pi}$.

(2.27)

That is shown in Fig. 2.7, where the modulus of the two operators for both the discrete and continuous cases are compared.

The numerical integration method described above tends to amplify bias errors and low frequency noise, turning the two into some sort of low frequency drift or linear trend. Clearly, if just the PSD of the integral of $x(t)$ were required, the simplest thing would be to find the PSD of $x(t)$ and operate on that to produce the PSD of the integral of $x(t)$. However, suppose that the PSD of z is desired, where $x(t)$ is defined by

$$z(t) = x(t) + \int_{-\infty}^{t} y(\tau)\, d\tau. \tag{2.28}$$

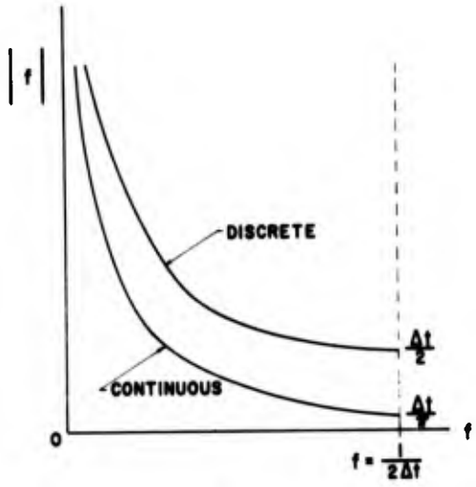
The PSD of $z(t)$ may be shown to be

$$G_z(f) = G_x(f) + \frac{1}{(2\pi f)^2} G_y(f) + 2C_{x\int y}(f). \tag{2.29}$$

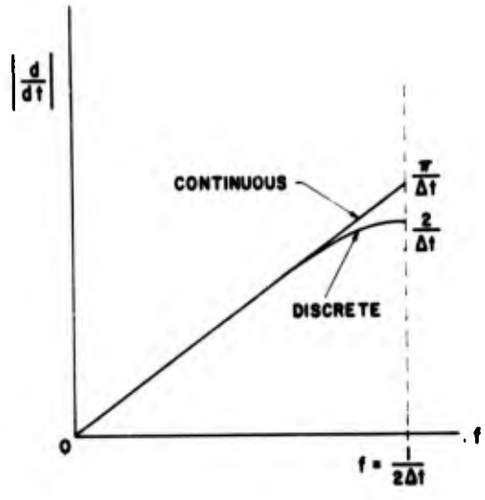
Thus, the PSD of G_z is the sum of the PSD of $x(t)$ and $y(t)$ (the latter multiplied by a term to account for the integration) plus a cross spectrum term. Specifically, $C_{x\int y}(f)$ is

$$C_{x\int y}(f) = 2\int_{-\infty}^{\infty} \cos 2\pi f\tau \left\{ \lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} x(t+\tau) \left[\int_{-\infty}^{t} y(t')\,dt' \right] dt \right\} d\tau$$

$$\text{for } f \geqslant 0. \qquad (2.30)$$



(a)



(b)

Fig. 2.7. (a) Comparison of continuous and discrete integration, and (b) comparison of continuous and discrete differentiation

If x and y are uncorrelated, this expression will be equal to zero. In that case, G_z could be computed as the sum

$$G_z(f) = G_x(f) + \frac{1}{(2\pi f)^2} G_y(f). \tag{2.31}$$

Thus, the integral of y would not have to be computed. However, if there is correlation between the two, then y must be integrated and added to x to form z. This means that trend error may be introduced into the expression for z, thus contaminating the PSD of z.

A related problem occurs with numerical differentiation. It has been found in practice that differentiation amplifies high frequency components, in many cases to an excessive extent. Both problems can be alleviated by removing a trend or altering the form of integration or differentiation. Other expressions for the latter operations will be discussed in Chapter 3. Trend removal is taken up in the following section.

2.5 Trend Removal

It is sometimes necessary to remove a linear or slowly varying trend from a particular time history. Such trends are likely to be found in the data, for example, when one or more of the components has been integrated. Integration introduces two types of errors. First of all, if the calibration zero point was incorrect, there will be a small error term with each time sample. When integrated, this constant term will become a straight line. Such a linear trend may produce large errors in PSD and related calculations.

The second type of error comes about because the integration procedure tends to amplify power corresponding to low frequency noise. Such noise is usually present in the data. When integrated, it takes the form of a random but slowly varying trend. Just how rapidly the trend varies depends to some extent upon the sampling interval.

The varying trend can best be removed by using high-pass filtering, a topic to be discussed in the next chapter. Polynomial trends may be removed using least square techniques. That is done in the following manner. Suppose, as usual, that x_i is a function sampled at the constant interval Δt for $i = 0, \cdots, N$, and that it is desired to fit a polynomial of the form

$$\hat{x}_i = \sum_{k=0}^{K} c_k (i\Delta t)^k. \tag{2.32}$$

The set of data points $\{x_i\}$ is an estimate of the polynomial content of x_i for polynomials of degree less than or equal to K. The usual procedure consists of defining an intermediate function, G, of the polynomial coefficient terms:

$$G(\mathbf{c}) = \sum_{i=0}^{N} \left[x_i - \sum_{k=0}^{K} c_k (i\Delta t)^k \right]^2 . \tag{2.33}$$

G is always positive for any value of $\mathbf{c} = (c_0, \cdots, c_k)$. It may be minimized using standard techniques from differential calculus. Taking derivatives of Eq. (2.33) with respect to c_j and setting them equal to zero,

$$\frac{\partial G}{\partial c_j} = \sum_{i=0}^{N} 2 \left[x_i - \sum_{k=0}^{K} c_k (i\Delta t)^k \right] [-(i\Delta t)^j] = 0. \tag{2.34}$$

Rearranging terms yields K equations of the form

$$\sum_{k=0}^{K} c_k \sum_{i=0}^{N} (i\Delta t)^{k+j} = \sum_{i=0}^{N} x_i (i\Delta t)^j, \quad j = 0, \cdots, K. \tag{2.35}$$

For large K values, it becomes tedious to solve for c. Fortunately, the need for solutions for K larger than 3 or 4 are rare. If a program is only to remove low order polynomials, then the computer memory that would be used to solve the above by means of matrix inversion techniques can be used for a direct calculation of the coefficients. For example, the following solutions are found for $K = 0$,

$$c_0 = \frac{1}{N+1} \sum_{i=0}^{N} x_i, \tag{2.36a}$$

and for $K = 1$,

$$\begin{aligned} c_0 &= \frac{2(2N+1) \sum_{i=0}^{N} x_i - 6 \sum_{i=0}^{N} i x_i}{(N+1)(N+2)} \\ c_1 &= \frac{6 \left(2 \sum_{i=0}^{N} i x_i - N \sum_{i=0}^{N} x_i \right)}{\Delta t N (N+1)(N+2)} . \end{aligned} \tag{2.36b}$$

For an odd number of points, a tremendous simplification in the formulas can be made. The procedure assumes that $i = -(N/2), \cdots, 0, \cdots, N/2$. Thus,

the odd power summation expressions in $(i\Delta t)$ will be zero, thereby eliminating many terms in the calculations. The formulas are

for $K = 1$,

$$c_0 = \frac{\Sigma x_i}{N}$$

$$c_1 = \frac{\Sigma i x_i}{\Sigma i^2 \Delta t}$$

where

$$\Sigma \equiv \sum_{i=-N/2}^{N/2} ;$$

for $K = 2$,

$$c_0 = \frac{\Sigma i^2 \Sigma i^2 x_i}{(\Sigma i^2)^2 - N\Sigma i^4}$$

$$c_1 = \frac{\Sigma i x_i}{\Sigma i^2 \Delta t}$$

$$c_2 = \frac{\Sigma i^2 \Sigma x_i - N\Sigma i^2 x_i}{[(\Sigma i^2)^2 - N\Sigma i^4]\,(\Delta t)^2};$$

for $K = 3$,

$$c_0 = \frac{\Sigma x_i \Sigma i^4 - \Sigma i^2 \Sigma i^2 x_i}{N\Sigma i^4 - (\Sigma i^2)^2}$$

$$c_1 = \frac{\Sigma i^4 \Sigma i^3 x_i - \Sigma i^6 \Sigma i x_i}{[(\Sigma i^4)^2 - \Sigma i^2 \Sigma i^6]\,\Delta t}$$

$$c_2 = \frac{\Sigma i^2 \Sigma x_i - N\Sigma i^2 x_i}{[(\Sigma i^2) - N\Sigma i^4]\,(\Delta t)^2}$$

$$c_3 = \frac{\Sigma i^4 \Sigma i x_i - \Sigma i^2 \Sigma i^3 x_i}{[(\Sigma i^4)^2 - \Sigma i^2 \Sigma i^6]\,(\Delta t)^3};$$

and for $K = 4$,

$$c_0 = \frac{\Sigma x_i - \Delta t^2 b_2 \Sigma i^2 - \Delta t^4 b_4 \Sigma i^4}{N}$$

$$c_1 = \frac{\Sigma_i x_i [\Sigma i^2 \Sigma i^6 - (\Sigma i^4)^2] - \Sigma i^4 (\Sigma i^2 \Sigma i^3 x_i - \Sigma i^4 \Sigma i x_i)}{\{\Sigma i^2 [\Sigma i^2 \Sigma i^6 - (\Sigma i^4)^2]\} \Delta t}$$

$$c_2 = \frac{\Sigma i^2 \Sigma x_i - N \Sigma i^2 x_i - \Delta t^4 b_4 (\Sigma i^2 \Sigma i^4 - N \Sigma i^6)}{[(\Sigma i^2)^2 - N \Sigma i^4] (\Delta t)^2}$$

$$c_3 = \frac{\Sigma i^2 \Sigma i^3 x_i - \Sigma i^4 \Sigma i x_i}{[\Sigma i^2 \Sigma i^6 - (\Sigma i^4)^2] (\Delta t)^3}$$

$$c_4 = \frac{(N \Sigma i^4 x_i - \Sigma i^4 \Sigma x_i)[(\Sigma i^2)^2 - N \Sigma i^4] + (\Sigma i^2 \Sigma x_i - N \Sigma i^2 x_i)(\Sigma i^4 \Sigma i^2 - N \Sigma i^6)}{\{[\Sigma i^2 \Sigma i^4 - N \Sigma i^6]^2 - [(\Sigma i^4)^2 - N \Sigma i^8][(\Sigma i^2)^2 - N \Sigma i^4]\} (\Delta t)^4};$$

where

N = Number of equally spaced samples of x (N must be odd)

$$\Sigma i^2 = \frac{N(N^2 - 1)}{12}$$

$$\Sigma i^4 = \frac{N(N^2 - 1)(3N^2 - 7)}{240}$$

$$\Sigma i^6 = \frac{N(N^2 - 1)(3N^4 - 18N^2 + 31)}{1344}$$

$$\Sigma i^8 = \frac{N(N^2 - 1)(5N^6 - 55N^4 + 239N^2 - 381)}{11520}$$

$$t_i = i\Delta t, \; i = -\frac{N}{2}, \cdots, \frac{N}{2}. \tag{2.37}$$

Underflow or overflow is quite possible when computing terms of the form $\Sigma i_j x_i$, especially on machines with limited word size. The user should consider doing the summation portion of the calculation in double precision. To clarify the nature of the difficulty, consider the following example.

Example 2.4 Suppose it is desired to calculate the sample mean of x_i where i ranges from 1 to 2,000. Furthermore, suppose that x_i is approximately unity for each i. The hypothetical computer on which this calculation is to be performed is assumed to have five decimal digits in each word. That is, the word, the basic unit of calculation, is composed of five individual decimal digits. However, to simplify the positioning of the decimal place, the hypothetical machine uses floating point arithmetic. In this mode, each such five-digit word is broken into two parts. The first, with two digits, is essentially an exponent of ten, while the second part carries the significant digits. In this particular machine, the floating-point representation for x would be

xx	xxx
A	B
in	in
2 digits	3 digits

$$x = B \cdot 10^{A-50}. \tag{2.38}$$

Thus, the five digits in storage 50100 would represent 1.00, and 46327 would be equivalent to 0.000327. If these two numbers were added by the machine, using special floating-point addition, the result would be 1.00. That is, the smaller number would effectively be shifted out the low end of the add register. This is termed *underflow.*

Continuing with the example, if 2,000 units were added to this machine, the result would not be 2,000, but 1,000 because after summing up the first thousand, the units thereafter would underflow. The computed sample mean would be 0.5 rather than 1.0, a significant error. If the summation were performed in double precision, there would have been no error in the mean value computed in the example. Double precision is a capability found in most machines (either in the hardware or in the software, usually the latter for small computers), which combines two storage units into a single one. This yields twice as many significant digits and a considerably extended range for the exponent of the floating-point number.

CHAPTER 3
DIGITAL FILTERING

3.1 Basic Concepts

The principal use of filtering in data reduction is to smooth the data being analyzed. The first smoothing filters used were of the moving average type. These were followed by polynomial filters, i.e., those which pass polynomials without change while attenuating jitter. Ormsby [12] in 1959 introduced a low-pass filter that showed marked improvement over the polynomial type. Later writers have introduced recursive or feedback filters showing significant advancement in performance over previous filters. These latter types will be emphasized in this work. Digital filters are frequently analyzed using z-transform methods. This technique has not been used in this chapter because it does not describe filter action in terms familiar to the data user.

Broadly speaking, a filter is a device or a physical process that operates on a time history and (usually) changes the time history in some manner. This definition would include attenuation networks, amplifiers, nonlinear devices of all sorts, as well as the trivial filter that transmits the time history without change. Linear filters are the single, most important, category. There are a number of ways by which this type of filter could be defined. The most basic definition is through the unit impulse response function and the convolution integral. Let x be the unfiltered time history and y be the result of the action of the filter upon x. Suppose that if $x(t) = \delta(t)$, then $y(t) = h(t)$. That is, if the only input to the filter is a single delta function at time $t = 0$, then the output of the filter is the unique function $h(t)$, the unit impulse response function. The general relationship between the output from the filter, y, and the input to the filter for any x is then given by

$$y(t) = \int_{-\infty}^{\infty} h(\tau)\, x(t-\tau)\, d\tau. \tag{3.1}$$

This convolution of $h(t)$ and $x(t)$ produces $y(t)$. The Fourier or Laplace transform of $h(t)$ yields the system transfer function.

Example 3.1 If the filter in question is of the form shown in Fig. 3.1, where $x(t)$ and $y(t)$ are, respectively, the input and output voltages, then the unit impulse response function is

$$h(t) = \begin{cases} \dfrac{1}{RC}\, e^{-t/RC} & t \geqslant 0 \\ 0 & t \leqslant 0. \end{cases}$$

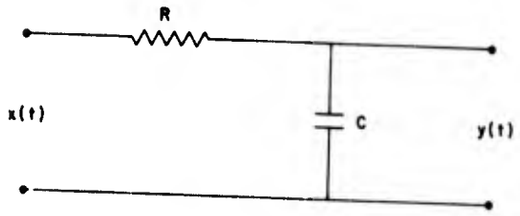


Fig. 3.1. RC filter.

For that particular function, the limits of the convolution can be simplified somewhat if $x(t) = 0$ for $t < 0$,

$$y(t) = \frac{1}{RC}\int_0^t e^{-(t-\tau)/RC}\, x(\tau)\, d\tau$$

$$= \frac{1}{RC}\int_0^t e^{-\tau/RC}\, x(t-\tau)\, d\tau \qquad t \geqslant 0. \tag{3.3}$$

The relationship between $x(t)$ and $y(t)$ could have been expressed also as a differential equation,

$$RC\dot{y}(t) + y(t) = x(t). \tag{3.4}$$

The frequency response function of the filter, $H(f)$, is the Fourier transform of $h(t)$,

$$H(f) = \int_{-\infty}^{\infty} h(t)\, e^{-j2\pi ft}\, dt. \tag{3.5}$$

Thus, for the function cited in Example 3.1,

$$H(f) = \frac{1}{1 + jRC2\pi f} \tag{3.6}$$

The usefulness of this function arises from the simplification that occurs after taking Fourier transforms. The Fourier transformation of both sides of Eq. (3.1) will yield

$$Y(f) = H(f)\, X(f). \tag{3.7}$$

That is, the convolution becomes a product after Fourier transformation. The inverse of this is discussed in Example 1.4. Taking absolute values of both sides of Eq. (3.7) results in

$$|Y(f)| = |H(f)||X(f)|. \tag{3.8}$$

Setting this aside for a moment, consider having as the input (in this case, from $t = -\infty$ to $t = \infty$) a sinusoidal function of the form

$$X(t) = A\cos(\omega_0 t + \phi), \qquad A > 0. \tag{3.9}$$

The Fourier transform of this function is

$$X(f) = \frac{A}{2}\left[e^{j\phi}\delta(f - f_0) + e^{-j\phi}(f + f_0)\right], \tag{3.10}$$

where $f_0 = \omega_0/2\pi$. When this result is combined with Eq. (3.7), then the output from the filter is given by

$$Y(f) = H(f)\frac{A}{2}\left[e^{j\phi}\delta(f - f_0) + e^{-j\phi}\delta(f + f_0)\right]. \tag{3.11}$$

Since the right-hand side is zero except where $f = \pm f_0$, this expression can be written as

$$Y(f) = \left[H(f_0)\,\frac{A}{2}\,e^{j\phi}\right]\delta(f - f_0) + \left[H(-f_0)\,\frac{A}{2}\,e^{-j\phi}\right]\delta(f + f_0). \tag{3.12}$$

In other words, if the input to the filter is a sinusoid, then, disregarding transients, so is the output, as the right-hand side of Eq. (3.12) is also a sinusoidal function. Furthermore, the ratio of the amplitudes of the input and output is given by the absolute value of the transfer function evaluated at the frequency of the input sine wave. For this reason, $|H(f)|$ is called the gain factor of the filter.

Example 3.2 If $H(f)$ has the form given in Example 3.1, that is, as expressed in Eq. (3.6), then the above result will take the form

$$|Y(f)| = \frac{A}{\sqrt{1 + T^2(2\pi f)^2}}\,\delta(f - f_0). \tag{3.13}$$

As is usual, $RC = T$, a time constant.

Another useful attribute of the transfer function is its action on the power spectral density of the input to the filter. As will be recalled, $X_T(f)$ is defined by

$$X_T(f) = \int_{-T/2}^{T/2} x(t)\, e^{-j2\pi ft}\, dt. \tag{3.14}$$

Then the PSD of $x(t)$, $G_x(f)$ of $x(t)$, is defined by Eq. (1.39) as the limit

$$G_x(f) = 2 \lim_{T\to\infty} \frac{1}{T} |X_T(f)|^2, \quad f \geqslant 0$$

$$= 0, \qquad f < 0. \tag{3.15}$$

There are two other avenues of arriving at a definition of $G_x(f)$, as described in Chapter 1. One method uses the autocorrelation function as an intermediate step, and the other uses a bandpass filter followed by squaring and integration. All three definitions are equivalent under limiting circumstances.

The relationship between the input and output spectra of a filter is given by

$$G_y(f) = G_x(f)|H(f)|^2, \quad f \geqslant 0. \tag{3.16}$$

Thus, if $x(t)$ is white noise so that

$$G_x(f) = \begin{cases} 1 & f \geqslant 0 \\ 0 & f < 0, \end{cases} \tag{3.17}$$

then

$$G_y(f) = |H(f)|^2, \quad f \geqslant 0. \tag{3.18}$$

That is, if white noise is the input to a filter, the output takes the form of the absolute value squared of the transfer function. White noise as defined by Eq. (3.17) is actually a physical impossibility. This is because the variance of a function is given by the integral of its PSD,

$$\sigma_x^2 = \int_0^\infty G_x(f)\, df. \tag{3.19}$$

For white noise, σ_x^2 would be infinite. Because the concept of white noise is useful, the idea is retained by redefining the white noise function, hereafter denoted by $N_B(f)$:

$$N_B(f) = \begin{cases} 1 & 0 \leq f \leq B \\ 0 & \text{otherwise.} \end{cases} \tag{3.20}$$

When B is greater than the highest frequency of interest for an intended application, then $N_B(f)$ effectively has the characteristics of white noise and still has finite variance, thus avoiding the infinite energy and power problem.

A few general comments still need to be made about filters before passing on to the digital variety. The term *realizable filter* is frequently used to denote a filter for which $h(t)$ is zero if $t < 0$. Such a filter does not anticipate future events and, in many cases, may be fabricated using standard components, such as resistors, capacitors, and inductors if $x(t)$ is a voltage or spring-mass-dashpot systems if $x(t)$ is a mechanical displacement. Put another way, a filter must be realizable if it is to be used in real time. However, if $x(t)$ is not being filtered in real time, then the filter does not have to be realizable in the sense described above. In fact, it may be convenient to use a two-sided filter, i.e., one for which $h(t)$ may not be zero for $t < 0$. Again, if the data are recorded, it may be useful to run time backwards and filter the data in reverse order. As will be seen, this type of manipulation is very easy to do if the data are digital and stored in a computer.

There are certain types of filters referred to many times in applications. It is easier to discuss first idealized forms of these filters in comparison with the definitions.

An ideal *low-pass filter* is one for which $H(f)$ has the following property:

$$|H(f)| = \begin{cases} 1 & 0 \leq f \leq f_0 \\ 0 & f_0 < f \leq 1/2\Delta t. \end{cases}$$

Similarly, an ideal *high-pass filter* has a transfer function for which

$$|H(f)| = \begin{cases} 0 & 0 \leq f \leq f_0 \\ 1 & f_0 < f \leq 1/2\Delta t. \end{cases}$$

An ideal *bandpass filter* has the form

$$|H(f)| = \begin{cases} 0 & 0 \leq f < f_\ell \\ 1 & f_\ell \leq f \leq f_r \\ 0 & f_r < f \leq 1/2\Delta t. \end{cases}$$

An ideal *notch filter* has characteristics such that

$$|H(f)| = \begin{cases} 1 & 0 \leq f < f_\ell \\ 0 & f_\ell \leq f \leq f_r \\ 1 & f_r < f \leq 1/2\Delta t. \end{cases}$$

These filters are shown in Fig. 3.2.

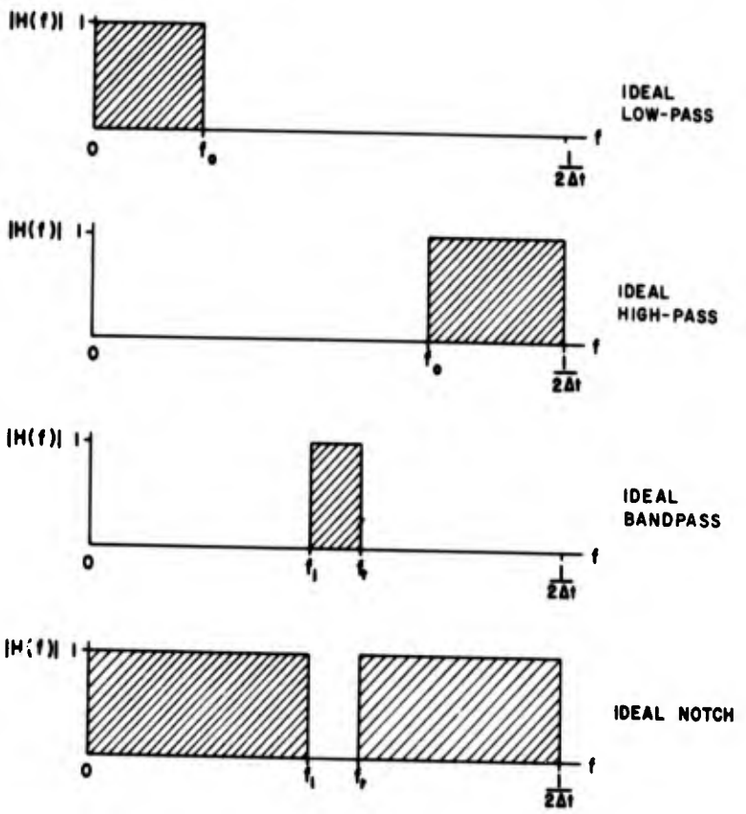


Fig. 3.2. Ideal filters.

It is not possible to have finite weighting functions that will exactly implement the ideal forms of these filters. In practice, filters are used that approximate the performance of their ideal counterpart. For example, almost any filter for which $H(f)$ is decreasing on the range $(0, 1/2\Delta t)$ is called a low-pass filter.

3.2 Nonrecursive Digital Filters

Most of the original work done in numerical filtering concerned nonrecursive filters, that is, those having the form

$$y_i = \sum_{k=M_1}^{M_2} h_k x_{i+k}. \tag{3.21}$$

Thus, there are $(M_2 + 1 - M_1)$ coefficients of the form h_k. Equation (3.21) is the numerical equivalent of

$$y(t) = \int_{-\infty}^{\infty} h(\tau)\, x(t-\tau)\, d\tau. \tag{3.22}$$

Equation (3.21) can be regarded as the numerical approximation of Eq. (3.1) rather than an analogy. The Δt term is usually absorbed into the set $\{h_k\}$. These terms are referred to as the filter weights. The filter described by Eq. (3.21) is symmetric when $h_{-k} = h_k$. It is actually not necessary, and in fact, some of the filters examined subsequently will not be symmetric.

For example, one of the simplest instances of this type of filter is the differentiating filter described in Chapter 2. It, of course, is nonsymmetric.

Taking the Fourier transform of Eq. (3.21) yields

$$Y(f) = X(f) \sum_{k=M_1}^{M_2} h_k\, e^{-j2\pi\Delta tkf}. \tag{3.23}$$

The transfer function, obtained by taking the ratio of Y to X, is

$$H(f) = \sum_{k=M_1}^{M_2} h_k\, e^{-j2\pi\Delta tkf}$$

$$= \sum_{k=M_1}^{M_2} \left\{ [h_k \cos(2\pi\Delta tkf)] - j\,[h_k \sin(2\pi\Delta tkf)] \right\}. \tag{3.24}$$

Thus, the modulus and phase angle are given by

$$|H(f)| = \sqrt{\left(\sum_{k=M_1}^{M_2} h_k \cos(2\pi\Delta tkf) \right)^2 + \left(\sum_{k=M_1}^{M_2} h_k \sin(2\pi\Delta tkf) \right)^2}$$

$$\Phi(f) = \arctan \left[\frac{\displaystyle\sum_{k=M_1}^{M_2} h_k \sin(2\pi\Delta tkf)}{\displaystyle\sum_{k=M_1}^{M_2} h_k \cos(2\pi\Delta tkf)} \right]. \tag{3.25}$$

In terms of $|H(f)|$ and $\Phi(f)$, $H(f)$ may be written

$$H(f) = |H(f)| \exp [-j\Phi(f)]. \tag{3.26}$$

Filter performance is usually expressed using some form of Eq. (3.25). Separate plots of $|H(f)|$ or $|H(f)|^2$ and $\Phi(f)$ with respect to frequency are more or less the standard method of examining the filtering action. Various expressions for $\Phi(f)$ that differ from the above in the manner of defining the sign are sometimes found. If the set of weights is symmetric about h_0 so that

$$h_k = h_{-k} \qquad \text{for all } k, \tag{3.27}$$

then the expression

$$\sum_{k=-M}^{M} h_k \sin (2\pi\Delta tkf) \tag{3.28}$$

is identically zero. It follows immediately that $\Phi(f)$ is identically zero in this case for all f. It would at first seem that the computing cost would be doubled when going from a nonsymmetric to a symmetric filter to obtain the zero phase characteristic. On closer examination, it turns out that, with proper programming, the number of multiplications need not increase at all. Thus, only M additional additions must be performed, perhaps requiring only a 25-percent increase in computing time.

Example 3.3 A binomial filter* is one whose weight has binomial† coefficients

$$h_{k-N/2} = \binom{N}{N-k} \frac{1}{2^N}, \qquad k = 0, \cdots, N, N \text{ even}. \tag{3.29}$$

The transfer function is

$$H(f) = \frac{Y(f)}{X(f)} = \frac{1}{2^N}\left[\binom{N}{0} e^{j2\pi\Delta t \frac{N}{2} f} + \cdots + \binom{N}{\frac{N}{2}} + \cdots + \binom{N}{N} e^{-j2\pi\Delta tf\frac{N}{2}}\right]$$

$$= \frac{1}{2^N}\left[e^{j\pi\Delta tf} + e^{-j\pi\Delta tf}\right]^N = \frac{1}{2^N}\left[2\cos(\pi\Delta tf)\right]^N = \cos^N(\pi\Delta tf). \tag{3.30}$$

*Origin unknown.

†As usual, $\binom{p}{q} = \dfrac{p!}{(p-q)!q!} = \dfrac{p(p-1)\cdots(p-q+1)}{1,2,\cdots,q}$.

When plotted, the result resembles Fig. 3.3. The binomial filter is one example of a low-pass filter. One important parameter for any low-pass filter is its half-power frequency. This frequency, here labeled f_{hp}, is the frequency for which $|H(f)|^2$ has been reduced to one-half of its maximum value, which occurs at $f = 0$. For the binomial filter,

$$f_{hp} = \frac{1}{\pi \Delta t} \arccos \left[\left(\frac{1}{2} \right)^{1/2N} \right]. \tag{3.31}$$

Equation (3.31) indicates that increasing N decreases f_{hp} and hence moves the half-power point to the left.

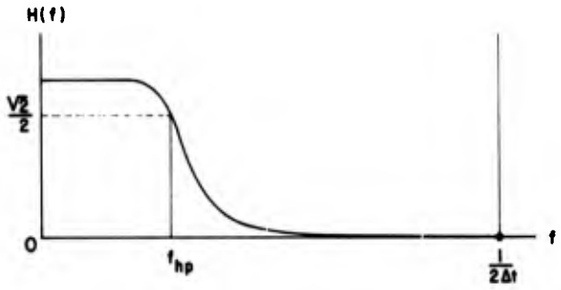


Fig. 3.3. Transfer function of the binomial filter.

The binomial filter is not presently employed to any great extent for several good reasons. It is not very flexible, in that it is extremely limited as to the placement of the half-power point. Another problem is that its filtering action is not sharp and is far from being a good or even a fair approximation of that of the ideal low-pass filter. On the positive side, it does have good characteristics in its action on polynomials. As will be seen later in the chapter, it will pass an Nth-order polynomial without attenuation.

A much more usable result is due to Ormsby [12] who developed a set of numerical low-pass and bandpass filters, collectively referred to as Ormsby filters. The numerical procedure is as follows. The particular filter shape desired is defined in the frequency plane on the range $-1/(2\Delta t) \leqslant f \leqslant 1/(2\Delta t)$ and is made to be symmetric about $f = 0$. The inverse transform is evaluated (using a computer) for $t = k\Delta t$, $k = 0, \cdots, m$. The value chosen for m is usually less than 100. The inverse transform terms thus obtained are then multiplied by Δt and used for the filter weights.

Example 3.4 Suppose an ideal low-pass filter of the Ormsby type is required where $H(f)$ is defined as

$$H(f) = \begin{cases} 1 & 0 \leq f \leq f_c \\ 0 & f_c < f \leq 1/2\Delta t. \end{cases} \tag{3.32}$$

Then the $\{h_k\}$ are given by

$$h_k = \Delta t \int_{-1/(2\Delta t)}^{1/(2\Delta t)} H(f)\, e^{j2\pi fk\Delta t}\, df = \Delta t \int_{-f_c}^{f_c} \cos(2\pi fk\Delta t)\, df$$

$$= \frac{\Delta t}{2\pi k\Delta t}\left[\sin(2\pi fk\Delta t)\right]_{-f_c}^{f_c} = \frac{1}{\pi k}\sin(2\pi f_c k\Delta t), \qquad k = 0, \cdots, m. \tag{3.33}$$

As $h_k = h_{-k}$, the weights for $k = -1, \cdots, -m$ do not need to be calculated. Because the set of weights is truncated at $k = \pm m$, there is an error between the desired $H(f)$ as given by Eq. (3.32) and the inverse of the truncated set. The filter actually used will only approximate Eq. (3.32), with the approximation tending to improve with increasing m.

Ormsby improved the performance of these filters for fixed values of m by adding a term in the frequency domain which makes the transition from one to zero less acute. Specifically, he defined a filter of the form

$$H(f) = \begin{cases} 1 & 0 \leq f < f_c \\ \dfrac{f - f_d}{f_c - f_d} & f_c \leq f \leq f_d \\ 0 & f_d < f \leq 1/(2\Delta t). \end{cases} \tag{3.34}$$

This filter is shown in Fig. 3.4.* The portion from f_c to f_d is the only new feature. The transform of this portion, when found, can simply be added to the previous result. The transform of the triangular portion is given by

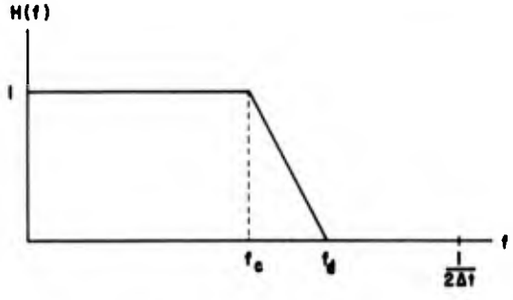


Fig. 3.4. The Ormsby low-pass filter.

*Ormsby was more general than indicated above. He allowed the $[f - f_d)/(f_c - f_d)]$ to be raised to a power p. The case examined here is for $p = 1$, probably the most common in actual practice.

$$\Delta t \left\{ \int_{-f_d}^{-f_c} \frac{f+f_d}{-f_c+f_d} e^{-j2\pi fk\Delta t}\, df + \int_{f_c}^{f_d} \frac{f-f_d}{f_c-f_d} e^{-j2\pi fk\Delta t}\, df \right\}$$

$$= 2\Delta t \int_{f_c}^{f_d} \frac{f-f_d}{f_c-f_d} \cos(2\pi fk\Delta t)\, df$$

$$= \frac{2\Delta t}{f_c-f_d} \left[\frac{\cos(2\pi fk\Delta t)}{(2\pi k\Delta t)^2} + \frac{f \sin(2\pi fk\Delta t)}{(2\pi k\Delta t)} - f_d \frac{\sin(2\pi fk\Delta t)}{(2\pi k\Delta t)} \right]_{f_c}^{f_d}$$

$$= \frac{2\Delta t}{(f_c-f_d)(2\pi k\Delta t)} \left\{ \frac{1}{(2\pi k\Delta t)} [\cos(2\pi f_d k\Delta t) - \cos(2\pi f_c k\Delta t)] \right.$$

$$\left. -(f_c-f_d)\sin(2\pi f_c k\Delta t) \right\}. \tag{3.35}$$

Combining this with the result of Eq. (3.33) and simplifying yields a new set of filter weights,

$$h_k = \frac{4\Delta t}{(f_c-f_d)(2\pi k\Delta t)^2} \{\sin[\pi k\Delta t(f_c+f_d)] - \sin[\pi k\Delta t(f_c-f_d)]\}. \tag{3.36}$$

This set of weights is bounded by a term that contains $1/k^2$, whereas the simpler filter can be shown to be bounded by a term containing $1/k$. Thus, the absolute value of the weights tends to become smaller with increasing k in both cases. However, the bound for the expression for h_k given by Eq. (3.36) will drop off much more rapidly.

Ormsby also developed differentiating and bandpass filters using the same sort of technique. The differentiating filter is simply the low-pass filter with a factor of $2\pi f$ multiplying the integrand. His bandpass filter is essentially the inverse transform of the ideal bandpass filter with triangular portions added on at the sides to give improved performance.

Filters of the Ormsby type are widely used. While not as economical as the recursive types to be discussed in the following section, they do have other advantages. In particular, the low-pass filters have zero phase because of their symmetry, and they tend to pass polynomial functions without change. The reason for this will be discussed in a later section dealing with that problem.

The chief disadvantage of the Ormsby filter is economic; 50 to 100 weights are used in many cases where a 6-weight recursive filter could be employed. Also, the actual transfer functions tend to be "bumpy," that is, to exhibit a form of Gibbs phenomenon.

3.3 Recursive Filters

In the previous section, the filters discussed were simple or nonrecursive. Recursive filters, a family with a more complex structure, are examined in this section. The principal difference between the two types is that in the nonrecursive filters, the output is a finite sum of the input terms only, whereas in the recursive case, the previous output is also used as an input. In usual engineering terminology, such a procedure is called feedback. The most general expression for the variety of filter to be dealt with in the section takes the form

$$y_i = \sum_{k=M_1}^{M_2} h_k x_{i+k} + \sum_{\ell=1}^{M_3} g_\ell y_{i-\ell}. \tag{3.37}$$

This type is called a compound recursive filter. Graphically, interpreting the difference Δt as a delay, a filter of this form would be as shown in Fig. 3.5. The triangles represent multiplication by the value shown within the triangle, the rectangles with Δt represent a delay of Δt seconds, and the long elliptical shape with a plus sign represents an adder, i.e., an electronic device capable of adding the voltages corresponding to the various functions. Thus an operation such as that described by Eq. (3.37) can be realized either on a digital computer as a summation operation or in an analog mode, using electrical components.

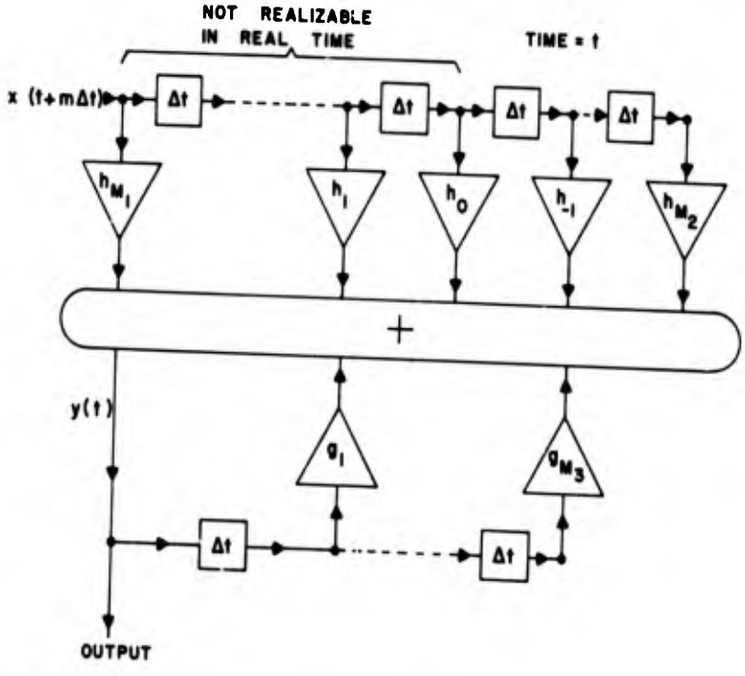


Fig. 3.5. Compound recursive filter.

A somewhat simpler form of the recursive filter is given by

$$y_i = cx_i + \sum_{k=1}^{M} h_k y_{i-k}. \tag{3.38}$$

The analog implementation of this is shown in Fig. 3.6. The majority of the standard filters to be examined will be of this variety. When the term recursive filter is used hereafter without other modifiers, it is this latter filter that is being discussed rather than the compound filter defined by Eq. (3.37).

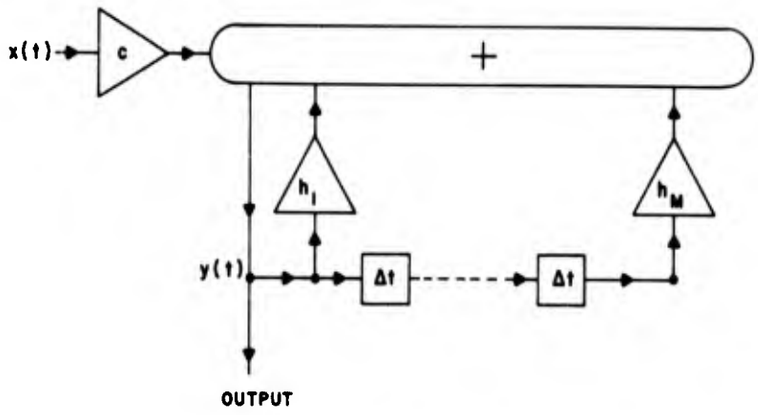


Fig. 3.6. Standard recursive filter.

One of the simplest recursive filters is

$$y_i = ay_{i-1} + (1-a)x_i, \tag{3.39}$$

where $a = e^{-(\Delta t/RC)}$, which is analogous to the low-pass RC filter in the continuous case. The modulus squared of the transfer function is

$$|H(f)|^2 = \left|\frac{Y(f)}{X(f)}\right|^2 = \frac{(1-a)^2}{(1+a^2) - 2a\cos(2\pi f \Delta t)}. \tag{3.40}$$

For a half-power point at $f = f_{hp}$, then

$$a = 2 - \cos(2\pi f_{hp}\Delta t) - \sqrt{\cos^2(2\pi f_{hp}\Delta t) - 4\cos(2\pi f_{hp}\Delta t) + 3} \tag{3.41}$$

This filter is convenient for computing averages. To show this more clearly, the defining relationship can be rewritten in the form

$$y_i = (1-a)\sum_{j=-\infty}^{i} a^{i-j}x_j.$$

If $\mathbf{E}[x_i] = u$, where the E denotes expectation, then

$$\mathbf{E}[y_i] = \mathbf{E}\left[(1-a)\sum_{j=-\infty}^{i} a^{i-j}x_j\right] = \mu\left[(1-a)\sum_{j=-\infty}^{i} a^{i-j}\right]$$

$$= \mu\left[(1-a)\sum_{j=0}^{\infty} a^{j}\right] = \mu\left[(1-a)\frac{1}{(1-a)}\right] = \mu. \qquad (3.42)$$

Thus, this process may be used to estimate the mean of $\{x_i\}$ with a time-weighting characteristic much like the RC filter. The procedure can be extended to estimating mean-square values.

Another problem is the following: After how many intervals will an individual pulse be reduced to one-half its original output height? Suppose the following conditions occur:

$$y_{-1} \equiv 0$$

$$x_i = \begin{cases} 1 & i = 0 \\ 0 & i > 0. \end{cases} \qquad (3.43)$$

Then y_i will have the following form:

$$y_i = a^i(1-a) \qquad (3.44)$$

and

$$\frac{y_i}{y_0} = a^i. \qquad (3.45)$$

Thus the value of k for which this ratio will reach one-half is given by

$$a^k = \frac{1}{2}$$

$$a = \left(\frac{1}{2}\right)^{1/k}. \qquad (3.46)$$

Put the other way,

$$k = -\frac{\ell n 2}{\ell n a}. \tag{3.47}$$

The k so derived is the half-life of the filter; for the given a, in k steps the output will be reduced to one-half of what it was originally.

As seen in Eq. (3.45), y_i decreases geometrically. The summation of all such $\{y_i\}$ terms is one. Another possible criterion would be to find the k for which the sum of the first k terms is one-half. That is,

$$\sum_{i=0}^{k-1} a^i(1-a) \approx \frac{1}{2}. \tag{3.48}$$

This would seem to be another form of half-life. Equation (3.48) can be re-written as

$$\sum_{i=0}^{k-1} a^i(1-a) = (1-a)\sum_{i=0}^{k-1} a^i = (1-a)\left[\frac{1-a^k}{1-a}\right] = 1 - a^k \approx \frac{1}{2}, \tag{3.49}$$

which yields

$$a^k = \frac{1}{2}$$

$$a = \left(\frac{1}{2}\right)^{1/k} \tag{3.50}$$

Conveniently, this is the same as Eq. (3.46).

Finally, returning to the topic of half-power points, the frequency at which the power is reduced to one-half is given by

$$f_{hp} = \frac{1}{2\pi\Delta t}\cos^{-1}\left[1 - \frac{(1-a)^2}{2a}\right], \quad a \geq 3 - 2\sqrt{2}. \tag{3.51}$$

For a less than $(3 - 2\sqrt{2})$, the absolute value squared of the transfer function is greater than one-half for all frequencies, so that there is no half-power point.

3.4 Second-Order Filters

The differential equation defining the second-order system is

$$\ddot{y} + 2\zeta(2\pi f_n)\,\dot{y} + (2\pi f_n)^2\, y = ax, \tag{3.52}$$

where f_n is the undamped natural frequency and ζ is the damping ratio. This relationship has a transfer function H whose modulus squared is

$$|H(f)|^2 = \frac{a^2}{(f^2 - f_n^2) + (2\zeta f_n f)^2}. \tag{3.53}$$

Defining $f_p = f_n\sqrt{1 - 2\zeta^2}$, we may show that the $|H(f)|^2$ has its peak value when $f = f_p$. A suitable normalizing value for $|H(f)|^2$ is

$$a = \sqrt{(f_n^2 - f_p^2)^2 + 4\zeta^2 f_n^2 f_p^2} = \sqrt{4f_n^2\zeta^2(1-\zeta^2)}. \tag{3.54}$$

This equation will yield unity gain when $f = f_p$. It also has been shown that the half-power bandwidth B of the transfer function, i.e., the distance between half-power points for small ζ, is given approximately by

$$B = 2f_n\zeta. \tag{3.55}$$

Therefore, if B and f_p are given and it is desired to find a bandpass filter with these given characteristics, then Eq. (3.52) can be used by solving for the proper values of f_n and ζ as follows:

$$f_n = \frac{f_p}{\sqrt{1 - 2\zeta^2}} \tag{3.56}$$

and

$$\zeta = \sqrt{\frac{B^2}{4f_p^2 + 2B^2}}. \tag{3.57}$$

For $f_n > B$,

$$\zeta \approx \frac{B}{2f_p}. \tag{3.58}$$

The digital equivalent of the differential equation describing the second-order filter is a difference equation of the form

$$y_i = cx_i + h_1 y_{i-1} + h_2 y_{i-2}, \quad i = 1, 2, \ldots. \tag{3.59}$$

The variables and parameters are defined as follows:

$$h_1 = 2\exp[-2\pi f_n \Delta t\zeta]\cos[2\pi f_n \Delta t\sqrt{1-\zeta^2}]$$

$$h_2 = -\exp[-4\pi f_n \Delta t\zeta]$$

f_n = undamped natural frequency of the system in hertz

ζ = damping ratio

c = a normalizing coefficient.

The values for h_1 and h_2 given above are obtained by techniques described more fully in the next section. This function will have its primary poles in the same locations in the frequency plane as the poles* of the system described in Eq. (3.52). Thus, the undamped natural frequency and the damping ratios of the two processes will be the same. However, there will be deviations between the two transfer functions in other respects, as will be shown.

The Fourier transform of Eq. (3.59) may be used to obtain the transfer function of the filter, $H(f)$. The amplitude value squared of the transfer function is

$$|H(f)|^2 = \frac{c^2}{[1-h_1\cos(2\pi f\Delta t)-h_2\cos(4\pi f\Delta t)]^2+[h_1\sin(2\pi f\Delta t)+h_2\sin(4\pi f\Delta t)]^2}. \tag{3.60}$$

After taking derivatives and equating to zero, we may show $|H(f)|^2$ to have a maximum where f satisfies the following relationship:

$$\cos(2\pi f\Delta t) = \frac{h_1(h_2-1)}{4h_2}. \tag{3.61}$$

Inserting the values for h_1 and h_2 given above and simplifying, we find that

$$\cos\lambda_p = \cos(\lambda_n\sqrt{1-\zeta^2})\cosh(\lambda_n\zeta), \tag{3.62}$$

where

$\lambda_p = 2\pi\Delta t f_p$

f_p = the peak (resonant) frequency

$\lambda_n = 2\pi\Delta t f_n$.

When $\zeta = 0$, then $\lambda_p = \lambda_n$ as would be expected. Equation (3.62) may be rewritten as

$$\cos\lambda_p = \cos\left(\lambda_n\sqrt{1-\zeta^2}\right)\cos(j\lambda_n\zeta)$$

$$= \frac{1}{2}\left\{\cos[\lambda_n(\sqrt{1-\zeta^2}+j\zeta)] + \cos[\lambda_n(\sqrt{1-\zeta^2}-j\zeta)]\right\}. \tag{3.63}$$

*See Section 3.5.

Note that the absolute value of the argument of the cosines in the above expression in both cases is simply λ_n. It is not possible to solve Eq. (3.62) explicitly for λ_n in terms of λ_p and ζ since it is a transcendental equation. In fact, there are no solutions for some values of λ_p and ζ.

It would be convenient if digital parameters could be determined as readily as those of the analog case. Approximations may be used for large Q, $Q = 1/2\zeta$, where the analog results are probably quite accurate. That is, if B and f_p are given, then f_n and ζ can be obtained from Eqs. (3.56) and (3.57). This can be shown by applying the approximation

$$\cos x \approx 1 - \frac{x^2}{2} \tag{3.64}$$

to Eq. (3.63). The result follows:

$$f_n \approx \sqrt{\frac{f_p{}^2 + (\zeta/4\pi\Delta t)^2}{1 - \zeta^2}} \tag{3.65}$$

For $f_p >> (\zeta/4\pi\Delta t)$, which corresponds to a high Q, then Eq. (3.65) reduces to Eq. (3.57).

A more general result can be obtained using pole placement. In Section 3.6, a general method for obtaining bandpass filters is given which reduces to the second-order case as the lowest order implementation. The pole piacement procedure makes it relatively easy to place the half-power points exactly where desired.

Also note that Eq. (3.60) may be written as

$$|H(f)|^2 = \frac{1}{(2e^{-\lambda_n \zeta})^2} \times \tag{3.66}$$

$$\frac{c^2}{[\cos\lambda - \cos(\lambda_n\sqrt{1-\zeta^2} + j\zeta\lambda_n)]\ [\cos\lambda - \cos(\lambda_n\sqrt{1-\zeta^2} - j\zeta\lambda_n)]},$$

where

$$\lambda = 2\pi f \Delta t.$$

This shows more clearly the repetitive nature of the poles caused by aliasing of frequencies.

A comparison of the two forms of the basic second-order filter is given in Table 3.1, which also includes expressions for the variance of the output when the input is white noise with one-sided PSD equal to N.

Table 3.1. Comparison of Continuous and Discrete Second-Order Filters

FORM	DIFFERENTIAL — DIFFERENCE EQUATION	MODULUS SQUARED	PEAK	BANDWIDTH	VARIANCE OF OUTPUT WHEN INPUT IS WHITE NOISE WITH ONE-SIDED PSD N.
ANALOG	$\ddot{y}+2\zeta\omega_n\dot{y}+\omega_n^2 y = cx$	$\frac{c^2}{\omega^4-\omega_n^2(2\zeta\omega_n\omega)^2}$	$\omega_p=\omega_n\sqrt{1-\zeta^2}$	$B=2f_p\zeta$ $f_n>>B$	$\frac{N\omega_n}{8\zeta}$
DIGITAL	$y_i=cx_i+h_1y_{i-1}+h_2y_{i-1}$	$\frac{c^2}{4e^{-2\lambda_n\zeta}[\cos\lambda-\cos A]\cdot[\cos\lambda-\cos B]}$ $A=\lambda_n\sqrt{1-\zeta^2}+j\zeta\lambda_n$ $B=\lambda_n\sqrt{1-\zeta^2}-j\zeta\lambda_n$	$\cos\lambda_p=\cos\lambda_n\sqrt{1-\zeta^2}\cdot\cosh\lambda_n\zeta$	$B=2f_p\zeta$	$\frac{N\sinh 2\zeta\lambda_n}{4e^{-2\lambda_n\zeta}\left[1+\left(\frac{\sinh\zeta\lambda_n}{\sin\lambda_n\sqrt{1-\zeta^2}}\right)^2\right]}$

3.5 Higher Order Recursive Filters

The basic recursive filter is, as given in Eq. (3.38),

$$y_i = cx_i + \sum_{k=1}^{M} h_k y_{i-k}. \tag{3.67}$$

The transfer function of this operation is

$$H(f) = \frac{c}{1 - \sum_{k=1}^{M} h_k e^{-j2\pi fk\Delta t}}. \tag{3.68}$$

The denominator of the right-hand side of Eq. (3.68) is a polynomial in powers of the exponential $\exp[-j2\pi f\Delta t]$. To simplify the notation, a single letter is substituted for the exponential. Thus, the expression $\exp[-j2\pi f\Delta t]$ is written as z or $1/z$, depending on the author. Many characteristics of the filter may be obtained by replacing the exponential with z or $1/z$ and examining the resulting polynomial. The study of digital filters in terms of rational expressions in z is called z-transform theory and is discussed in many texts dealing with digital control systems. While this avenue has been fruitful for solving a number of problems, an equivalent derivation of results can be accomplished while retaining the frequency interpretation of the filter. The frequency domain is much more intuitive insofar as the audience of this work is concerned, so it is there that the necessary manipulations will be performed.

Let $P_M(f)$ be defined by

$$P_M(f) = 1 - \sum_{k=1}^{M} h_k e^{-j2\pi fk\Delta t}. \tag{3.69}$$

$P_M(f)$ is an Mth-order polynomial in exp $[-j2\pi f\Delta t]$. Therefore, there are, by the fundamental theorem of algebra, M values of exp $[-j2\pi f'\Delta t]$ such that $P_M(f)$ is equal to zero. In general, these values of exp $[-j2\pi f'\Delta t]$ will be complex, in which case they must appear as pairs of complex conjugate numbers. A prime mark ($'$) has been added to the variable f to denote that it is a complex variable. As $P_M(f)$ is in the denominator of $H(f)$, the values of exp $[-j2\pi f'\Delta t]$ for which $P_M(f)$ is equal to zero are, using standard complex variable terminology, poles. Because the complex exponential function is periodic, there are not simply M poles, but rather an infinite number of them. However, each pair of conjugate poles appears once in the infinite strip in the complex plane where $-1/(2\Delta t) \leqslant \mathrm{Re}(f') \leqslant 1(2\Delta t)$. The poles are repeated in other such strips running parallel to the one defined by the latter inequality. The strip of the complex f' plane for which the inequality holds will be called the principal domain of f'.

As implied above, the variable f' is now a complex variable having real and imaginary parts and can be written in the form

$$(j2\pi\Delta t)f' = (j2\pi\Delta t)f + (\Delta t)\sigma. \tag{3.70}$$

The variables f and σ are always real numbers. The variable f has the usual interpretation of frequency. The variable σ is related to damping. Any pair of poles in the complex f plane will be denoted as Λ_m and Λ_m^*. As usual, the asterisk here means that Λ_m^* is the complex conjugate of Λ_m. For given f_m and σ_m,

$$\begin{aligned} \Lambda_m &= (j2\pi\Delta t)f_m + \Delta t\sigma_m \\ \Lambda_m^* &= -(j2\pi\Delta t)f_m + \Delta t\sigma_m. \end{aligned} \tag{3.71}$$

The quadratic term involving these is

$$P_m(f') = \left[1 - e^{(\Lambda_m - j2\pi\Delta t f')}\right]\left[1 - e^{(\Lambda_m^* - j2\pi\Delta t f')}\right]. \tag{3.72}$$

Thus, when $(j2\pi\Delta t)f'$ equals either Λ_m or Λ_m^*, the quadratic $P_m(f')$ is equal to zero.

The importance of the poles may be seen from the following argument: Except for a constant of proportionality, the poles completely specify the expression

$$P_m(f) = 1 - \sum_{k=1}^{M} h_k e^{-j2\pi fk\Delta t}. \tag{3.73}$$

That is, given the poles, the whole transfer function is uniquely defined (except for a multiplicative constant). Therefore the placement of the poles is crucial to the performance of the filter. Common practice is to define the pole locations in terms of the natural frequency and damping ratio, as was done for the second-order filter in the preceding section. The mth natural frequency will be denoted as λ_m and will be understood to include a $2\pi\Delta t$ factor. Similarly, the damping ratio will be denoted by ζ_m. Thus, in terms of these new variables,

$$\begin{aligned} -\Delta t\sigma_m &= \zeta_m \lambda_m \\ 2\pi\Delta t f_m &= \lambda_m \sqrt{1-\zeta_m^2} \\ \Lambda_m &= j\lambda_m\sqrt{1-\zeta_m^2} - \zeta_m\lambda_m, \quad \lambda_m \geq 0, \ \zeta_m \geq 0. \end{aligned} \tag{3.74}$$

These relationships are shown in Fig. 3.7. The angle θ_m is defined by

$$\zeta_m = \cos\theta_m. \tag{3.75}$$

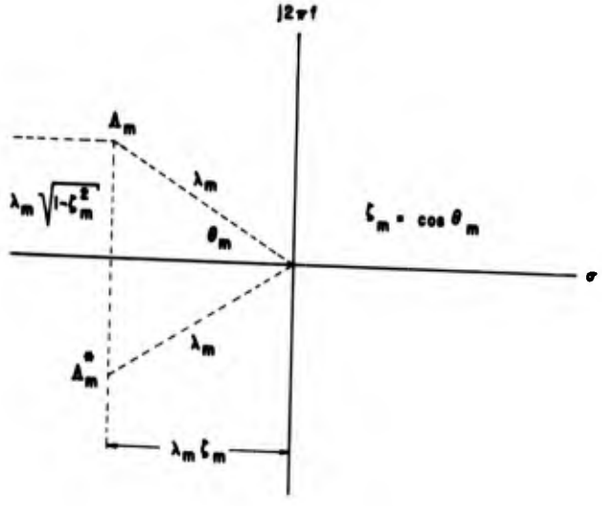


Fig. 3.7. Location of poles Λ_m and Λ_m^*.

The reason for the sign reversal in the definition of ζ_m in Eq. (3.74) is to insure stability. It may be shown that if $\sigma < 0$, then the filter will be stable. Because of the way they were defined, σ will be negative when both λ_m and ζ_m

are positive. Stability is a term used in control theory to indicate that the filter output is bounded and has finite energy when the input is bounded and has finite energy.

Example 3.5 Consider the filter defined by

$$y_i = x_i + a y_{i-1}. \tag{3.76}$$

Assume the following:

$$\begin{aligned} a &> 1 \\ y_{-1} &= 0 \\ x_0 &= 1 \\ x_i &= 0, i > 0. \end{aligned} \tag{3.77}$$

Then the output will have the form

$$y_i = a^i. \tag{3.78}$$

This function is unbounded because, for large enough i, a^i will be bigger than any finite bound chosen. The location of the single pole of this equation may be obtained from

$$1 - a e^{-j\omega \Delta t f'} = 0. \tag{3.79}$$

Solving this for f' yields (considering the value in the principal domain only)

$$-j\omega \Delta t f' = -\ell n a, \tag{3.80}$$

from which it follows, using Eq. (3.70), that

$$\Delta t \sigma = \ell n a. \tag{3.81}$$

Thus, whenever a is greater than one, σ is positive and the output is unbounded. When a is exactly equal to one, then σ is zero and a borderline case occurs: y_i is always equal to unity. While this is bounded, the energy output is infinite, whereas the input had only finite energy.

Returning to the expression for $P_m(f)$ and using the various definitions, the quadratic term can now be written as

$$\begin{aligned} P_m(f) &= \left[1 - e^{(\Lambda_m - j\lambda)}\right]\left[1 - e^{(\Lambda_m^* - j\lambda)}\right] \\ &= \left[1 - e^{-j\lambda} e^{j\lambda_m \sqrt{1 - \zeta_m^2}} - \zeta_m \lambda_m\right] \times \\ &\quad \left[1 - e^{-j\lambda} e^{-j\lambda_m \sqrt{1 - \zeta_m^2}} - \zeta_m \lambda_m\right] \\ &= \left[1 - e^{-j\lambda} e^{-\zeta_m \lambda_m} \left\{e^{-j\lambda_m \sqrt{1 - \zeta_m^2}} + e^{j\lambda_m \sqrt{1 - \zeta_m^2}}\right\}\right. \\ &\quad \left. + e^{-2j\lambda} e^{-2\zeta_m \lambda_m}\right] \end{aligned}$$

$$P_m(f) = \left[1 - \left\{2\cos\left(\lambda_m\sqrt{1-\zeta_m^2}\right) e^{-\zeta_m\lambda_m}\right\} e^{-j\lambda} + \left\{e^{-2\zeta_m\lambda_m}\right\} e^{-2j\lambda}\right]. \tag{3.82}$$

That is, the mth quadratic term in the denominator of the transfer function will always have the form

$$P_m(\lambda) = 1 - b_{1m}\, e^{-j\lambda} - b_{2m}\, e^{-2j\lambda}, \tag{3.83}$$

where the coefficients b_{1m} and b_{2m} are defined by

$$b_{1m} = 2\cos\left(\lambda_m\sqrt{1-\zeta_m^2}\right) e^{-\zeta_m\lambda_m} \tag{3.84}$$

and

$$b_{2m} = -e^{-2\zeta_m\lambda_m}. \tag{3.85}$$

It is possible to have Λ_m as a real number as indicated in the previous example, in which case the mth factor rather than being quadratic is simply first order as shown in Eq. (3.86):

$$P_m(\lambda) = \left(1 - e^{-k_m}\, e^{-j\lambda}\right) = \left(1 - b_{0m}\, e^{-j\lambda}\right), \tag{3.86}$$

and

$$b_{0m} = e^{-\Lambda_m}. \tag{3.87}$$

In this case there is only one coefficient, b_{0m}. The denominator of the transfer function therefore consists of a product of linear and quadratic terms as described above. It follows that

$$P_M(f) = \left(1 - b_{01}e^{-j\lambda}\right) \ldots \left(1 - b_{0p}e^{-j\lambda}\right) \cdot \left(1 - b_{11}e^{-j\lambda} - b_{21}e^{-2j\lambda}\right) \ldots \left(1 - b_{1q}e^{-j\lambda} - b_{2q}e^{-2j\lambda}\right), \tag{3.88}$$

where

$$M = p + 2q.$$

The importance of such expressions arises from the fact that when the multiplication is carried out, the b coefficients can be combined to obtain the h_k coefficients, which are the desired weights of the digital filter. Some examples will illustrate the point. For $M = 1$, $P_1(f)$ is given by

$$P_1(f) = 1 - b_{01}e^{-j\lambda} = 1 - h_1 e^{-2j\lambda}, \tag{3.89}$$

where

$$h_1 = b_{01}.$$

Thus for P_1, h_1 and the b coefficient are identical.

When there are two recursive terms, there are two possibilities: Either the poles are complex conjugates or they are unrelated and real. If they are complex conjugates, P_2 is given by

$$P_2(f) = 1 - b_{11}e^{-j\lambda} - b_{21}e^{-2j\lambda},$$

where

$$h_1 = b_{11}$$

and

$$h_2 = b_{21}. \tag{3.90}$$

This case is discussed in detail in the immediately preceding section. On the other hand, if both the poles are real, P_2 will be given by

$$P_2(f) = (1 - b_{01}e^{-j\lambda})(1 - b_{02}e^{-j\lambda}),$$

where

$$h_1 = b_{01} + b_{02}$$

and

$$h_2 = b_{01}b_{02}. \tag{3.91}$$

There are also two possibilities for P_3. The first of these is one real root and two complex roots, in which case P_3 will have the following form:

$$P_3(f) = (1 - b_{01}e^{-j\lambda})(1 - b_{11}e^{-j\lambda} - b_{21}e^{-2j\lambda})$$

$$= 1 - (b_{01} + b_{11})e^{-j\lambda} - (b_{21} - b_{11}b_{01})e^{-2j\lambda} - (b_{01}b_{21})e^{-3j\lambda}, \tag{3.92}$$

where

$$h_1 = b_{01} + b_{11},$$

$$h_2 = b_{21} - b_{01}b_{11},$$

and

$$h_3 = -b_{01}b_{21}.$$

If there are three real poles, then P_3 will have the form

$$P_3(f) = (1 - b_{01}e^{-j\lambda})(1 - b_{02}e^{-j\lambda})(1 - b_{03}e^{-j\lambda})$$

$$= 1 - (b_{01} + b_{02} + b_{03})e^{-j\lambda} - (-b_{01}b_{02} - b_{01}b_{03} - b_{02}b_{03})e^{-2j\lambda}$$

$$- b_{01}b_{02}b_{03}e^{-3j\lambda}, \tag{3.93}$$

where

$$h_1 = b_{01} + b_{02} + b_{03},$$

$$h_2 = -(b_{01}b_{02} + b_{01}b_{03} + b_{02}b_{03}),$$

and

$$h_3 = b_{01}b_{02}b_{03}.$$

The number of forms that P_M can take increases with the size of m. Thus, P_4 will have (a) four real roots, (b) two real roots and two imaginary roots, or (c) two pairs of complex conjugate roots.

An algorithm for computing the $\{h_k\}$, given the poles, is due to Daniel Brown [13]. Assuming that all of the poles may be grouped into conjugate pairs and that there are $m/2$ such pairs, then the $\{h_k\}$ may be computed using the expression

$$h_k^{(\ell)} = h_k^{(\ell-1)} - b_{\ell 1}h_{k-1}^{(\ell-1)} - b_{\ell 2}h_{k-2}^{(\ell-2)},$$

where

$$h_o^{(m/2)} = -1$$

$$h_k^{(o)} = 0, \; k \neq 0. \tag{3.94}$$

The terms $h_k^{(\ell)}$ are intermediate for $\ell < m/2$.

3.6 High-Pass, Low-Pass, and Bandpass Filters

The basic design for an electrical low-pass filter is often patterned after that developed by Butterworth [14]. The modulus squared of the frequency response function of the Butterworth filter takes the form

$$|H(f)|^2 = \frac{1}{1 + \left(\frac{f}{f_c}\right)^K}. \tag{3.95}$$

$H(f)$ is the frequency response function of the filter, and f_c is the half-power cutoff frequency, i.e., that frequency for which the amplitude squared of $H(f)$ is reduced by half from its value at $f = 0$. The even integer K determines the sharpness of the slope of the squared gain of the filter; as K increases, so does the filter performance.

A digital filter can be realized, as proven by Holtz and Leondes [15], that has the same form as Eq. (3.95). Otnes [16] demonstrated how the filters could be synthesized. As will be shown below, a set of numerical filter weights $\{h_k\}$ and c can be chosen such that

$$|H(f)|^2 = \frac{1}{1 + \left(\dfrac{\sin(\pi\Delta t f)}{\sin(\pi\Delta t f_c)}\right)^{2M}}, \quad 0 \leq f \leq \frac{1}{2\Delta t}. \tag{3.96}$$

This function is equal to unity for $f = 0$. For $f = f_c$, the cutoff frequency $|H(f)|^2$ is equal to one-half. For $f = 1/(2\Delta t)$, the folding (Nyquist) frequency, $|H(f)|^2 \to 0$ for large M. Thus, over the domain $[0, 1/(2\Delta t)]$, the principal domain in the digital case, this filter acts in a manner similar to the Butterworth filter.

The method of realizing this filter is much the same as that used by Butterworth; namely, through proper mapping of poles on the unit circle in the complex plane, as described in the preceding sections. As discussed previously, the definition of f must be extended to make f complex, so that $H(f)$ becomes a complex function of a complex variable and is defined over the entire complex plane rather than simply along the frequency axis. Therefore, suppose that

$$(\pi\Delta t)f = \alpha + j\beta. \tag{3.97}$$

Here α and β are both real variables. When $\beta = 0$, then α corresponds to f in the previous sense.

Examining Eq. (3.96), the denominator is zero when

$$\left(\frac{\sin(\pi\Delta t f)}{\sin(\pi\Delta t f_c)}\right)^{2M} + 1 = 0. \tag{3.98}$$

The poles of $H(f)$ are roots of unity. However, they are not the Kth or even $2K$th roots of unity. Rather, they turn out to be a subset of the set of the $4K$th roots of unity.

The poles of the digital counterpart of the Butterworth filters are somewhat more complex. The expression for $H(f)$ is periodic, as may be seen from Eq. (3.96). As usual, equivalent results can be obtained when the analysis is restricted to the range

$$-\frac{1}{2\Delta t} < \text{Re } f < \frac{1}{2\Delta t}. \tag{3.99}$$

All of the poles within the infinite strip defined by Eq. (3.99) will be repeated in strips running parallel to it. Suppose then that a filter with $2M$ poles is desired. These poles will have the form

$$\sin(\pi f_i \Delta t) = \sin(\pi f_c \Delta t)\left\{\sin\left[\pi\left(\frac{1}{2}+\frac{1}{M}\left(i-\frac{1}{2}\right)\right)\right] - j\cos\left[\pi\left(\frac{1}{2}+\frac{1}{M}\left(i-\frac{1}{2}\right)\right)\right]\right\}$$

$$\equiv a_i + jb_i,\ i = 1, \ldots, M. \tag{3.100}$$

The set $\{a_i + jb_i\}$ plus its complex conjugate form a family of $2M$ poles in the left-hand part of the (a, b) plane.

The next step is to define α_i and β_i such that

$$\sin(\alpha_i + j\beta_i) = a_i + jb_i. \tag{3.101}$$

The inversion of Eq. (3.101) is the complex arcsine function

$$\alpha_i + j\beta_i = \sin^{-1}(a_i + jb_i) = -j\ell n\left\{j(a_i + jb_i) + \sqrt{1-(a_i+jb_i)^2}\right\}. \tag{3.102}$$

An alternate form of the same conformal mapping may be used. For the quadrant of interest, that is, for $i = 1, \ldots, M$, the desired parameters may be found to be

$$\alpha_i = \arctan\left[\sqrt{\frac{D_i}{b_i^2}} - 1\right] \operatorname{sgn}(a_i),$$

$$\beta_i = \ell n(E_i), \tag{3.103}$$

where

$$D_i = \frac{1}{2}\left[-C_i + \sqrt{C_i^2 + 4b_i^2}\right],$$

$$E_i = \sqrt{D_i + 1} + \sqrt{D_i},$$

and

$$C_i = 1 - (a_i^2 + b_i^2),\ i = i, \ldots, M. \tag{3.104}$$

The next step is a transformation of α_i and β_i to polar coordinates;

$$\lambda_i = 2\sqrt{\alpha_i^2 + \beta_i^2}$$

$$\zeta_i = \cos\left[-\arctan\left(\frac{\alpha_i}{\beta_i}\right)\right]. \tag{3.105}$$

An alternative definition for λ_i is

$$\lambda_i = |2\pi\Delta t f_i| = 2\ [(\pi\Delta t)\ |f_i|]\ . \tag{3.106}$$

This reduces to the expression in Eq. (3.97). Some formulations would include the factor of 2 in the α and β terms. Care must be exercised to insure that there is no confusion on this point.

Define the ith quadratic in the dummy variable Z by

$$q_i(Z) = 1 - 2Z\cos\left(\lambda_i\sqrt{1-\zeta_i^2}\right)\exp(-\zeta_i\lambda_i) + Z^2\exp(-2\zeta_i\lambda_i). \tag{3.107}$$

The various powers of Z that result when the quadratic terms are multiplied out will serve as an index for locating the filter weights. Explicitly, the product of the M quadratic polynomials is then

$$B(Z) = \prod_{i=1}^{M} q_i(Z)$$

$$= 1 - \sum_{k=1}^{2M} h_k Z^k. \tag{3.108}$$

The set of weights $\{h_k\}$ defined by Eq. (3.108) is the desired set of recursive weights. The single nonrecursive weight c is obtained using the formula

$$c = 1 - \sum_{k=1}^{2M} h_k. \tag{3.109}$$

Summarizing the above steps, a computer routine written to generate such weights would

1. Compute the M poles in the plane for which $\alpha > 0$, $\beta < 0$. These poles lie on a quarter-circle.
2. Do the conformal mapping defined by Eqs. (3.102) or (3.103) and (3.104). The latter may be used on computers which do not have complex arithmetic software or when double precision is required.
3. Combine each such term with its complex conjugate and form M quadratic coefficient triplets.
4. Find the recursive weights by combining the quadratic coefficients.
5. Finally, define the nonrecursive weight in a manner such that $|H(0)|^2 = 1$, thus normalizing the filter.

Some problems will be encountered when this procedure is implemented. The nature of the difficulty can be seen more easily through the use of an

unstable filter of the same type, that is, one having all the roots of unity of a given degree. This formulation was derived by Otnes [16]. It is unstable, but the expressions for the weights will help gain insight into the nature of certain difficulties:

$$y_i = (-1)^M \left(\frac{2}{c}\right)^N x_{i-M} + \left[\binom{N}{1} y_{i-1} - \binom{N}{2} y_{i-2} + \ldots - (-1)^M \left[\binom{N}{M} + \left(\frac{2}{c}\right)^N\right] y_{i-M} \ldots - \binom{N}{N-1} y_{i-N-1} + \binom{N}{N} y_{i-N}\right], \tag{3.110}$$

where $c = csc\ (\pi \Delta t f_c)$, $M = N/2$, and N must be even. Tedious manipulation will show that the frequency response function has the form

$$H(f) = \frac{1}{1 + \left(\dfrac{\sin(\pi f \Delta t)}{\sin(\pi f_c \Delta t)}\right)^{2M}}. \tag{3.111}$$

Note that this $H(f)$ is completely real for real f and that its poles are equally spaced around the circle with center at the axis of the $(\alpha - \beta)$ plane. The coefficient of the $(i - M)$th term is

$$h_M = (-1)^M \left[\binom{N}{M} + \left(\frac{2}{c}\right)^N\right] \approx (-1)^M (2^N) \left[\frac{1}{\sqrt{\pi N}} + \sin^N(\pi f_c \Delta t)\right]. \tag{3.112}$$

The $\sin^N(\pi f_c \Delta t)$ portion of the expression may become small with respect to the $1/\sqrt{\pi N}$ term for certain values of N and f_c. Depending on computer word size, the sine part of the expression will actually be lost out of the lower end of the computer registers for large enough N. When this underflow occurs, the filter shape deviates drastically from that of Eq. (3.112). The degradation of the filter varies with the amount of underflow, and the filter shape may deteriorate completely when the sine term is lost. This problem carries over to the stable case, which uses only the poles in the $\beta < 0$ half-plane.

High-pass filters can be generated in a like manner. The desired transfer function in this case has the form

$$|H(f)|^2 = \frac{1}{1 + \left(\dfrac{\cos(\pi f \Delta t)}{\cos(\pi f_c \Delta t)}\right)^{2M}}$$

$$= \frac{1}{1 + \left(\dfrac{\sin\left(\pi f \Delta t + \dfrac{\pi}{2}\right)}{\cos(\pi f_c \Delta t)}\right)^{2M}}. \tag{3.113}$$

Thus, in Eq. (3.100) the $\sin(\pi f_c \Delta t)$ term is replaced with $\cos(\pi f_c \Delta t)$, and $\pi/2$ is subtracted from the right-hand side of the a_l term of Eq. (3.103). Alternatively, the high-pass set of weights $\{h_k\}$ for the cutoff frequency f_c may be obtained by computing the low-pass set of weights for frequency $(1/(2\Delta t) - f_c)$ and then changing the signs for the odd-numbered weights from plus to minus. The single nonrecursive weight for the high-pass case is

$$c = 1 - \sum_{k=1}^{2M} (-1)^k h_k. \tag{3.114}$$

Bandpass filters may also be obtained via this same conformal mapping. One expression for a bandpass filter which works well is

$$|H(f)|^2 = \frac{1}{1 + \left(\dfrac{\cos(2\pi \Delta t f_0) - \cos(2\pi \Delta t f)}{A}\right)^{2M}}. \tag{3.115}$$

The center of the bandpass is near, but not exactly at, f_0. To see the action of this, consider the function $g(f)$ defined by

$$g(f) = [\cos(2\pi \Delta t f_0) - \cos(2\pi \Delta t f)]. \tag{3.116}$$

The function g is equal to zero for $f = \pm f_0$ and is symmetric about $f = 0$. This property of symmetry is desirable for the filter form of Eq. (3.115) to be a possible candidate for a bandpass filter. Another requirement is that

$$|H(f_c \pm B)|^2 = \frac{1}{2}. \tag{3.117}$$

Here f_c is the center frequency, and $2B$ is the width of the filter between half-power points. The requirement of having the half-power points at frequencies $(f_c - B)$ and $(f_c + B)$ can be met by properly defining f_0 and A. Appropriate expressions are

$$\cos(2\pi\Delta t f_0) = \cos(2\pi\Delta t f_c)\cos(2\pi\Delta t B)$$

$$A = -\sin(2\pi\Delta t f_c)\sin(2\pi\Delta t B). \tag{3.118}$$

As usual, it is necessary that the poles of H lie in the left half of the complex plane. That is, expressions for f_i must be solutions of

$$\left[\frac{\cos(2\pi\Delta t f_0) - \cos(2\pi\Delta t f)}{A}\right]^{2M} = -1. \tag{3.119}$$

The solution for an Mth-order realizable filter is

$$\cos(2\pi\Delta t f_i) = \cos\left\{(2\pi\Delta t f_c)\cos(2\pi\Delta t B) + \sin(2\pi\Delta t f_c)\sin(2\pi\Delta t B)\right.$$
$$\left.\cdot \sin\left[\pi\left(\frac{1}{2} + \frac{1}{M}\left(i - \frac{1}{2}\right)\right)\right]\right\} - j\left\{\sin(2\pi\Delta t f_c)\sin(2\pi\Delta t B)\right.$$
$$\left.\cdot \cos\left[\pi\left(\frac{1}{2} + \frac{1}{M}\left(i - \frac{1}{2}\right)\right)\right]\right\}, \quad i = 1, \ldots, M. \tag{3.120}$$

This could be written as

$$\cos(a_i + j\beta) = a_i + jb_i, \tag{3.121}$$

where

$$a_i = \left\{\cos(2\pi\Delta t f_c)\cos(2\pi\Delta t B) + \sin(2\pi\Delta t f_c)\sin(2\pi\Delta t B)\right.$$
$$\left.\cdot \sin\left[\pi\left(\frac{1}{2} + \frac{1}{M}\left(i - \frac{1}{2}\right)\right)\right]\right\} \tag{3.122a}$$

and

$$b_i = \left\{\sin(2\pi\Delta t f_c)\sin(2\pi\Delta t B)\right.$$
$$\left.\cdot \cos\left[\pi\left(\frac{1}{2} + \frac{1}{M}\left(i = \frac{1}{2}\right)\right)\right]\right\}, \quad i = 1, \ldots, M. \tag{3.122b}$$

Making use of the fact that $\cos 0 = \sin(\pi/2 - 0)$, we can express Eq. (3.121) as

$$(2\pi\Delta t) f_i \equiv a_i + j\beta_i$$

$$\equiv \frac{\pi}{2} - \arcsin[a_i + jb_i], \quad i = 1, \ldots, M. \tag{3.123}$$

The mapping is rewritten into this form because the previous mapping for the low-pass filter, with a few modifications, may now be used for this case. These are

$$a_i = \frac{\pi}{2} - \left(\arctan \sqrt{\frac{D_i}{b_i^2} - 1} \right) \operatorname{sgn}(a_i),$$

$$\beta_i = -\ell n(E_i),$$

$$\lambda_i = \sqrt{a_i^2 + \beta_i^2},$$

and

$$C = \sqrt{\left(1 - \sum_{k=1}^{2M} h_k \cos(2\pi \Delta t f_0)\right)^2 + \left(\sum_{k=1}^{2M} h_k \sin(2\pi \Delta t f_0)\right)^2}. \tag{3.124}$$

Thus, the basic arcsine transformation may be used to compute the weights for all three cases. The arccosine or arctangent mappings would probably work equally well. The fact that only one mapping need be used is convenient in computer implementations.

3.7 Miscellaneous Topics in Filtering

This section will discuss miscellaneous results of interest. The topics to be covered are polynomial filters, filter synthesis, and some special filters and filtering techniques.

Polynomial filters have been discussed by numerous authors. A polynomial filter is one that will pass a polynomial of a certain degree without alteration while smoothing whatever noise may have been added to the polynomial. Such filters would be useful when processing trajectory data. Ormsby [12] indicated and Brillinger [17] later proved that necessary and sufficient conditions for a filter to pass an Nth-order polynomial without changes are

$$1.\ H(0) = 1$$

$$2.\ \frac{d^i}{df^i} H(f)\bigg|_{f=0} = 0,\ i = 1, \ldots, N. \tag{3.125}$$

There are many filters that meet these conditions, including the binomial filter discussed in Example 3.2. Unfortunately, many of the smoothing filters had as their only virtue the ability to pass polynomials. One such filter is the following.

Example 3.6 The moving average filter simply averages over $(2M + 1)$ points;

$$y_i = \frac{1}{(2M+1)} \sum_{k=-M}^{M} x_{i+k}. \tag{3.126}$$

The transfer function of this may be found to be

$$H(f) = \frac{\sin\left[(2M+1)\pi\Delta tf\right]}{(2M+1)\sin(\pi\Delta tf)}. \tag{3.127}$$

This function, which is the digital equivalent of $(\sin x/x)$, is certainly not impressive as a smoothing filter. Its action is irregular and apt to fluctuate. Yet it is widely used, especially by the economists. It will, however, pass a linear term or a constant without alteration.

One constraint which Eq. (3.125) places upon the filter is that the weighting function must be symmetric. This is because the derivatives must be zero, which tends to necessitate the imaginary part of the transfer function being zero.

Nonsymmetric filters, such as the recursive ones described in the preceding section, in general do have a nonzero imaginary part and, thus, have a nonzero phase function. If that is a problem, it may be overcome easily, provided that the data are available in a reasonable form. The procedure consists of using a low-pass filter with half as many poles as would be required normally. The data are then filtered in the reverse direction with the same filter. The resulting transfer function would be of the form

$$H(f) = \frac{e^{-j\Phi(f)}}{\sqrt{1+\left(\dfrac{\sin(\pi\Delta tf)}{\sin(\pi\Delta tf_a)}\right)^{2R}}} \quad \frac{e^{j\Phi(f)}}{\sqrt{1+\left(\dfrac{\sin(\pi\Delta tf)}{\sin(\pi\Delta tf_a)}\right)^{2R}}}$$

$$= \frac{1}{1+\left(\dfrac{\sin(\pi\Delta tf)}{\sin(\pi\Delta tf_a)}\right)^{2R}} \tag{3.128}$$

The phase functions cancel, leaving $H(f)$ completely real. Unfortunately, the half-power point is no longer at f_a. In order to put it at the cutoff frequency f_c, it turns out that f_a must be given by

$$\sin(\pi\Delta tf_a) = \frac{\sin(\pi\Delta tf_c)}{(\sqrt{2}-1)^{1/(2R)}}. \tag{3.129}$$

That is, the filter used twice must have an apparent cutoff frequency f_a defined in Eq. (3.129) in order to attain the actual cutoff frequency f_c. The result of this is that the filtering action is not quite as sharp as when a single filter with

twice the poles and a nonzero phase function is employed. This may be seen from the following:

$$\left|\left\{\frac{\text{(Transfer function of a simple } 4R\text{-pole low-pass filter)}}{\text{(Transfer function of the double } 2R\text{-pole low-pass filters)}}\right\}\right|^2$$

$$= \frac{1 + 2\lambda\left(\frac{\sin \pi\Delta tf}{\sin \pi\Delta tf_c}\right)^{2R} + \lambda\left(\frac{\sin \pi\Delta tf}{\sin \pi\Delta tf_c}\right)^{4R}}{1 + \left(\frac{\sin \pi\Delta tf}{\sin \pi\Delta tf_c}\right)^{4R}}, \tag{3.130}$$

where $\lambda = \sqrt{2} - 1$. Examination of Eq. (3.130) will show that the simple $4R$-pole filter will have a modulus which is larger than the double $2R$ filter when $0 \leqslant f < f_c$ and smaller, when $f_c < f \leqslant 1/(2\Delta t)$. Equality holds at $f = f_c$. The $4R$-pole filter therefore more closely approximates the performance of the ideal low-pass filter.

One important use of low-pass filters is in decimation. Used in this sense, *decimation* refers to the process of filtering the data and then throwing away data points. For example, suppose that, as usual, $\{x_i\}$ represents data points spaced with an interval Δt. The folding frequency is at $1/(2\Delta t)$ Hz. An rth-order decimation would consist of keeping only every rth point. That is, keeping a single point, discarding $(r - 1)$ points, keeping a point, discarding $(r - 1)$ points, etc. The new sampling rate $\Delta t'$ would therefore be

$$\Delta t' = r\Delta t. \tag{3.131}$$

The new folding frequency therefore is

$$f_N' = \frac{1}{2\Delta t'} = \frac{1}{r}\left(\frac{1}{2\Delta t}\right). \tag{3.132}$$

If the discarding is done without filtering, all the information above $1/(2r\Delta t)$ Hz will be aliased (folded) back into the interval $[0, 1/(2r\Delta t)]$. In order to avoid this, the data must be filtered to eliminate the information on the interval $[1/(2r\Delta t), 1/(2\Delta t)]$. This may be accomplished through the use of a low-pass filter with its cutoff frequency at $1/(2r\Delta t)$ Hz.

Example 3.7 Suppose that a certain launch vehicle has a transducer sampled by the ADC at a rate of 400 samples per second. The launch vehicle is active for a period of five minutes. It is desired to plot a time history of the transducer on a piece of graph paper 30 inches long.

The flight (at least the part to be plotted) is 300 seconds in duration, or 10 seconds per inch of plot. This would result in 4000 data points per inch on the final plot. The plot would look "hashy," it would take a considerable amount

of computer time to generate the plot tape, and a correspondingly large amount of time for the plotting device to turn the tape into the desired graph.

On the other hand, if a 100th-order decimation is performed, then only 40 points per inch (or a total of 1200 points) will be plotted. For a small computation cost for the filtering, the cost of plotting may be cut by a factor of 100. In addition, for the purpose of getting an overall idea of what the data function is doing during the flight, the filtered plot may very well be more intelligible than the unsmoothed one. In this example,

$$\Delta t = \frac{1}{400} = 0.0025,$$

$$r = 100,$$

$$\Delta t' = 0.25,$$

$$f_N = 200\,Hz,$$

and

$$f_N' = 2\,Hz. \tag{3.133}$$

The cutoff frequency for the low-pass filter is 2 Hz. Either an Ormsby filter as described in Example 3.4 or a recursive filter as described in Section 3.6 is suitable, the latter being much cheaper to use.

CHAPTER 4
FOURIER SERIES AND FOURIER TRANSFORM COMPUTATIONS

4.1 Standard Fourier Transform and Fourier Series Evaluation

Fourier series and, more generally, Fourier transforms arise repeatedly in the analysis of vibration time histories and other time series. In addition to simplifying theoretical analyses, certain data analysis questions are answered via Fourier transforms. In the case of digitized data, the finite, discrete Fourier transform is most important. The classical Fourier series is, for all practical purposes, computationally identical to the Fourier transform although they differ theoretically. Thus, attention may be directed toward the more general Fourier transform.

The continuous, infinite-range transform as discussed in Section 1.4 is

$$X(f) = \int_{-\infty}^{\infty} x(t)\, e^{-j2\pi ft}\, dt$$

$$= \int_{-\infty}^{\infty} x(t) \cos 2\pi ft\, dt - j \int_{-\infty}^{\infty} x(t) \sin 2\pi ft\, dt, \quad -\infty < f < \infty. \tag{4.1}$$

Attention shall be subsequently concentrated on the finite transform computed from discrete data:

$$Z(f) = \Delta t \sum_{i=0}^{N-1} z(i)\, e^{-j2\pi fi\Delta t}, \quad -\infty < f < \infty. \tag{4.2}$$

This differs from Eq. (1.75) in that the complex variable z rather than the real variable x is used. The sequence $\{z(i),\ i = 0, \ldots, N-1\}$ is allowed to be a sequence of complex variables. For a real sequence $\{x_i,\ i = 0, \ldots, N-1\}$, the cosine and sine transforms will usually be evaluated separately:

$$C_x(f) = \Delta t \sum_{i=0}^{N-1} x_i \cos 2\pi fi\Delta t, \tag{4.3}$$

$$Q_x(f) = \Delta t \sum_{i=0}^{N-1} x_i \sin 2\pi fi\Delta t. \tag{4.4}$$

If no attention whatsoever is paid to computational aspects, Eq. (4.2) is evaluated directly. That is, a frequency value f is selected, and the summations are evaluated with the necessary sines and cosines being obtained. The basic computational loop is

1. Evaluate $i \cdot \Delta t \cdot f \cdot 2\pi = \theta$,
2. Compute $\cos\theta$, $\sin\theta$,
3. Compute $x_i \cos\theta, x_i \sin\theta$,
4. Accumulate both sums,
5. Test for $(N-1)$th data point,
6. Increment i, return to (1).

This basic loop would require about 542 μsec on an IBM 7090. Thus, for $N = 10^3 = 1000$ data points, each value of the transform requires 310,000 μsec = 310 msec. For $N = 10^4 = 10{,}000$ data points, the time is 3,100,000 μsec = 3.1 sec. If all possible 5000 frequency points were evaluated, the computing time would be over four hours. The usual selection of frequency points at which to evaluate the Fourier transform or series is

$$f_k = k\Delta f = \frac{k}{T} = \frac{k}{N\Delta t}, \quad k = 0 \ldots, N/2. \tag{4.5}$$

The cosine transform then becomes

$$X_k = X(k\Delta f) = \sum_{i=0}^{N-1} x_i \cos 2\pi \frac{ik}{N}, \quad k = 0, 1, \ldots, N/2. \tag{4.6}$$

Now, sines and cosines can be evaluated recursively:

$$\begin{bmatrix} \cos\frac{2\pi k}{N}(i+1) \\ \sin\frac{2\pi k}{N}(i+1) \end{bmatrix} = \begin{bmatrix} \cos\frac{2\pi k}{N} & -\sin\frac{2\pi k}{N} \\ \sin\frac{2\pi k}{N} & -\cos\frac{2\pi k}{N} \end{bmatrix} \begin{bmatrix} \cos\frac{2\pi k}{N} i \\ \sin\frac{2\pi k}{N} i \end{bmatrix},$$

$$i = 0, 1, \ldots, N/2. \tag{4.7}$$

The computing procedure then becomes

1. Compute $\cos[2\pi(i+1)/N]$ and $\sin[2\pi(i+1)/N]$ recursively.
2. Compute $x_i \cos\theta, x_i \sin\theta$,
3. Accumulate both sums.
4. Test for the $(N-1)$th data point,
5. Return to (1).
6. Repeat Steps 1 through 5 $N/2$ times.

In a carefully constructed machine language program, the above takes about 71 machine cycles, or about 142 μsec on the IBM 7090. If 1000 data points

are involved, then 14,200 μsec = 14.2 msec are necessary for each frequency point. If 500 frequency points are evaluated, the total time is $10^3 \times 7{,}100$ μsec = 7.1 sec. If 10,000 points are involved, then the time is 142,000 μsec = 142 msec for each point and 710 sec for the total job. Hence, about 12 minutes is required in this nominal case for just the basic computations. Time requirements for data input, printing, and plotting of results are additional. The computational time grows as N^2, and the time requirements can quickly become excessive. Record lengths of 5 to 10,000 points are not unusual in vibration data analysis, hence the motivation in this field and others for efficient computational techniques.

4.2 Fast Fourier Transforms

Recently, algorithms that reduce Fourier transform computational times by a startling degree have come to light. In 1965, the first widely noticed publication of a high-speed algorithm occurred [18]. The crux of the procedure has been traced back to at least 1928. The approach followed by Gentleman and Sande [19] and Bergland [20] will be used for expository purposes here.

The algorithms are most simply designed for complex sequences. Thus the formula analyzed is the following:

$$Z_k = \sum_{i=0}^{N-1} z_i \exp\left[-j\frac{2\pi ik}{N}\right], \qquad k = 0, 1, \ldots, N-1. \tag{4.8}$$

The factor Δt is omitted to simplify notation. It serves as a scale factor and only becomes important when the plotting and printing of results are involved. Some further simplification in notation is desirable. The following notation for the complex exponential is introduced:

$$W = \exp\left[-j\frac{2\pi}{N}\right]. \tag{4.9}$$

Then the complex exponentials in Eq. (4.9) become

$$W^{ik} = \exp\left[-j\frac{2\pi ik}{N}\right]. \tag{4.10}$$

Equation (4.8) is then rewritten to become

$$Z_k = \sum_{i=0}^{N-1} z_i W^{ik}, \qquad k = 0, 1, \ldots, N-1. \tag{4.11}$$

The fast Fourier transform (FFT) algorithms may be developed when the number of data points is a nonprime (composite) integer. Consider the simplest case $N = A \times B$. Equation (4.11) can be rewritten as

$$Z_{(c+dA)} = \sum_{b=0}^{B-1} \sum_{a=0}^{A-1} z_{(b+aB)}\, W^{(b+aB)(c+dA)}, \tag{4.12}$$

where

$$\begin{aligned} i &= (b + aB) \qquad \text{(time index)} \\ k &= (c + dA) \qquad \text{(frequency index)}, \end{aligned}$$

$$a, c = 0, 1, \ldots, A - 1; b, d = 0, 1, \ldots, B - 1.$$

The exponent of W in Eq. (4.12) can be expanded to give

$$\begin{aligned} W^{(b+aB)(c+dA)} &= W^{bc}\, W^{bdA}\, W^{acB}\, W^{adAb} \\ &= W^{bc}\, W^{bdA}\, W^{acB}. \end{aligned} \tag{4.13}$$

This is true since a and d are integers and the complex exponential W raised to any integral power is unity. A directly related fact used later is that

$$\exp\left[-j\,\frac{2\pi}{2}\right] = -1. \tag{4.14}$$

Equation (4.12) can now be rearranged since portions of the complex exponential can be factored out from the innermost sum. Thus,

$$\begin{aligned} Z_{(c+dA)} &= \sum_{b=0}^{B-1} \sum_{a=0}^{A-1} z_{(b+aB)}\, W^{bc}\, W^{bdA}\, W^{acB}, \\ &= \sum_{b=0}^{B-1} W^{bdA} \sum_{a=0}^{A-1} z_{(b+aB)}\, W^{acB}\, W^{bc}, \end{aligned}$$

$$c = 0, 1, \ldots, A - 1; d = 0, 1, \ldots, B - 1. \tag{4.15}$$

Now, if it is assumed that the complex exponentials have been computed in advance, then the number of complex multiply-add operations necessary to evaluate the double summation of Eq. (4.15) is $AB(A + B)$ rather than $(AB)(AB)$, as it would be if Eq. (4.11) were evaluated directly.

To evaluate this number of operations, first note that

$$\begin{aligned} W^{bdA} &= \exp\left[-j\,\frac{2\pi}{B}\,bd\right], \qquad b, d = 0, 1, \ldots, B - 1 \\ W^{acB} &= \exp\left[-j\,\frac{2\pi}{A}\,ac\right], \qquad a, c = 0, 1, \ldots, A - 1. \end{aligned} \tag{4.16}$$

These are precisely the exponentials needed for a B-point and an A-point discrete transform, respectively. Equation (4.15) can be recognized as a sequence of two Fourier transforms applied to data sequences of length A and B respectively. Redefine the original data as

$$u_b(a) = z_{(b+aB)}, \qquad \begin{matrix} a = 0, 1, \ldots, A-1 \\ b = 0, 1, \ldots, B-1. \end{matrix} \tag{4.17}$$

In terms of a digital computer core storage, the data have been relabeled as B sequences, each of length A. Figure 4.1 illustrates this for $A = 2, B = 5, N = 10$.

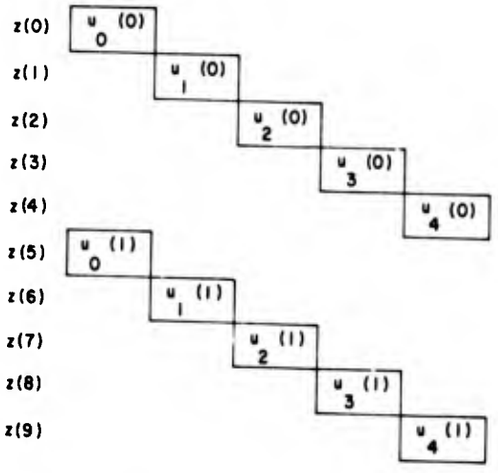


Fig. 4.1. Storage arrangement for ten points.

Now, the B Fourier transforms of the $u_b(a)$ (each of length A) are computed

$$U_b(c) = \sum_{a=0}^{A-1} u_b(a)\, W^{acB}, \qquad b = 0, 1, \ldots, B-1.$$

For the example $A = 2$, $B = 5$, the results are stored as shown in Fig. 4.2. Now each value $U_b(c)$ must be multiplied by the quantity W^{bc}, termed a "twiddle" factor by Gentleman and Sande [19]. Define the new quantities

$$v_c(b) = U_b(c)\, W^{bc}. \tag{4.19}$$

Now, A Fourier transforms of length B are computed, namely

$$V_c(d) = \sum_{b=0}^{B-1} v_c(b)\, W^{bdA}, \qquad d = 0, 1, \ldots, B-1, \tag{4.20}$$

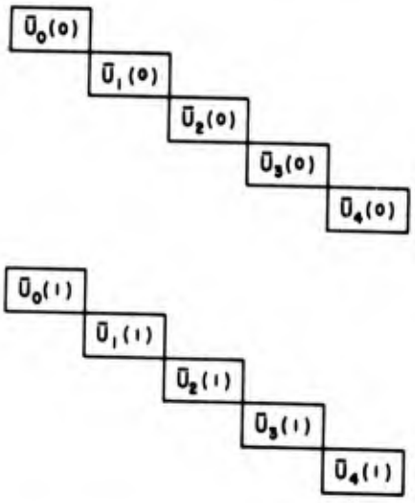


Fig. 4.2. Storage after the first operation.

where

$$Z_{(c+dA)} = V_c(d).$$

Hence, in two stages, the Fourier transform of the entire sequence has been computed. The final values are stored as shown in Fig. 4.3. It should be noted that the arrangement of the Fourier transformed values is not the same as that of the original data. This fact requires that the results be shuffled in core storage to obtain the meaningful arrangement. This will be illustrated in more detail in the discussion of $N = 2^p$. The number of multiply-add operations is now noted to be

$$B \cdot A \cdot A$$

operations for the B transforms of length A plus

$$A \cdot B \cdot B$$

for the A transforms of length B. The total number of multiply-add operations then is

$$B \cdot A \cdot A + A \cdot B \cdot B = AB(A + B).$$

In practice, programming the algorithm for N as a power of two has special advantages. In this case, the intermediate transforms are two elements, and the exponentials have values of +1 or −1. This allows complex multiplication operations to be avoided and saves additional time. For $N = 4^p$, similar savings can be accomplished since the values of the four exponentials are $+1, +j, -1, -j$.

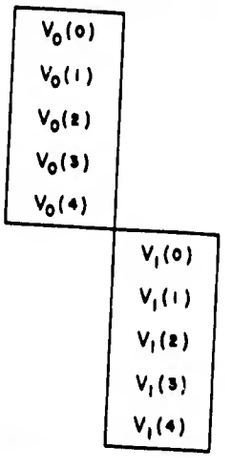


Fig. 4.3. Final storage arrangement.

If real and imaginary parts are worked with individually, complex multiplication operations can still be avoided to a large extent. The power of two is the smallest prime factor, however, and sequences can usually be augmented with zero-value data points to obtain an appropriate length with no insurmountable problems. In some applications, it is meaningful to subdivide the record into shorter length (possibly overlapping) records and obtain averages of several transforms. Attention will now be directed to algorithms for sequences of length $N = 2^p$.

Two basic versions of the algorithm are fairly easily derived, and the approach of Bergland [20] is followed. First, write the indices i and k in the binary notation

$$\begin{aligned} i &= i_{p-1}2^{p-1} + i_{p-2}2^{p-2} + \ldots + i_0 = (i_{p-1}, \ldots, i_0) \\ k &= k_{p-1}2^{p-1} + k_{p-2}2^{p-2} + \ldots + k_0 = (k_{p-1}, \ldots, k_0), \end{aligned} \tag{4.21}$$

where each component i_ν and k_ν takes on only the values 0 or 1. To illustrate the relatively complicated appearance of the indices, consider the case $N = 2^3 = 8$. Then for example, the number 5 is

$$i = 5_{10} = 1 \cdot 2^2 + 0 \cdot 2 + 1 \cdot 2^0 = 101_2 = (1, 0, 1),$$

where the subscript indicates the number base. Thus the components i_ν and k_ν are the binary digits arranged from the most significant bit to the least significant bit. The equation for the Fourier transform, Eq. (4.11), is now written below with the indices raised and enclosed in parentheses because of their complicated form:

$$Z(k_{p-1}, k_{p-2}, \ldots, k_0) =$$

$$Z(k_{p-1}2^{p-1} + k_{p-2}2^{p-2} + \ldots + k_0) = \sum_{i_0=0}^{1} \sum_{i_1=0}^{1} \cdots \sum_{i_{p-1}=0}^{1} \times z(i_{p-1}2^{p-1} + i_{p-2}2^{p-2} + \ldots i_0) W^{ik}$$

$$= \sum_{i_0=0}^{1} \sum_{i_1=0}^{1} \sum_{i_{p-1}=0}^{1} \times z(i_{p-1}, i_{p-2}, \ldots, i_0) W^{ik}, \quad (4.22)$$

where each successive summation consists of two terms. Now, as was done in Eq. (4.13) for two factors, the complex exponential can be expanded. When the exponent of W is expanded, it is seen that integral powers of $e^{-j2\pi}$ may be factored out and thus disappear since any integral power of $e^{-j2\pi}$ is unity. For example, consider the product of only the $k_{\ell-1}$ bit position in the k index with the entire i index:

$$W^{k_{\ell-1}2^{\ell-1}(i_{p-1}2^{p-1} + \ldots + i_0)} = W^{k_{\ell-1}2^{\ell-1}(i_{p-1}2^{p-1} + \ldots + i_{p-\ell+1}2^{p-\ell+1})} \times W^{k_{\ell-1}2^{\ell-1}(i_{p-\ell}2^{p-\ell} + \ldots + i_0)}$$

$$= W^{k_{\ell-1}2^{\ell-1}(i_{p-\ell}2^{p-\ell} + \ldots + i_0)}. \quad (4.23)$$

In the above equation, coefficients of bit positions of order $i_{p-\ell+1}$ or greater combine with the factor $2^{\ell-1}$. Since $2^{p-\ell+1} \cdot 2^{\ell-1} = 2^p = N$, the factor $N = 2^p$ in $\exp[-j2\pi/N]$ is cancelled, leaving integral powers of $\exp[-j2\pi]$. Hence, in each successive summation in Eq. (4.23), the dependence on the bit positions greater than $(p - \ell + 1)$ is eliminated. This allows successive portions of the complex exponential to be factored out of the inner summations as constants.

Recursive equations that have the ultimate effect of reducing the amount of arithmetic may now be directly obtained. Consider the innermost sum first, that is for $k = p$. Then, Eq. (4.22) becomes

$$Z(k_{p-1}, \ldots, k_0) = \sum_{i_0=0}^{1} \sum_{i_1=0}^{1} \cdots \sum_{i_{p-2}=0}^{1} \times \left[\sum_{i_{p-1}=0}^{1} z(i_{p-1}, \ldots, i_0) W^{k_0 i_{p-1}2^{p-1}} \right] \times W^{k(i_{p-2}2^{p-2} + \ldots + i_0)}. \quad (4.24)$$

The innermost sum is performed over the two values 0 and 1 of the bit i_{p-1}. Thus, the bracketed quantity in Eq. (4.24) is a function of $i_{p-2}, i_{p-3}, \ldots, i_0$ and k_0 and may be written as

$$A_1(k_0, i_{p-2}, i_{p-3}, \ldots, i_0) = \sum_{i_{p-1}=0}^{1} z(i_{p-1}, \ldots, i_0) W^{k_0 i_{p-1} 2^{p-1}}. \tag{4.25}$$

The second stage in the recursion is

$$A_2(k_0, k_1, i_{p-3}, \ldots, i_0) = \sum_{i_{p-2}=0}^{1} A_1(k_0, i_{p-2}, i_{p-3}, \ldots, i_0) W^{(k_1 2 + k_0) i_{p-2} 2^{p-2}}. \tag{4.26}$$

In general, for the ℓth stage

$$A_\ell(k_0, k_1, \ldots, k_{\ell-1}, i_{p-\ell-1}, \ldots, i_0) = \sum_{i_{p-\ell}=0}^{1} A_{\ell-1}(k_0, k_1, \ldots, k_{\ell-2}, i_{p-\ell}, \ldots, i_0) \times W^{(k_{\ell-1} 2^{\ell-1} + \ldots + k_0) i_{p-\ell} 2^{p-\ell}}. \tag{4.27}$$

This is termed the "Cooley-Tukey" fast Fourier transform algorithm. As noted in Refs. 19 and 20, an alternate form of the algorithm (the Sande-Tukey version) can be obtained if the roles of the indices are interchanged in Eq. (4.23). The recursion is

$$\hat{A}_\ell(k_0, k_1, \ldots, k_{\ell-1}, i_{p-\ell-1}, \ldots, i_0) = \sum_{i_{p-\ell}=0}^{1} \hat{A}_{\ell-1}(k_0, k_1, \ldots, k_{\ell-2}, i_{p-\ell}, \ldots, i_0) \times W^{k_{\ell-1} 2^{\ell-1} (i_{p-\ell} 2^{p-\ell} + \ldots + i_0)}. \tag{4.28}$$

This recursion corresponds to the two-factor derivation appearing earlier in this section. For both cases, one begins with the original data

$$A_0(i_{p-1}, i_{p-2}, \ldots, i_0) = \hat{A}_0(i_{p-1}, i_{p-2}, \ldots, i_0) = z(i_{p-1}, i_{p-2}, \ldots, i_0), \tag{4.29}$$

and ends at the pth application of the recursion with the Fourier coefficients

$$Z(k_{p-1}, k_{p-2}, \ldots, k_0) = A_p(k_0, k_1, \ldots, k_{p-1}) = \hat{A}_p(k_0, k_1, \ldots, k_{p-1}). \quad (4.30)$$

Notice that the final result has the binary components of its index reversed. Thus, if data have been stored in the memory location with the binary address corresponding to the index, a bit reversal must be performed to obtain the correct core memory address and to rearrange the results in their natural sequence.

Both recursions may be written in the form of a Fourier transform multiplied by a twiddle factor. It is simplest for the Sande-Tukey version:

$$\hat{A}_\ell(k_0, k_1, \ldots, k_{\ell-1}, i_{p-\ell-1}, \ldots, i_0)$$
$$= W^{(i_{p-\ell-1}2^{p-\ell-1} + \ldots + i_0)k_{\ell-1}2^{\ell-1}}$$
$$\times \sum_{i_{p-\ell}=0}^{1} \hat{A}_{\ell-1}(k_0, k_1, \ldots, k_{\ell-2}, i_{p-\ell}, i_0) W^{k_{\ell-1} i_{p-\ell} 2^{p-1}}. \quad (4.31)$$

The time variable is $i_{p-\ell}$ and the frequency variable $k_{\ell-1}$ for the two-term Fourier transform being computed at this stage of the recursion.

The Cooley-Tukey version requires a modification of the recursion. Let the first stage be

$$B_1(k_0, i_{p-2}, \ldots, i_0) = W^{k_0 i_{p-2} 2^{p-2}}$$
$$\times \sum_{i_{m-1}=0}^{1} A(i_{p-1}, \ldots, i_0) W^{k_0 i_{p-1} 2^{p-1}}. \quad (4.32)$$

The second stage is

$$B_2(k_0, k_1, i_{p-3}, \ldots, i_0)$$
$$= W^{(k_1 2 + k_0) i_{p-3} 2^{p-3}}$$
$$\times \sum_{i_{p-2}=0}^{1} B_1(k_0, i_{p-2}, \ldots, i_0) W^{i_{p-2} k_1 2^{p-1}}. \quad (4.33)$$

The ℓth stage of the recursion is

$$B_\ell(k_0, k_1, \ldots, k_{\ell-1}, i_{p-\ell-1}, \ldots, i_0)$$
$$= W^{(k_{\ell-1}2^{\ell-1} + \ldots + k_0) i_{p-\ell-1} 2^{p-\ell-1}}$$
$$\times \sum_{i_{p-\ell}=0}^{1} B_{\ell-1}(k_0, k_1, \ldots, k_{\ell-2}, i_{p-\ell}, \ldots, i_0) W^{i_{p-\ell} k_{\ell-1} 2^{p-1}}. \quad (4.34)$$

The quantity B_ℓ is related to A_ℓ in the other form of the recursion by

$$\begin{aligned} &B_\ell(k_0, k_1, \ldots, k_{\ell-1}, i_{p-\ell-1}, \ldots, i_0) \\ &\quad = A_\ell(k_0, k_1, \ldots, k_{\ell-1}, i_{p-\ell-1}, \ldots, i_0) \\ &\quad\quad \times W^{(k_{\ell-1}2^{\ell-1} + \ldots + k_0)i_{p-\ell-1}2^{p-\ell-1}}. \end{aligned} \tag{4.35}$$

Substituting Eq. (4.35) in Eq. (4.34), it follows that

$$\begin{aligned} &A_\ell(k_0, k_1, \ldots, k_{\ell-1}, i_{p-\ell-1}, \ldots, i_0) W^{(k_{\ell-1}2^{\ell-1} + \ldots + k_0)i_{p-\ell-1}2^{p-\ell-1}} \\ &\quad = W^{(k_{\ell-1}2^{\ell-1} + \ldots + k_0)i_{p-\ell-1}2^{p-\ell-1}} \\ &\quad\quad \times \sum_{i_{p-\ell}=0}^{1} A_{\ell-1}(k_0, k_1, \ldots, k_{\ell-2}, i_{p-\ell}, \ldots, i_0) \\ &\quad\quad \times W^{(k_{\ell-2}2^{\ell-2} + \ldots + k_0)i_{p-\ell}2^{p-\ell}} W^{i_{p-\ell}k_{\ell-1}2^{p-1}} \\ &\quad = W^{(k_{\ell-1}2^{\ell-1} + \ldots + k_0)i_{p-\ell-1}2^{p-\ell-1}} \\ &\quad\quad \times \sum_{i_{p-\ell}=0}^{1} A_{\ell-1}(k_0, k_1, \ldots, k_{\ell-2}, i_{p-\ell}, \ldots, i_0) \\ &\quad\quad \times W^{(k_{\ell-1}2^{\ell-1} + k_{\ell-2}2^{\ell-2} + \ldots + k_0)i_{p-\ell}2^{p-\ell}}. \end{aligned} \tag{4.36}$$

Thus, the recursions are equivalent except at the final stage where the twiddle factor is omitted.

The summation appearing in both recursions can easily be written out in full to obtain one final form for programming.

a. The Cooley-Tukey version:

$$\begin{aligned} &B_\ell(k_0, k_1, \ldots, k_{\ell-2}, \overset{k_{\ell-1}}{\underset{\downarrow}{0}}, i_{p-\ell-1}, \ldots, i_0) \\ &\quad = [B_{\ell-1}(k_0, k_1, \ldots, k_{\ell-2}, \overset{i_{p-\ell}}{\underset{\downarrow}{0}}, i_{p-\ell-1}, \ldots, i_0)] \\ &\quad\quad + B_{\ell-1}(k_0, k_1, \ldots, k_{\ell-2}, 1, i_{p-\ell-1}, \ldots, i_0)] \\ &\quad\quad \times W^{(k_{\ell-2}2^{\ell-2} + \ldots + k_0)i_{p-\ell-1}2^{p-\ell-1}}, \end{aligned} \tag{4.37a}$$

$$B_\ell(k_0, \ldots, k_{\ell-2}, \overset{k_{\ell-1}}{\underset{\downarrow}{1}}, i_{p-\ell-1}, \ldots, i_0)$$

$$= [B_{\ell-1}(k_0, k_1, \ldots, k_{\ell-2}, \overset{i_{p-\ell}}{\underset{\downarrow}{0}}, i_{p-\ell-1}, \ldots, i_0)$$

$$- B_{\ell-1}(k_0, k_1, \ldots, k_{\ell-2}, 1, i_{p-\ell-1}, \ldots, i_0)]$$

$$\times W^{(k_{\ell-2}2^{\ell-2} + \ldots + k_0) i_{p-\ell-1} 2^{p-\ell-1}},$$

$$\ell = 1, 2, \ldots, p. \tag{4.37b}$$

b. The Sande-Tukey version:

$$\hat{A}_\ell(k_0, k_1, \ldots, k_{\ell-2}, \overset{k_{\ell-1}}{\underset{\downarrow}{0}}, i_{p-\ell-1}, \ldots, i_0)$$

$$= [\hat{A}_{\ell-1}(k_0, k_1, \ldots, k_{\ell-2}, \overset{i_{p-\ell}}{\underset{\downarrow}{0}}, i_{p-\ell-1}, \ldots, i_0)$$

$$+ \hat{A}_{\ell-1}(k_0, k_1, \ldots, k_{\ell-2}, 1, i_{p-\ell-1}, \ldots, i_0)]$$

$$\times W^{(i_{p-\ell-1}2^{p-\ell-1} + \ldots + i_0)\ \overset{k_{\ell-1}}{\underset{\downarrow}{(0)}}\ 2^{\ell-1}} \tag{4.38a}$$

$$\hat{A}_\ell(k_0, k_1, \ldots, k_{\ell-2}, \overset{k_{\ell-1}}{\underset{\downarrow}{1}}, i_{p-\ell-1}, \ldots, i_0)$$

$$= [\hat{A}_{\ell-1}(k_0, k_1, \ldots, k_{\ell-2}, \overset{i_{p-\ell}}{\underset{\downarrow}{0}}, i_{p-\ell-1}, \ldots, i_0)$$

$$- \hat{A}_{\ell-1}(k_0, k_1, \ldots, k_{\ell-2}, 1, i_{p-\ell-1}, \ldots, i_0)]$$

$$\times W^{(i_{p-\ell-1}2^{p-\ell-1} + \ldots + i_0)\ \overset{k_{\ell-1}}{\underset{\downarrow}{(1)}}\ 2^{\ell-1}}. \tag{4.38b}$$

This form illustrates that the difference between the two versions can be characterized by the twiddle factors, which are

a. Cooley-Tukey (C-T)

$$T_1 = W^{(k_{\ell-2}2^{\ell-2} + \ldots + k_0)i_{p-\ell-1}2^{p-\ell-1}} \tag{4.39}$$

b. Sande-Tukey (S-T)

$$T_2 = W^{(i_{p-\ell-1}2^{p-\ell-1} + \ldots + i_0)k_{\ell-1}2^{\ell-1}} \tag{4.40}$$

In principle, the two versions require the same amount of arithmetic. In practice, the Sande-Tukey version is sometimes simpler because of the exponent for W. In both the C-T and S-T versions, from Eqs. (4.37) and (4.38) it is seen that the index required for B_ℓ or A_ℓ can be broken into three parts:

$$\overbrace{k_0 2^{p-1} + k_1 2^{p-2} + \ldots + k_{\ell-2}2^{p-\ell-3}}^{\text{Part 1}} \overbrace{+ k_{\ell-1}2^{p-\ell-2}}^{\text{Part 2}} \overbrace{+ i_{p-\ell-1}2^{p-\ell-1} + \ldots + i_0}^{\text{Part 3}}. \tag{4.41}$$

Note that exponent $(k_{\ell-2}2^{\ell-2} + \ldots + k_0)i_{p-\ell-1}2^{p-\ell-1}$ in the C-T twiddle factor is a bit-reversed version of the first part of the index (Eq. 4.41) multiplied by a power of two. In the S-T version, the exponent $(i_{p-\ell-1} \cdot 2^{p-\ell-1} + \ldots + i_0)2^{p-1}$ is the third part of the index (Eq. 4.41)–not bit-reversed–multiplied by a power of two. Thus, in some instances it is easier to perform the tally necessary to obtain the complex exponential argument for the S-T version than for the C-T version.

The equations for the algorithms may appear simpler if some specific examples for the binary indices are illustrated. Consider the case when $p = 4$ and $N = 2^4 = 16$. Then the decimal and binary indices are

DECIMAL		BINARY		
$z(0_{10})$	=	$z(0\cdot 2^3 + 0\cdot 2^2 + 0\cdot 2 + 0)$	=	$z(0,0,0,0)$,
$z(1_{10})$	=	$z(0\cdot 2^3 + 0\cdot 2^2 + 0\cdot 2 + 1)$	=	$z(0,0,0,1)$,
$z(10_{10})$	=	$z(1\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2 + 0)$	=	$z(1,0,1,0)$,
$z(15_{10})$	=	$z(1\cdot 2^3 + 1\cdot 2^2 + 1\cdot 2 + 1)$	=	$z(1,1,1,1)$.

Further consider the indices involved in B_ℓ for the ℓ = 2nd stage of the recursion. Equation (4.37a) and (4.37b) become for this case

$$B_2(k_0,0,i_1,i_0) = [B_1(k_0,0,i_1,i_0) + B_1(k_0,1,i_1,i_0)]W^{k_0 i_0 2},$$

$$B_2(k_0,1,i_1,i_0) = [B_1(k_0,0,i_1,i_0) - B_1(k_0,1,i_1,i_0)]W^{k_0 i_0 2}.$$

The new results B_2 are stored as indicated in Fig. 4.4.

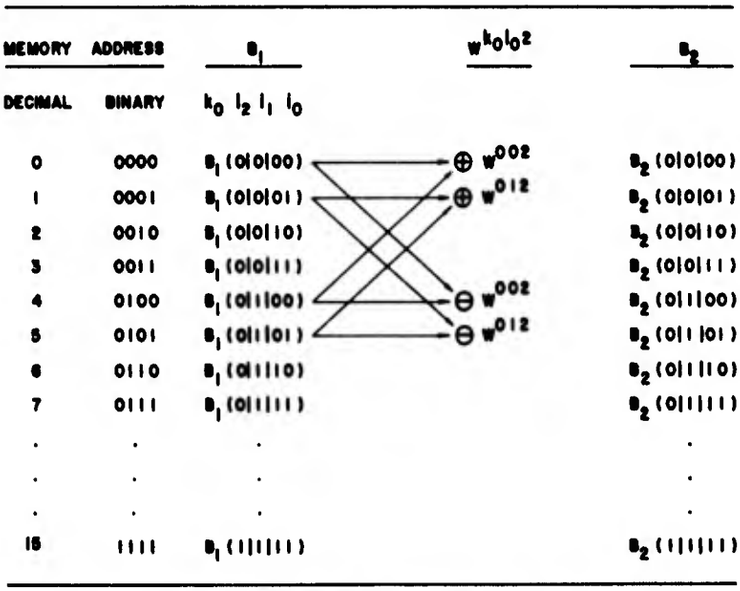


Fig. 4.4. Illustration of storage at second stage of recursion for $p = 4$.

In some respects, the programming necessary to implement the fast Fourier transform algorithm reduces to a fairly straightforward procedure. First consider the diagram in Fig. 4.5, which is an extension of Fig. 4.4. This diagram is for the case $N = 2^3$, $p = 3$ and is taken from McCowan [21]. Solid lines indicate

that the quantity in the location at the beginning of the solid line is multiplied by W, raised to the power in the circle, added to the quantity from the dashed line, and stored in the indicated location. For example, in the first iteration $x(1, 0, 0)$ is multiplied by W^0, added to $z(0, 0, 0)$, and stored in location with binary address (0, 0, 0). In equation form,

$$z_{1(0)} = z_{0(0)} + z_{0(4)} W^0.$$

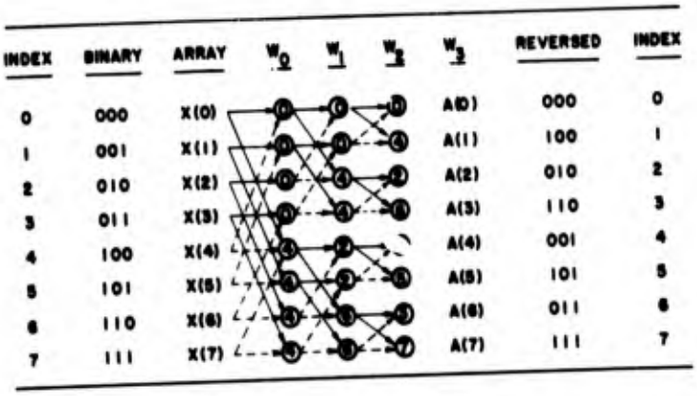


Fig 4.5. C-T algorithm diagram, $N = 8$, $p = 3$.

This diagram is an implementation of Eq. (4.27). The relation $W^{N/2} = -1$ is used, as there are always pairs of exponentials differing by $N/2$. Thus, only one multiply is performed, followed by an add in one case and a subtract in another. Note also the patterns that appear in the diagram, in that operations are performed on blocks of data whose lengths are divided by two at each stage of the recursion. Facts such as these simplify determining the locations in which data must be stored and in turn simplify the programming.

The following rules for the algorithm are due to C. M. Rader, according to McCowan [21]. The exponents of the complex exponential are determined from the following rule:

1. The number in the circle of the jth node in the ℓth array (for the ℓth recursion) is found by (1) writing the binary number j, (2) scaling (shifting) it by $(p - \ell)$ places to the right, and (3) reversing the order of the p bits. Thus, in array 2, the last element, $B_2(7)$, has binary address $7_{10} = 111_2$. Scaling $p - 2 = 1$ place to the right gives 011, and reversing the order of the bits gives $110_2 = 6_{10}$. This rule follows directly from an examination of the exponent of W in Eq. (4.27).

The origin of the solid and dashed lines is determined by the following rule:

2. In the ℓth array, node j has a solid line from a node in array $(\ell - 1)$ whose location is the same except that the bit in position $(p - \ell)$ must be a one. The

dashed-arrow origin is determined similarly. The only difference is that the bit in position $(p - \ell)$ must be a zero.

These rules can be compared with Eq. (4.27) and their origin determined by careful inspection of the subscripts. The final iteration indicated in Fig. 4.5 is the rearrangement obtained by reversing the order of the binary digits of an index. In an actual program, the bit-reversal procedure can be accomplished in various ways:

1. A reverse tally can be maintained where high-order bits, rather than low-order bits, are added in.
2. The following two-step recursive procedure can be employed:

(a) Test leading bits of the previous index, setting them equal to zero until a zero bit is found.

(b) Set this zero bit to a one bit, and this is the desired index.

For example, when $p = 3$, we begin with $000_2 = 0_{10}$. Obtained from this is $100_2 = 4_{10}$, which is $001_2 = 1_{10}$ with bits reversed. Obtained next is $010_2 = 2_{10}$, etc. The authors have been told that computer instructions exist to accomplish this, but know of no specific examples. Note that the S-T algorithm does not require a bit reversal for the complex exponential argument, whereas the C-T version does. Both versions require bit reversals to rearrange the final results.

The complex exponentials can be determined in advance and stored in a table or computed as the calculations proceed. In generating in advance, the recursion

$$e^{-j\frac{2\pi ik}{N}} = e^{-j\frac{2\pi(i-1)k}{N}} e^{-j\frac{2\pi}{N}}$$

$$W^{ik} = W^{(i-1)k} W \tag{4.42}$$

should be used. In terms of sines and cosines, this is Eq. (4.7). If the complex exponentials are computed as required, standard sine/cosine subroutines are employed. This does not normally result in any significant loss of computational efficiency. This is true since only N complex exponentials are necessary. The computing time for an IBM 7094 system sine/cosine subroutine is about 250×10^{-6} seconds. Thus the time for an 8192-point case would be about

$$T = 8192 \times 250 \times 10^{-6} < 10^4 \times 250 \times 10^{-6} = 2.5 \text{ sec.}$$

Computational times of such magnitude are easily lost in the system overhead.

4.3 Matrix Formulation of Algorithm

In McCowan [21], an alternate approach to a characterization of an FFT algorithm is presented. This explanation is in terms of factoring a matrix. It is analogous to decomposing an aircraft maneuver into its roll, pitch, and yaw

components. A rotation of plane coordinates through an angle θ can be written in matrix form as

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}. \tag{4.43}$$

In three dimensions, any general rotation can be written in terms of three two-dimensional components (see Fig. 4.6), roll (rotation in $[y, z]$ coordinate plane), pitch (rotation in $[x, z]$ coordinate plane), and yaw (rotation in $[y, z]$ plane).

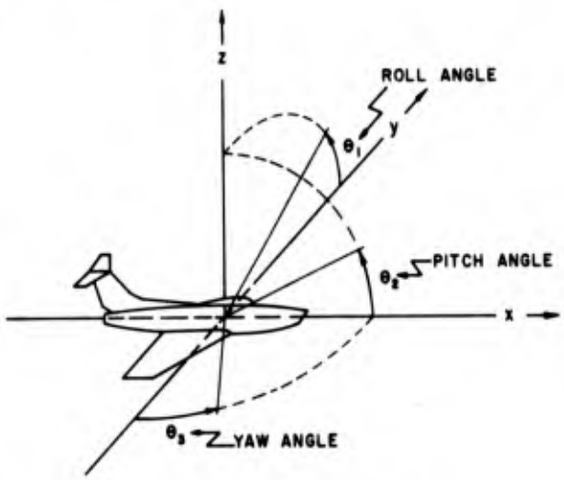


Fig. 4.6. Illustration of roll, pitch, and yaw components.

Thus the total rotation is the product of three two-dimensional matrices

$$\mathbf{T}(\theta_1,\theta_2,\theta_3) = \underset{\text{yaw}}{\begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 \\ \sin\theta_3 & \cos\theta_3 & 0 \\ 0 & 0 & 1 \end{bmatrix}} \underset{\text{pitch}}{\begin{bmatrix} \cos\theta_2 & 0 & -\sin\theta_2 \\ 0 & 1 & 0 \\ \sin\theta_2 & 0 & \cos\theta_2 \end{bmatrix}}$$

$$\times \underset{\text{roll}}{\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_1 & -\sin\theta_1 \\ 0 & \sin\theta_2 & \cos\theta_2 \end{bmatrix}} \tag{4.44}$$

The actual computations necessary for performing a transformation of coordinates can thus be broken down into three two-dimensional matrix-vector multiplications rather than one three-dimensional matrix-vector multiplication. The number of operations for the single three-dimensional multiplication is

$$\text{No. Ops. (3-dimen.)} = 3^2,$$

while for three two-dimensional multiplications,

$$\text{No. Ops. (3 2-dimen.)} = 2^2 + 2^2 + 2^2 = 2(2+2+2) = 12.$$

If the above is generalized mathematically to four dimensions,

$$\text{No. Ops. (4-dimen.)} = 4^2 = 16$$

$$\text{No. Ops. (4 2-dimen.)} = 2^2 + 2^2 + 2^2 + 2^2 = 2(2+2+2+2) = 16.$$

For a five-dimensional rotation,

$$\text{No. Ops. (5-dimen.)} = 5^2 = 25,$$

$$\text{No. Ops. (5 2-dimen.)} = 2(2+2+2+2+2) = 20.$$

The time savings that can potentially materialize from this approach when the dimension of the transformation becomes large enough are readily seen. Roughly speaking, this is what happens for fast Fourier transform algorithms. A discrete, finite Fourier transform of N data points may be viewed as a rotation in N-dimensional space. In mathematical terms, it is an orthogonal linear transformation. The exponent for W also specifies its position in the matrix. In matrix equation form,

$$\begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ \vdots \\ X_{N-1} \end{bmatrix} = \begin{bmatrix} W^{0\cdot 0} & W^{0\cdot 1} & W^{0\cdot 2} & \cdots & W^{0\cdot(N-1)} \\ W^{1\cdot 0} & W^{1\cdot 1} & W^{1\cdot 2} & \cdots & W^{1\cdot(N-1)} \\ W^{2\cdot 0} & W^{2\cdot 1} & W^{2\cdot 2} & \cdots & W^{2\cdot(N-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ W^{(N-1)\cdot 0} & W^{(N-1)\cdot 1} & W^{(N-1)\cdot 2} & \cdots & W^{(N-1)(N-1)} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N-1} \end{bmatrix} \tag{4.45}$$

or in matrix notation,

$$\mathbf{X} = \mathbf{W}\mathbf{x}. \tag{4.46}$$

For the special case $N = 2^3 = 8, p = 3$, the matrix is

$$\mathbf{W} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & W^1 & W^2 & W^3 & W^4 & W^5 & W^6 & W^7 \\ 1 & W^2 & W^4 & W^6 & W^8 & W^{10} & W^{12} & W^{14} \\ 1 & W^3 & W^6 & W^9 & W^{12} & W^{15} & W^{18} & W^{21} \\ 1 & W^4 & W^8 & W^{12} & W^{16} & W^{20} & W^{24} & W^{28} \\ 1 & W^5 & W^{10} & W^{15} & W^{20} & W^{25} & W^{30} & W^{35} \\ 1 & W^6 & W^{12} & W^{18} & W^{24} & W^{30} & W^{36} & W^{42} \\ 1 & W^7 & W^{14} & W^{21} & W^{28} & W^{35} & W^{42} & W^{49} \end{bmatrix}$$

This matrix can be factored into $p + 1 = 4$ matrices

$$\mathbf{W} = \mathbf{W}_3 \mathbf{W}_2 \mathbf{W}_1 \mathbf{W}_0, \tag{4.47}$$

or, in general, the factorization is

$$\mathbf{W} = \mathbf{W}_p \mathbf{W}_{p-1} \ldots \mathbf{W}_1 \mathbf{W}_0. \tag{4.48}$$

The last matrix, $\mathbf{W}_p$, is always a permutation matrix, which performs the reversed-bit memory location reordering.

The W matrices for the Cooley-Tukey factorization for the special case $N = 2^3$ are

$$W_0 = \begin{bmatrix} 1 & 0 & 0 & 0 & W^0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & W^0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & W^0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & W^0 \\ 1 & 0 & 0 & 0 & W^4 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & W^4 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & W^4 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & W^4 \end{bmatrix}$$

$$W_1 = \begin{bmatrix} 1 & 0 & W^0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & W^0 & 0 & 0 & 0 & 0 \\ 1 & 0 & W^4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & W^4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & W^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & W^2 \\ 0 & 0 & 0 & 0 & 1 & 0 & W^6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & W^6 \end{bmatrix}$$

$$W_2 = \begin{bmatrix} 1 & W^0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & W^4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & W^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & W^6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & W^1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & W^5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & W^3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & W^7 \end{bmatrix}$$

$$W_3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The term W^0 is, of course, the same as unity. It was used rather than one to illustrate the symmetry. When the programming is specialized to take advantage of the specific forms of the matrices (i.e., does not multiply by zeroes or ones) the time savings of the algorithm result.

4.4 Important Related Formulas

In the application of fast Fourier transforms to spectral and correlation analysis, a number of specialized relations regarding finite discrete Fourier transforms are useful. Many of these relations are noted here. A detailed discussion of their application to correlation and spectra is left to Chapters 5 and 6.

The basic FFT algorithms are developed for complex data sequences. When applied to time series data analysis, real data are collected and, thus, transforms of real sequences are desired. Two real transforms may be computed simultaneously by inserting one time series in the real part and one in the imaginary part of the complex sequence to be transformed. In equation form,

$$z_i = x_i + jy_i, \qquad i = 0, 1, \ldots, N-1. \tag{4.49}$$

The transforms of x_i and y_i then are

$$X_k = \frac{Z_k + Z^*_{(N-k)}}{2} \tag{4.50}$$

and

$$Y_k = \frac{Z_k - Z^*_{(N-k)}}{2j}, \qquad k = 0, 1, \ldots, N/2, \tag{4.51}$$

where $Z_N = Z_0$, since the transform of a real data sequence is periodic with a period equal to that of the Nyquist frequency. Equations (4.50) and (4.51) are only unique out to $k = N/2$. That is, in terms of discrete transforms, if N real data points are transformed to N frequency points at spacing $1/T$, the Nyquist frequency occurs at the $(N/2)$th frequency point.

The derivation of Eqs. (4.50) and (4.51) is straightforward. First, the transform is

$$Z_k = \sum_{i=0}^{N-1} [x_i + jy_i]\, e^{-j\frac{2\pi ik}{N}}, \qquad k = 0, 1, \ldots, N-1, \tag{4.52}$$

where

$$\Delta f = \frac{1}{T} = \frac{1}{N\Delta t}. \tag{4.53}$$

Note that

$$e^{-j\frac{2\pi i(N-k)}{N}} = e^{-j\pi i}\, e^{j\frac{2\pi ik}{N}} = e^{j\frac{2\pi ik}{N}}, \tag{4.54}$$

since $e^{-j2\pi i} = 1$. Equation (4.50) is obtained directly:

$$[Z_k + Z^*_{(N-k)}] = \sum_{i=0}^{N-1} [x_i + jy_i]\; e^{-j\frac{2\pi ik}{N}} + \sum_{i=0}^{N-1} [x_i + jy_i]\; e^{-j\frac{2\pi ik}{N}}$$

$$= 2\sum_{i=0}^{N-1} x_i\; e^{-j\frac{2\pi ik}{N}}$$

$$= 2X_k, \qquad k = 0, 1, \ldots, N/2 - 1.$$

Equation (4.51) is obtained similarly.

The fast Fourier transform algorithms are naturally adapted to methods for doubling the length of data sequences that may be accommodated by the basic algorithm. For $N = 2^p$, each stage of the algorithm divides by two the number of points in the sequence upon which the corresponding final results depend. For example, the even-indexed results (before reverse-bit unscrambling) are stored in the first half of the memory allocated for data. Thus the even indexed results can be obtained from one pass through an algorithm based on half the storage. The odd-indexed results could be based on a second pass. As can be seen by analysis of the above statements, sequences decimated by two must be dealt with in order to obtain double length transforms. The most convenient external storage therefore is a semirandom access device, such as a disk or drum. A sequential device, such as a tape, is relatively inconvenient.

A double length transform may be computed in the following manner. Suppose the algorithm is for $N/2$ data points where $N = 2^p$. Then, if the data are z_i, $i = 0, 1, \ldots, N - 1$, and

$$W = \exp\left[\frac{-j2\pi}{N}\right],$$

the complex exponential for the algorithm of length $N/2$ is

$$W^2 = \exp\left[\frac{-j2\pi}{N/2}\right]. \tag{4.55}$$

Two separate transforms of the even- and odd-indexed data points, respectively, have now been computed.

$$A_k = \sum_{i=0}^{N/2-1} z_{2i}(W^2)^{ik}, \tag{4.56a}$$

$$B_k = \sum_{i=0}^{N/2-1} z_{2i+1}(W^2)^{ik}, \qquad k = 0, 1, \ldots, N/2 - 1. \tag{4.56b}$$

It can be shown that the full N-point transform of z_i is related to A_k and B_k by

$$Z_k = A_k + B_k\, W^k, \tag{4.57a}$$

$$Z_{N/2+k} = A_k - B_k\, W^k, \qquad k = 0, 1, \ldots, N/2 - 1. \tag{4.57b}$$

The above procedure could be repeated to allow successive doubling of the number of data points.

The proof of Eqs. (4.57a and b) is as follows:

$$\begin{aligned} Z_k &= A_k + B_k\, W^k \\ &= \sum_{i=0}^{N/2-1} z_{2i}(W^2)^{ik} + \sum_{i=0}^{N/2-1} z_{2i+1}(W^2)^{ik}\, W^k \\ &= \sum_{i=0}^{N/2-1} z_{2i}(W^2)^{ik} + \sum_{i=0}^{N/2-1} z_{2i+1}\, W^{(2i+1)k} \\ &= z_0\, W^{0\cdot k} + z_2\, W^{2\cdot k} + \ldots + z_{N-2}\, W^{(N-2)k} \\ &\quad + z_1\, W^{1\cdot k} + z_3\, W^{3\cdot k} + \ldots + z_{N-1}\, W^{(N-1)k} \\ &= \sum_{i=0}^{N-1} z_i\, W^{ik}, \qquad k = 0, 1, \ldots, N/2 - 1. \end{aligned}$$

This gives Eq. (4.56a). To obtain Eq. (4.56b), note that

$$A\left(\frac{N}{2}+k\right) = \sum_{i=0}^{N/2-1} z_{2i}(W^2)^{\left(i\frac{N}{2}+k\right)}$$

$$= \sum_{i=0}^{N/2-1} z_{2i}(W^2)^{ik}(W^2)^{i\frac{N}{2}}$$

$$= \sum_{i=0}^{N/2-1} z_{2i}(W^2)^{ik}$$

$$= A_k,$$

since

$$(W^2)^{i\frac{N}{2}} = W^{iN} = \exp\,[-j2\pi i] = 1.$$

Similarly, it can be shown that

$$B\left(\frac{N}{2}+k\right) = B_k.$$

Finally

$$W^{\frac{N}{2}+k} = W^{\frac{N}{2}} W^k = -W^k,$$

since

$$W^{\frac{N}{2}} = \exp\left(-j\frac{2\pi}{2}\right) = -1.$$

Then Eq. (4.56b) follows directly since

$$Z\left(\frac{N}{2}+k\right) = A\left(\frac{N}{2}+k\right) + B\left(\frac{N}{2}+k\right) W^{\left(\frac{N}{2}+k\right)}.$$

These formulas have direct application to the computation of spectral density functions and correlation functions to be discussed in Chapters 5 and 6.

Care must be used if a factor of $1/N$ for an N-point Fourier transform has been employed in the basic definition. Then A_k and B_k, being $N/2$-point transforms, must have a factor of $2/N$. But Z_k would require a factor of $1/N$. Thus, the right-hand side of Eqs. (4.57a and b) would be multiplied by 1/2 to account for this.

The actual coding of the basic fast Fourier transform algorithm reduces to a simple procedure (see Fig. 4.7). Basically, an inner loop is obtained which is executed first $N/2$ times, then $N/4$, etc. That is, the number of times through the

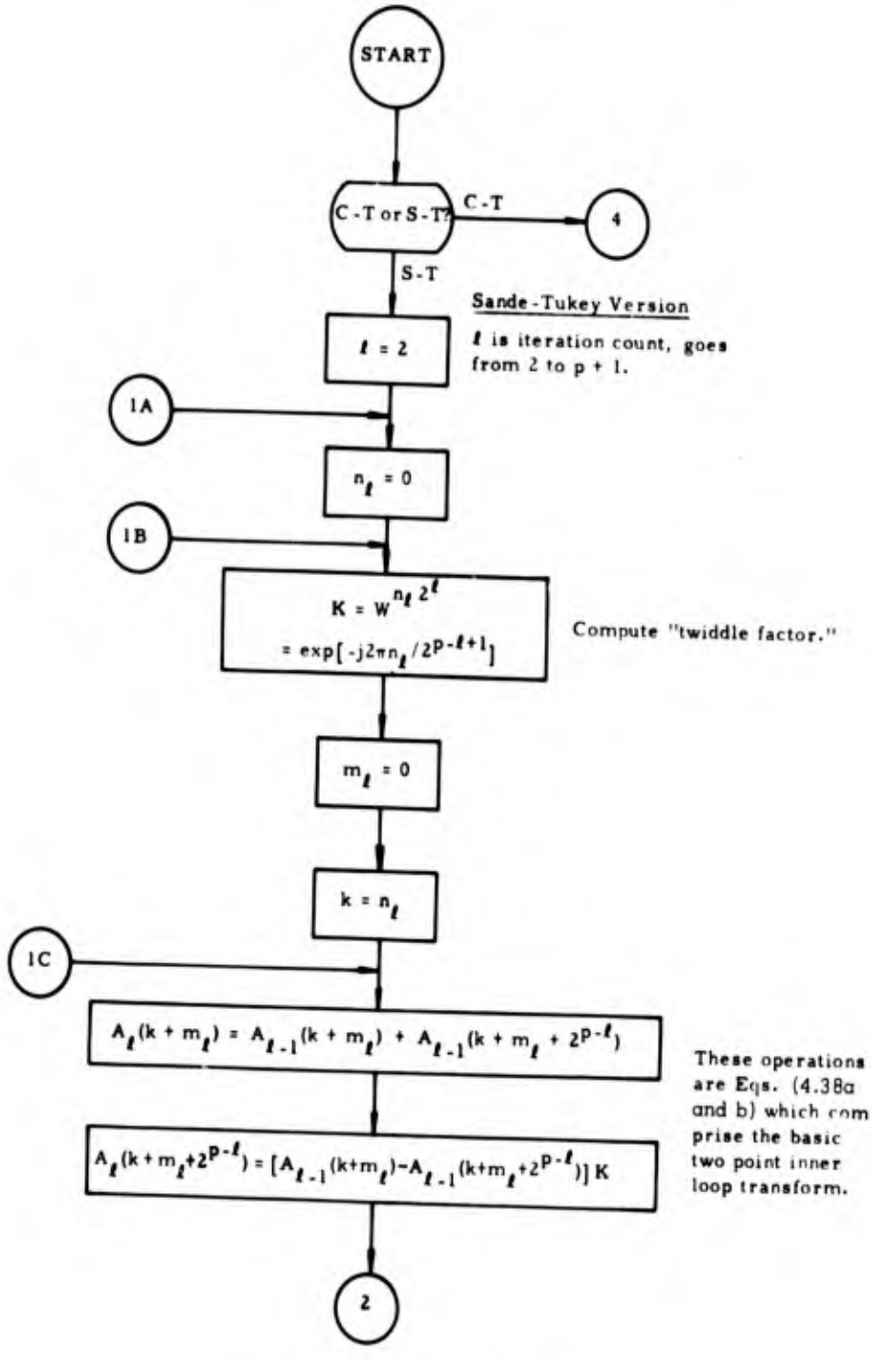


Fig. 4.7. Flow chart for basic FFT algorithm.

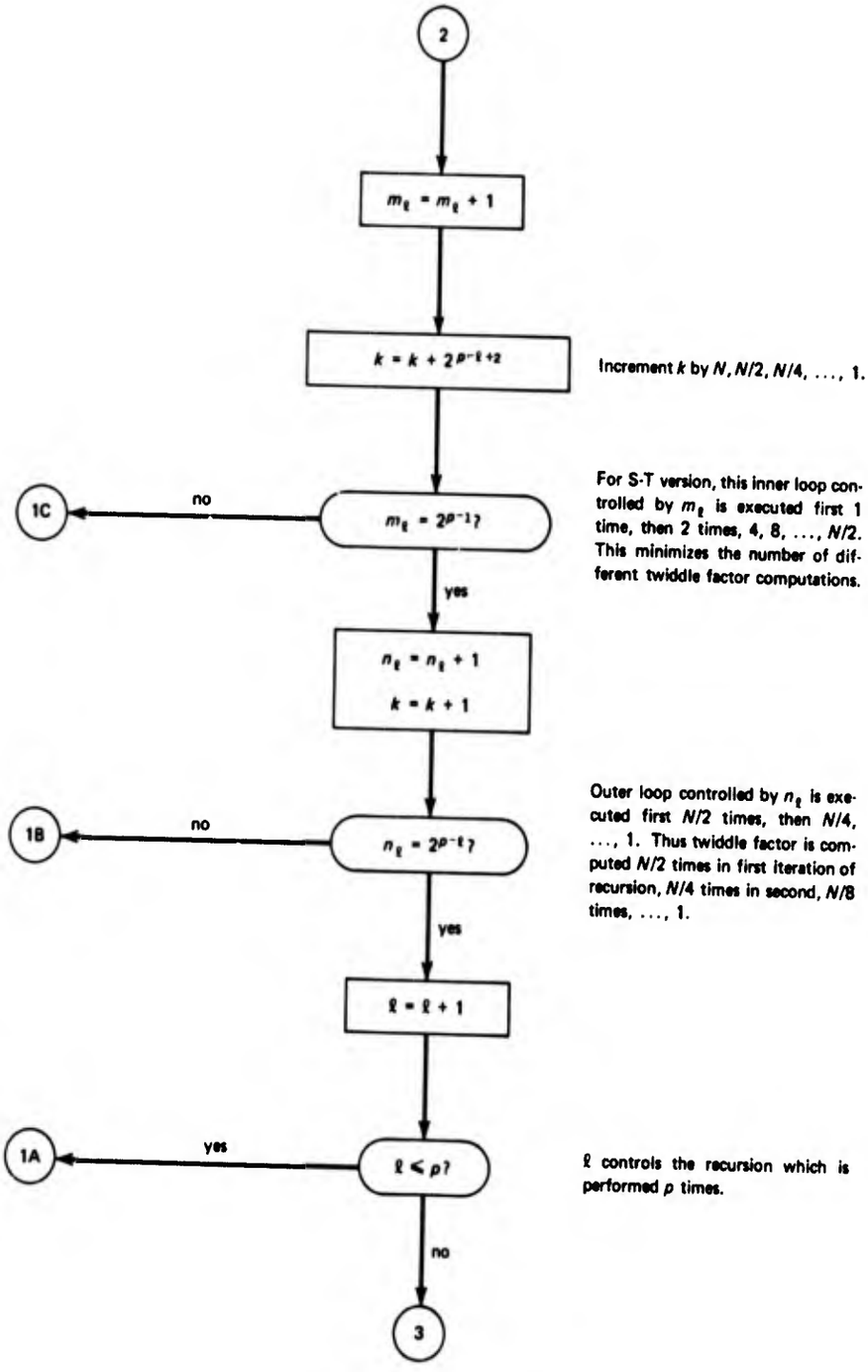


Fig. 4.7. Flow chart for basic FFT algorithm.

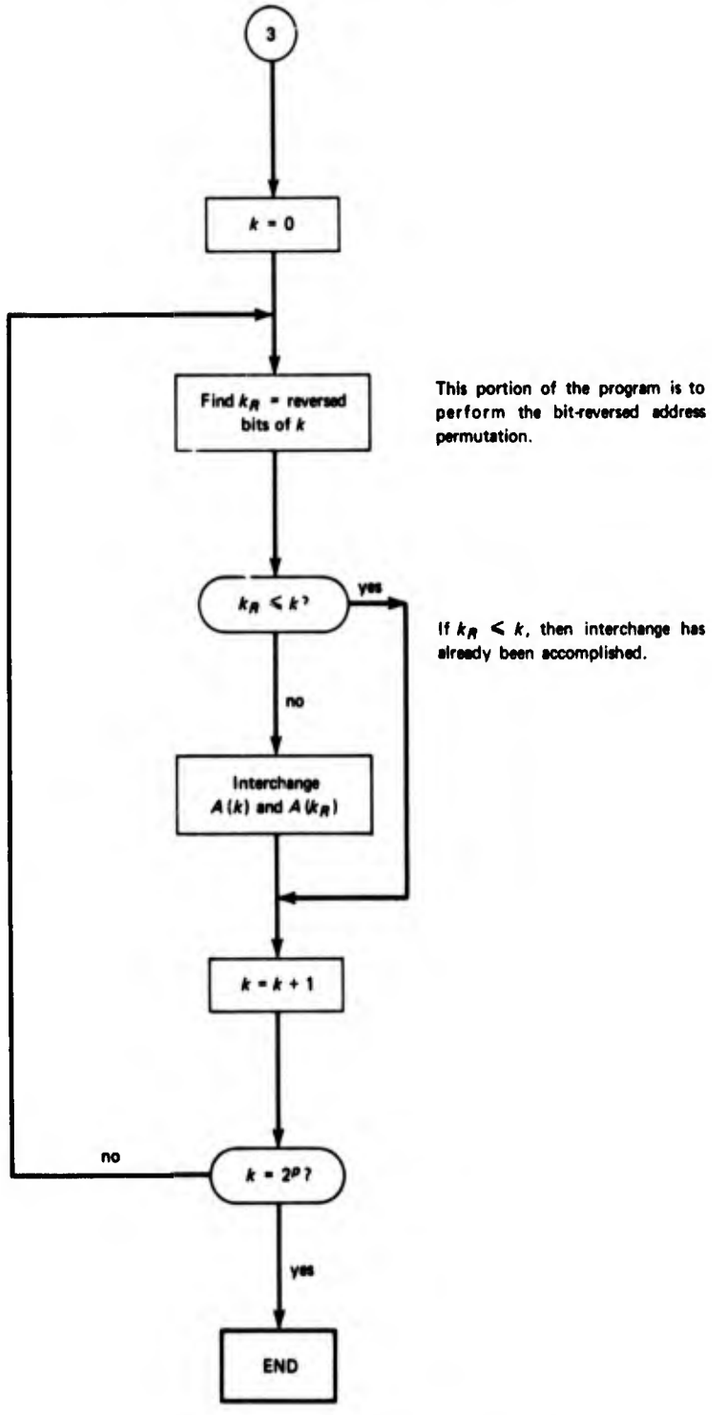


Fig. 4.7. Flow chart for basic FFT algorithm.

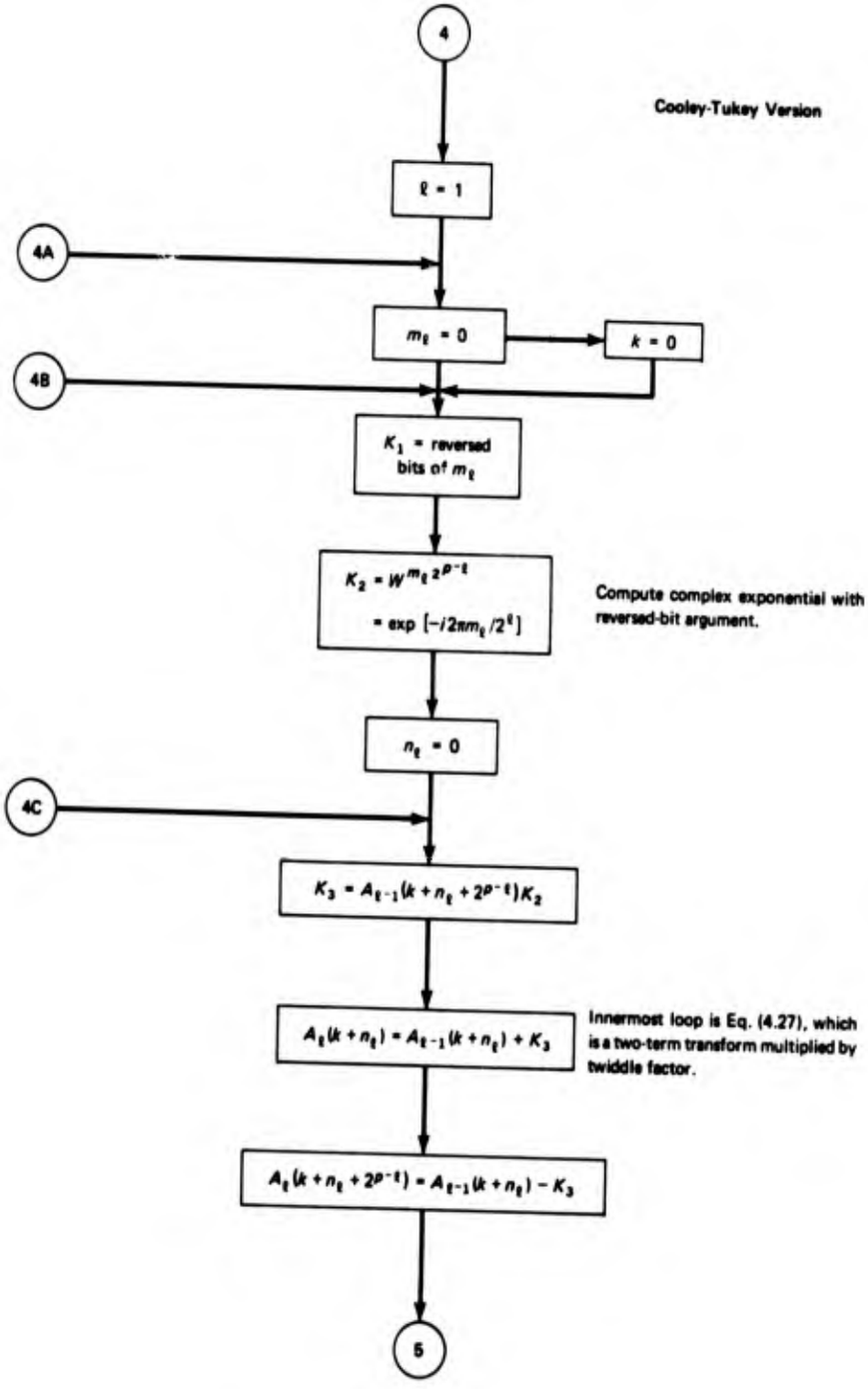


Fig. 4.7. Flow chart for basic FFT algorithm.

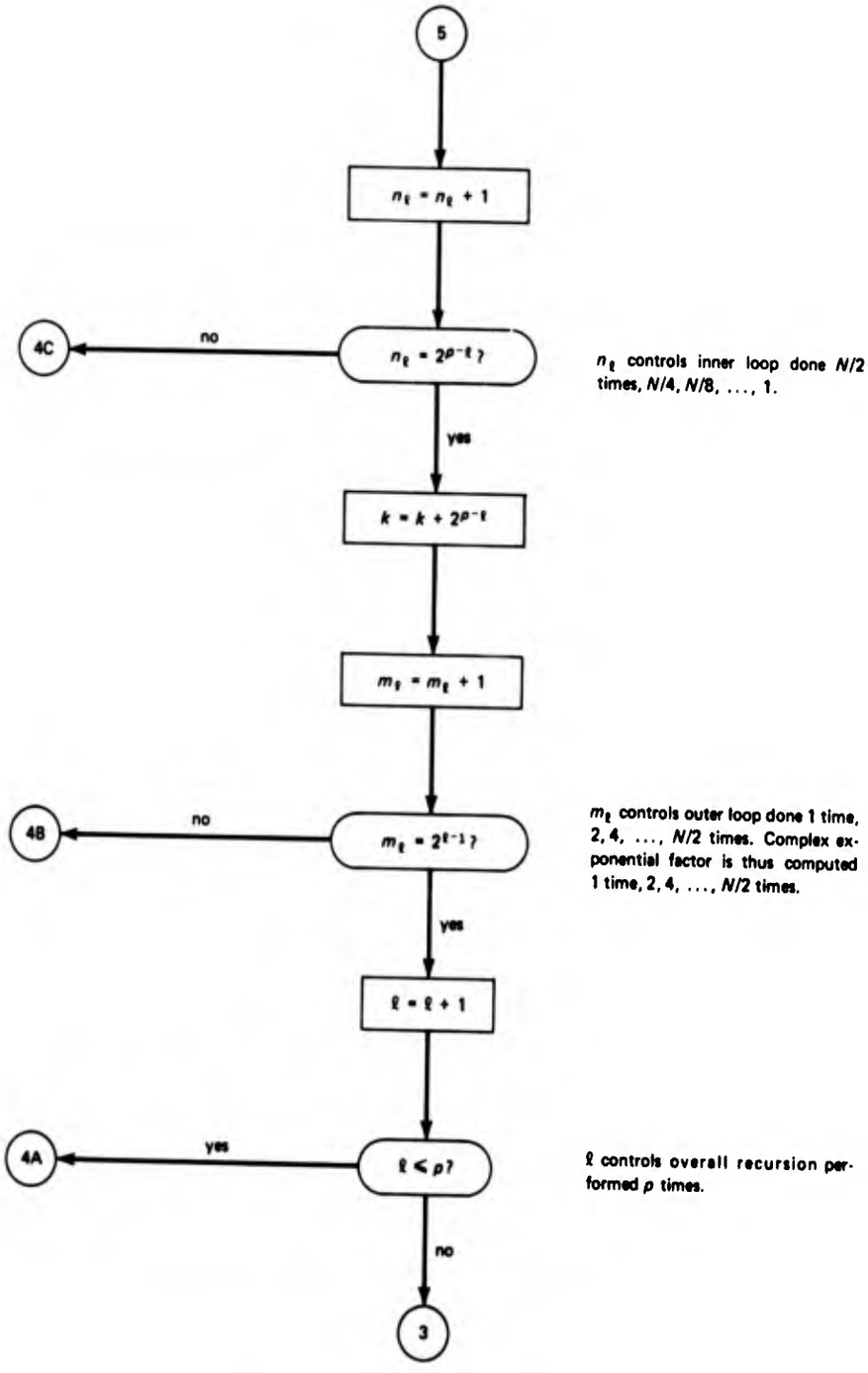


Fig. 4.7. Flow chart for basic FFT algorithm.

loop is divided by two each time. An outer loop is also used, and the number of times through the loop is doubled each time, e.g., 1, 2, 4, etc. The computation of the twiddle factor is minimized since it need only be computed once for each execution of the outer loop. These loops are illustrated in the flow chart, which is Fig. 4.7.

One point to note is that the innermost computation is a two-term, complex Fourier transform. However, only complex addition and subtraction need be used, rather than the much more time-consuming complex multiplication. This is true since the only values of the complex exponential involved are +1 and −1.

CHAPTER 5
CORRELATION FUNCTION COMPUTATIONS

5.1 Basic Concepts

This chapter contains a discussion of correlation function computations. Both auto- and crosscorrelation functions are naturally included since once the computational capability exists for one, the other is available with very little additional work.

Correlation functions have wide applications in shock and vibration analysis. When power or cross spectra can be applied to the analysis of a problem, the corresponding correlation functions can, in principle, be used in an equivalent manner since they are Fourier transform pairs. In practice there may be a strong reason for choosing one or the other. For example, the time delay between two signals can be determined from the phase of a cross spectrum, but would often show up in a more natural manner in the crosscorrelation function.

The computation of correlation functions has in the past required large amounts of computer time. The primary computational loop involved is a multiply-add sequence which often must be executed on the order of 10^7 times. This means, for example, that on a modern high-speed, large-scale digital computer where a fixed-point, multiply-add loop could be executed in, say, 20 microseconds, a total of 200 seconds, or 3 minutes and 20 seconds, would be required for just the execution of this loop. That would be the computational time required for just one single data record. In many physical applications, the number of data records to be analyzed both individually and jointly can run into fairly large numbers. Hence, there has been a real need to give attention to this correlation computation loop to make it less time consuming and more efficient.

It has been shown by Sande [22] that the computational speed for correlation functions can sometimes be reduced via the indirect route of using fast Fourier transforms. Many other specialized methods, based on other considerations, have been developed and may be more economical than the FFT's from one standpoint or another, depending on specific circumstances. For example, if one has invested in a high-speed, special purpose, multiply-add unit, such as those available from many computer manufacturers at the present time, straightforward computations may be the most economical. Direct programming of the correlation equation may be by far the simplest. Also, the adaptation of the computational procedures to more than a coreful of data is probably simplest for the basic correlation equation.

The basic equation to be evaluated is

$$\hat{R}_{xyr} = \frac{1}{N-r} \sum_{i=1}^{N-r} x_i y_{i+r}, \qquad r = -m, \ldots, -1, 0, 1, \ldots, m, \quad (5.1)$$

where

$$\hat{R}_{xyr} \equiv \hat{R}_{xy}(r\Delta t),$$

$$x_i \equiv x(i\Delta t),$$

$$y_i \equiv y(i\Delta t),$$

and usually,

$$\Delta\tau = \Delta t.$$

It is this equation that will form the basis of all correlation computations.

Two different types of time series data arise that are to be processed by a digital computer. Some processes are discrete by their very nature, say for example, the daily closing prices of a given stock. On the other hand, some data arise naturally as a continuous record such as the continuously recorded output of an accelerometer on the skin of a missile structure, which is intended to give a measure of the vibration at that point. The continuous record is digitized for digital analysis by an analog-to-digital conversion procedure. In either case, an important observation to make is that only a finite number of bits in a binary digital computer are required to represent any given individual data point. In actual practice, many analog-to-digital converters present their output at a precision of 8 to 10 bits (including the sign bit). For almost all applications, the recording instruments are no more accurate than one part in 256 to 1024 and hence, that quantization is fine enough. The dynamic range in terms of decibels is from about 42 dB for 8 bits to 54 dB for 10 bits. In some legitimate applications, greater dynamic ranges can exist in the data and transducers, but not usually.

As it turns out, for many time series of interest, much coarser quantization is often acceptable. In the extreme case, one can quantize to a single bit. That is, if the value of a signal is larger than or equal to zero, set the bit equal to zero; if the value of the signal is less than zero, set the bit equal to one. In simpler terms, the sign bit of the data point is the only information retained. Applications of this concept will be discussed in Section 5.5. In mathematical terms, if $x(t)$ is the original signal, then the one-bit quantized (hard-clipped) signal $y(t)$ is defined by the relation

$$y(t) = \operatorname{sgn} x(t) = \begin{cases} 1 & x(t) \geqslant 0 \\ -1 & x(t) < 0. \end{cases} \quad (5.2)$$

A very important observation is that when 8 to 10 bits or less are employed for the level of quantization, the number of unique values that a data point can have is not too great, namely $2^8 = 256$ to $2^{10} = 1024$ different values. By proper development of this concept, certain time-saving procedures can be implemented and are discussed in Sections 5.3, 5.4, 5.5, and 5.6.

In the following subsections, different procedures for computing the correlation function are discussed. First, the basic direct method of implementing Eq. (5.1) is described along with an indication of the number of operations necessary for its implementation. The other methods are

1. One that expands the summation and collects like terms so that they may be factored out to eliminate multiplications;
2. A method that involves relatively coarse quantization so that advantage may be taken of one of the special conversion instructions of the IBM 7090 class of machines which in effect perform a table look-up;
3. So-called hard clipping methods which are based on the one-bit quantization;
4. A "quarter-square" method that takes advantage of the fact that a cross product can be expanded as a linear combination of squares, which can then be efficiently obtained by table look-up procedures, and finally;
5. A direct Fourier transformation of the original time history to obtain the power spectrum and then an inverse Fourier transform to obtain the correlation function.

These methods will now be described.

5.2 Basic Correlation Functions for Computational Method

The fundamental correlation computational procedures consist of implementing Eq. (5.1) directly. Since $R_{xy}(\tau) = R_{yx}(-\tau)$, the two parts of the cross-correlation function can be computed for positive time delays. That is,

$$\hat{R}_{xyr} = \frac{1}{N-r} \sum_{i=1}^{N-r} x_i y_{i+r}, \tag{5.3a}$$

$$\hat{R}_{yxr} = \frac{1}{N-r} \sum_{i=1}^{N-r} y_i x_{i+r}, \qquad r = 0, 1, \ldots, m. \tag{5.3b}$$

For an autocorrelation function, $R_x(\tau) = R_x(-\tau)$. Hence, only Eq. (5.3a) need be evaluated in this case.

$$\hat{R}_{xr} = \frac{1}{N-r} \sum_{i=1}^{N-r} x_i x_{i+r}, \qquad r = 0, 1, \ldots, m. \tag{5.4}$$

The number of multiply-add operations required for the computation of $\hat{R}_{xy}(r)$ is roughly $(m + 1)(N - m/2) \approx mN - (m^2/2)$. This essentially defines the required computation time. For the IBM 7094, the innermost multiply-add loop in floating-point arithmetic takes roughly 12 machine cycles at 1.4 μsec per cycle. Thus, for a typical program operating on $N = 10\,000$ data points for $m = 1000$ lags, the computation time could be estimated as

$$T_a = nc\,T_1$$

$$= \left[mN - \frac{m^2}{2}\right]\left[12\right]\left[1.4 \times 10^{-6}\right]$$

$$= \left[10^4 \times 10^3 - \frac{10^6}{2}\right]\left[12\right]\left[1.4 \times 10^{-6}\right] = 160 \text{ sec},$$

where

T_a = the total computation time,

n = the number of multiply-add operations,

c = the number of machine cycles per multiply-add operation,

T_1 = the time for one machine cycle.

If the sampling interval is $\Delta t = 0.1$ millisecond, then the parameters in the example correspond to an analysis "bandwidth" of 10 Hz, performed in the frequency band 0 to 5000 Hz. This basic correlation computation method requires some adaptation if it is desired to compute correlation functions for record lengths that exceed the memory capacity of the computer.

An autocorrelation function can be computed as data are read into the digital computer to eliminate any record length restrictions. To illustrate this, suppose an autocorrelation function with $m = 5$ lag values is to be computed. All lag values can be determined by reading only $2(m+1) = 12$ data points at a time into the computer storage. The process is as follows:

1. Read in data points $x_1, \ldots, x_6$,
2. Compute and accumulate products;

R_{x0}	R_{x1}	R_{x2}	R_{x3}	R_{x4}	R_{x5}
x_1^2	x_1x_2	x_1x_3	x_1x_4	x_1x_5	x_1x_6
$+x_2^2$	$+x_2x_3$	$+x_2x_4$	$+x_2x_5$	$+x_2x_6$	
$+x_3^2$	$+x_3x_4$	$+x_3x_5$	$+x_3x_6$		
$+x_4^2$	$+x_4x_5$	$+x_4x_6$			
$+x_5^2$	$+x_5x_6$				
$+x_6^2$.					

The above products are summed by column to obtain partial sums that will eventually make up points on the correlation function.

3. Read in data points $x_7, \ldots, x_{12}$, save $x_1, \ldots, x_6$,
4. Compute and accumulate the products;

R_{x0}	R_{x1}	R_{x2}	R_{x3}	R_{x4}	R_{x5}
					x_2x_7
				x_3x_7	$+x_3x_8$
			x_4x_7	$+x_4x_8$	$+x_4x_9$
		x_5x_7	$+x_5x_8$	$+x_5x_9$	$+x_5x_{10}$
	x_6x_7	$+x_6x_8$	$+x_6x_9$	$+x_6x_{10}$	$+x_6x_{11}$
x_7^2	$+x_7x_8$	$+x_7x_9$	$+x_7x_{10}$	$+x_7x_{11}$	$+x_7x_{12}$
$+x_8^2$	$+x_8x_9$	$+x_8x_{10}$	$+x_8x_{11}$	$+x_8x_{12}$	
$+x_9^2$	$+x_9x_{10}$	$+x_9x_{11}$	$+x_9x_{12}$		
$+x_{10}^2$	$+x_{10}x_{11}$	$+x_{10}x_{12}$			
$+x_{11}^2$	$+x_{11}x_{12}$				
$+x_{12}^2$.					

These products are added to the previous partial sums by column.

5. The next set of six data values would be read into the storage space occupied by the first six. By proper programming, the appropriate cross products can be computed and the partial sums accumulated as the process goes along so that when the data have been read, all points on the correlation function will have been determined. This can easily be generalized to handle an arbitrary number of variables if the data are multiplexed. Denote the variables by $x_1(t), x_2(t), \ldots, x_p(t)$. The N digitized values of $x_1(t)$ are denoted by x_{11} $x_{12}, x_{13}, \ldots, x_{1N}$. For the remainder of the variables, the data are then arranged on the digital input tape in the sequence

$$x_{11}, x_{21}, \ldots, x_{p1}; \quad x_{12}, x_{22}, \ldots, x_{p2};$$

$$x_{13}, x_{23}, \ldots, x_{p3}; \quad \ldots x_{1N}, x_{2N}, \ldots, x_{pN}.$$

All auto- and crosscorrelation functions can then be computed in parallel with the required storage being $3p(m+1)$ cells. More than this amount of storage can, of course, be employed. That is merely the minimum requirement.

Performing the input of a subsequent section of data while computations are in progress requires some care in programming. A minimum of three buffer areas must be established in the core storage for an autocorrelation computation to avoid reading new data in on top of data that is still in use. Six input areas are necessary for crosscorrelation computations involving two sequences. The buffering procedure must progress as follows:

1. Read first set of data into Buffer 1;
2. Read second set of data into Buffer 2;
3. Initiate reading of next set of data into Buffer 3;
4. Wait for termination of read operation into Buffer 2;
5. Perform partial product accumulations on data in Buffer 1, overlapping as necessary into Buffer 2 to obtain lagged products;
6. Initiate reading of next set of data into Buffer 1;
7. Wait for termination of read operation into Buffer 3;
8. Perform partial product accumulations on data into Buffer 2, overlapping into Buffer 3 as necessary to obtain lagged products;
9. Initiate reading of next set of data into Buffer 2;
10. Wait for termination of read operation into Buffer 1;
11. Perform partial product accumulation on data in Buffer 3, overlapping into Buffer 1 as necessary to obtain lagged products.

The computation in Step 11 is slightly more involved than in Steps 5 and 8 if the buffer areas are contiguous and in sequence. To obtain the core memory address for the data in Buffer 1, a factor of $3N_b$, where N_b = buffer size, must be subtracted from the index for the lagged data obtained from Buffer 1. At the first pass through Step 11, lagged products are being accumulated:

$$P_r = \sum_{i=2N_b+1}^{3N_b} x_i\, x_{i+r}, \qquad r = 0, 1, \ldots, m. \tag{5.5}$$

However, for $i + r > 3N_b$, the data is located at $i + r - 3N_b$. Thus, Eq. (5.5) must be broken into two parts

$$\left.\begin{aligned} P_{1r} &= \sum_{i=2N_b+1}^{3N_b-r} x_i\, x_{i+r} \qquad (5.6a)\\ P_{2r} &= \sum_{i=3N_b-r+1}^{3N_b} x_i\, x_{i+r-3N_b} \qquad (5.6b) \end{aligned}\right\} \quad r = 0, 1, \ldots, m.$$

5.3 Factoring of Common Terms

A method employed in connection with seismic data analysis is described by Simpson [23]. This method uses a data scan to factor out common terms that

appear in the summation for the evaluation of the correlation function. This procedure can only be efficiently accomplished if a relatively small number of data quantization levels, as compared to the total number of data points, are possible so that relatively large numbers of common factors exist on the average. For example, suppose a sequence of data is obtained, as indicated below.

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
x_i:	3	−2	1	4	2	2	3	4	1	0	1	−1	3	2	−4	0	3	4	1	−2

Equation (5.1) can be expanded in the form

$$\begin{aligned}\hat{R}_{xyr} = {} & 1 \cdot [x_{3+r} + x_{9+r} + x_{11+r} + x_{19+r} - x_{12+r}] \\ & +2 \cdot [x_{5+r} + x_{6+r} + x_{14+r} - (x_{2+r} + x_{20+r})] \\ & +3 \cdot [x_{1+r} + x_{7+r} + x_{13+r} + x_{17+r}] \\ & +4 \cdot [x_{4+r} + x_{8+r} + x_{18+r} - x_{15+r}]. \end{aligned} \tag{5.7}$$

Note that when Eq. (5.1) has been rewritten in this form, only four instead of 20 multiplications are necessary. Because of the reduction in the number of multiplications, the evaluation of Eq. (5.7) can be performed considerably faster than that of Eq. (5.1). This is where the potential advantage lies.

Note that Eq. (5.7) is a function of the specific data being used. Therefore, the particular record of data being analyzed must be examined in order to generate the computational form given by Eq. (5.7). A complicated "program-writing program" is required; that is, a program that will generate a computational program of the form of Eq. (5.7) for each given input data record.

Table 5.1. Speed Advantage Ratio (SA) of Computing Time of Normal Method (N) to Factoring Method

SA	N
3.6	500
5.7	1000
8.1	2000
10.9	5000
12.4	10,000

A correlation function program involved in this procedure was written for the data analysis procedures described by Simpson [23]. A quantization of 100 levels (between 6 and 7 bits) was assumed and empirical speed advantage factors were determined, which are a function of the length of the data, as indicated in Table 5.1.

Hence, for the particular data involved in this application, a considerable saving in time resulted from this highly specialized computational method. Considerable programming effort is required to develop the necessary program-writing program as opposed to merely coding the direct computational method. This is a disadvantage that has to be balanced against the computational time savings.

5.4 Eight-Level Quantization Method

Personnel at the Jet Propulsion Laboratory have programmed a method based on a 4-bit quantization level. That is, plus or minus eight levels of quantization are allowed, which corresponds to roughly plus or minus one decimal digit. The advantage of this coarse quantization is that if a few enough number of levels is used, then table look-ups can be used rather than multiplication operations to determine the products between a pair of data values. Of course, when the number of potential products between two arbitrary data values is too large, then it is likely that the core storage is not large enough to contain the necessary table and this procedure cannot be used. Many of the fast correlation computation methods are based on being able to employ a table look-up procedure instead of direct multiplication. As will be shown in the next subsection, data can be quantized all the way to only one bit and still reasonable results can be obtained by a final functional adjustment of the results.

The Jet Propulsion Laboratory Computation Center used an IBM 7094 computer at the time this program was developed. A special instruction on the IBM 7094 provides an automatic table look-up, which may be used for performing the multiplications automatically. The particular instruction used is one of the conversion instructions (CAQ). By properly constructing the multiplication table of possible products between two data points and by properly placing the two sets of data to be multiplied, one of these conversion instructions can be used to perform the table look-up to obtain the desired products between two sets of data. Similar procedures are used for the conversion instructions to convert from binary to binary-coded decimal format. The data must be preprocessed in order to properly pack several data points into one computer word. The conversion instruction essentially adds in sequence the six, 6-bit characters in a computer word to some predetermined value and then accumulates the contents of this location in the accumulator. Hence, six pairs of data points can be multiplied in one instruction, which is accomplished much faster than regular multiplication operations.

The results of this procedure, based on both experimental and analytical results, indicate that very little precision is lost because of coarse quantization. That is to be intuitively expected since the correlation function is an averaged result, and results with more precision than the original data are often not of any particular interest. But when many numbers of a given accuracy are added, the effect is to gain significant digits of precision. That is, the resulting mean values contain more accuracy than the original data. The additional accuracy may or

may not be of value. One disadvantage encountered in employing such a procedure, and, as a matter of fact, the reason many engineers do not like the results of the 4-bit quantization method, is that the dynamic range is less. There is a fundamental quantization noise floor, which increases in proportion to the minimum level of quantization.

The 8-level method used at the Jet Propulsion Laboratory has an inner computational loop requiring 24 machine cycles to obtain the sum of six products. As an estimate, an additional two machine cycles might be required on the average to prepare the data in the necessary format of six data points per computer word. The total is then 26/6 cycles per multiply-add operation. For the previous example of $N = 10{,}000$ and $m = 1000$, the timing is approximately

$$T = \frac{160}{12} \cdot \frac{26}{6} = 57.8 \text{ sec.}$$

5.5 One-Bit Quantization, or Extreme Clipping Method

A relation between the correlation function of the extremely clipped signal Eq. (5.2) and an original Gaussian zero mean process from which it is obtained gives the normalized correlation function (correlation coefficient) $\rho_x(\tau)$ in terms of the correlation coefficient function of the clipped signal $\rho_y(\tau)$ by the formula

$$\rho_x(\tau) = \sin\left[\frac{\pi}{2}\rho_y(\tau)\right], \tag{5.8}$$

where $\rho_x(\tau)$ and $\rho_y(\tau)$ are defined as

$$\rho_x(\tau) = \frac{R_x(\tau)}{R_x(0)}; \quad \rho_y(\tau) = \frac{R_y(\tau)}{R_y(0)}. \tag{5.9}$$

See Weinreb [24] for a derivation of this relation and Hinich [25] for theoretical work on the spectrum estimation question. Clearly, when the one-bit quantization is performed, the multiplications that need be accomplished become trivial. Various ways exist for taking advantage of this trivial multiplication. One method employed is to again use one of the IBM 7094 conversion instructions. In this case the data are packed, 36 data points per computer word, by an initial examination of the input record. Then by properly generating the table look-up procedure and by employing the appropriate table, 36 products can in effect be looked up with one instruction by using a conversion operation. Six sets of products are obtained at a time since the conversion instruction operates on 6-bit character sets.

Procedures other than the use of the conversion instruction might prove to be just as efficient for evaluating a sum of one-bit products. Simple counters might be set up in which a point, say x_i, is selected. Then all the products of x_i with each data point y_{j+r}, $j = 1, \ldots, N - r$, could be determined by a simple examination

of the data. The problem is not one that can be solved in any general manner because it depends on the characteristics of the particular computer being used for the problem at hand.

There are certain problems connected with the use of the clipping methods for correlation computations. One problem is that requirements for increased record length are traded for computational speed, assuming a constant statistical accuracy is desired. Weinreb [24] shows that the variance of an autocorrelation function estimate is increased by a maximum factor of roughly $\pi^2/4 \approx 2.5$. This variance is proportional to T, so if the variance is to be kept constant, record lengths are required that are two and a half times as long as those needed if all the information available in the data is to be used. As further noted by Weinreb, roughly the same bound holds true for power spectral density estimates. Although these are not precise theoretical relationships, they provide guidelines for evaluating the tradeoff between accuracy and computational time when employing correlation function computations of the clipping type. In the Electrical Engineering Department of the University of Arizona, experiments have been performed to test the variability of methods employing quantized data. Rarely over a 150-percent increase in the variability of autocorrelation function estimates was experienced. Although the theoretical bound is a maximum increase of 250 percent, it seems to be much less in practice. This is because the derivation of the 250-percent bound assumes uncorrelated data samples. In a typical time series, there is considerable correlation between data points, which reduces the observed variability below the 250-percent maximum. Hinich [24] demonstrates that and gives further theoretical work, in which a narrow bandwidth (high-correlation) result is worked out to establish a 128-percent value.

Note that certain assumptions are involved in the use of the hard clipping method;

a. Either a Gaussian distribution is assumed, or
b. a sinusoid, or
c. a sinusoid plus Gaussian noise.

Equation (5.8) applies to data with these characteristics. Note that any periodic data with the same period (i.e., the same zero crossings) would give the same clipped correlation function as that of a sine wave whose clipped correlation function is a triangular wave prior to applying Eq. (5.8). Consider, for example, Fig. 5.1 to illustrate this idea. Thus, caution must be used when applying the clipped correlation ideas to various types of data.

Only normalized correlation functions are directly obtained by this method. Hence, the mean square value must be calculated separately for use as a scale factor. Also, the nonlinear effects that distort probability density functions to non-Gaussian shapes will be masked completely when such a computing method is used.

An alternative to employing the clipping idea to correlation function computations is clipping only one of the components of the product being computed. For example, if the crosscorrelation function is to be computed between $x(t)$

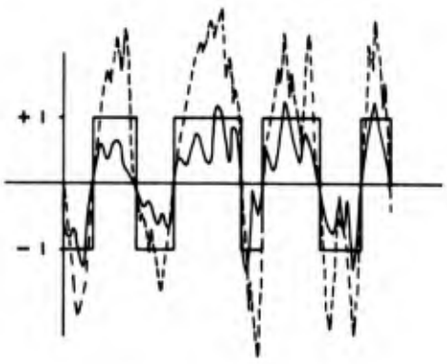


Fig. 5.1. Two time series having an identical extremely clipped version.

and $y(t)$, then clip only $x(t)$. In that case, multiplications are again traded for adds since the sequence y_i is merely added after adjustment for the proper sign of the point x_i that is multiplying it. In that case, the correlation function can be shown to be

$$R_{xy}(\tau) = \sqrt{\frac{\pi}{2}}\, R_{yy}(0)\, R_{xsgny}(\tau), \tag{5.10}$$

where $R_{xsgny}(\tau)$ denotes the correlation function of $x(t)$ with the hard-clipped version of $y(t)$.

Relative time estimates for this method are given in Ref. 26. For the IBM 7044, it is indicated that the computing time can be reduced at a ratio of about 6 to 1. The full clipping method would take proportionately less time. One could expect to perform 36 multiply-add operations in roughly the same amount of machine time. However, more time is required to prepare the data in the proper format.

5.6 Sum of Squares Method

An additional technique that has been suggested (see Schmid [27]) takes advantage of the finite quantization of the data and expresses the product as a sum of squares. The relation employed here expresses the fact that a cross product may be expressed as a linear combination of squares by the following relation:

$$xy = \frac{1}{2}\,[(x+y)^2 - x^2 - y^2]. \tag{5.11}$$

This is very similar to the fairly well-known "quarter-square" method used in construction of analog multipliers. (See Ref. 28, Section 3.1.2.) The quarter-square multipliers are based on the relation

$$xy = \frac{1}{4}\,[(x+y)^2 - (x-y)^2]. \tag{5.12}$$

Equation (5.11) may be modified into a form involving the correlation function immediately:

$$\hat{R}_{xyr} = \frac{1}{2N}\left[\sum_{i=1}^{N-r}(x_i + y_{i+r})^2 - \sum_{i=1}^{N-r} x_i^2 - \sum_{i=1}^{N-r} y_{i+r}^2\right]. \tag{5.13}$$

The key point in using such an indirect relation is that products may now be obtained by table look-up procedures. Tables of squares of two sets of data take up much less storage space than tables of cross products for the same number of data points. For example, say 10-bit quantization is being used so that a table of size 2^{20} cells would be required if all possible cross products were to be stored. The same table of squares of the sum of the data points would only take up 2^{11} cells. 2^{11} might be a very reasonable table size, whereas 2^{20} is not. The second and third sums of squares on the right-hand side of Eq. (5.13) are evaluated by simple recursion relations. A separate initial pass through the data is required to evaluate the sums of squares of all the data. Then the partial sums of squares are obtained from

$$\sum_{i=1}^{N-r} x_i^2 = \sum_{i=1}^{N-r+1} x_i^2 - x_{N-r+1}^2, \tag{5.14a}$$

$$\sum_{i=1}^{N-r} y_{i+r}^2 = \sum_{i=1}^{N-r+1} y_{i+r-1}^2 - y_r^2, \quad r = 0, 1, 2, \ldots, m. \tag{5.14b}$$

Since only one pass through the data is required initially plus one subtraction operation for each correlation function point, the amount of time required is negligible compared with that of the overall correlation function evaluation.

The other portion of Eq. (5.13) requires a more complicated method of evaluation. One possible method of programming for the IBM 7090 is given in Ref. 27. This method requires one pass through the data to set it up in a special tabular format. Then, with the aid of a table of squares, the quantity

$$\sum_{i=1}^{N-r}(x_i + y_{i+r})^2$$

can be generated in a computational loop requiring six machine cycles. The time required is about half that for a straightforward evaluation of the direct sum of

products. For this method, the data must be in fixed-point arithmetic. When the amount of data is too large to fit into the core memory simultaneously, more complicated approaches are required to apply this technique. In view of the potential speed of FFT approaches to correlation function evaluation, it is unlikely that there would be a need to apply this technique except in very special circumstances. In the case of a small machine not having hardware multiplication and division functions, this might be a valuable approach.

5.7 Correlation Functions Via Fast Fourier Transforms

The auto- or crosscorrelation function can be obtained from the power or cross spectral density function. The Wiener-Khinchin relations are applied for this approach. Since spectra can be obtained from Fourier transforms of time histories, the FFT can be applied to obtain correlation functions. Even though this appears to be a roundabout method, it proves to be from 5 to 100 times faster, depending on the maximum lag value desired. One can always obtain N lags of the correlation function nearly as fast as m lags even though m is considerably smaller than N. Considerable detail is given in Ref. 22 concerning this approach.

Basic Method

The basic procedure is as follows:

1. Compute the raw transform X_k of time series x_i; $i, k = 0, 1, \ldots, N-1$;
2. Compute "raw" spectrum $\hat{S}_{xk} = (\Delta t/N)\,|X_k|^2$;
3. Compute the inverse FFT to obtain the autocorrelation function

$$\hat{R}_{xr} = \mathcal{F}^{-1}[\hat{S}_{xk}].$$

Considerable amplification is necessary to illustrate the problems with this procedure. Some explanation is given here that applies to spectrum computations discussed in Section 6.3. Of course, since correlation functions and spectra are Fourier transform pairs, related considerations apply to each function. Thus the discussion in this section is highly correlated with the comments in Section 6.3.

Certain modifications that are not obvious must be made to this approach. It is shown in Ref. 22 that the usual correlation function is not obtained after Step 3 above, but rather a "circular" autocorrelation function $\hat{R}^c_{xr}$ defined by the relation

$$\hat{R}^c_{xr} = \frac{N-r}{N}\,[\hat{R}_{xr} + \hat{R}_{x(N-r)}], \tag{5.15}$$

which is illustrated in Fig. 5.2.

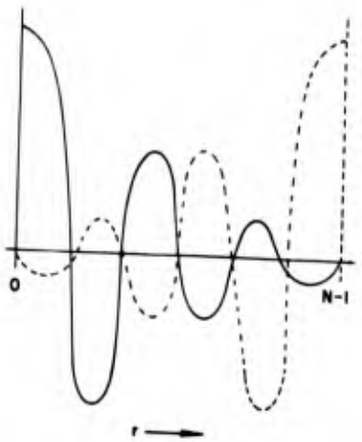


Fig. 5.2. Two parts of correlation function obtained if zeroes are not added.

The effect on the correlation function of adding zeroes to the data is to spread apart the two portions of the circular correlation function. If N zeroes are attached, the two pieces spread as illustrated in Fig. 5.3.

The preceding computational sequence can easily be modified to obtain the noncircular correlation function. If the length of the original sequence of data is a power of 2, $N = 2^p$, then N zeroes would be added to obtain all N lags. If the sequence length is not a power of 2, then the sequence could be filled with N_z zeroes until the first power of 2 was reached. The number of unbiased lag values obtained would be N_z in this case. If more were necessary, the sequence length would have to be doubled by augmenting with 2^p additional zeroes. The modified computational sequence is

1. Augment original time series x_i', $i = 0, 1, \ldots, N-1$ with N zeroes to obtain sequence x_i, $i = 0, 1, \ldots, 2N-1$;
2. Compute the $2N$-point FFT X_k, $k = 0, 1, \ldots, 2N-1$;
3. Compute the "raw" spectrum $\tilde{S}_{xk} = (\Delta t/N)\,|X_k|^2$, for $k = 0, 1, \ldots, 2N-1$;
4. Compute the inverse FFT of $\tilde{S}_{xk}$ and multiply by $N/(N-r)$ to obtain the correct divisor $\hat{R}_{xr} = N/(N-r)\,\mathcal{F}^{-1}\,[\tilde{S}_{xk}]$, $r = 0, 1, \ldots, 2N-1$;
5. Discard the last half of $\hat{R}_{xr}$ to obtain N correlation function points.

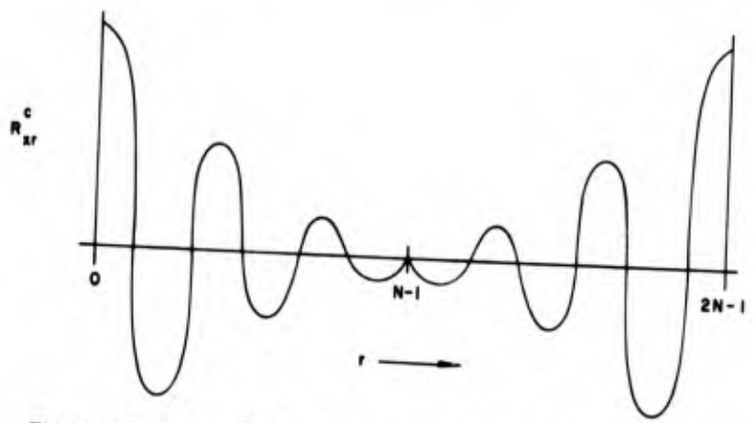


Fig. 5.3. Effect on circular correlation function when N zeroes are added.

For crosscorrelation functions, the same approach applies. In this case, Eq. (5.15) generalizes to

$$R^c_{xyr} = \frac{N-r}{N}\,[R_{xyr} + R_{yx(N-r)}]. \tag{5.16}$$

Hence, when zeroes are used to augment the sequence, the R_{yxr} portion of the crosscorrelation function is obtained except that $R_{yxr} = 0$ is included and $R_{yx0} = R_{xy0}$ does not appear. If the two sequences are arranged as follows

$$x_0, x_1, x_2, \ldots, x_{N-1}, 0_N, 0_{N+1}, \ldots, 0_{2N-1},$$

$$y_0, y_1, y_2, \ldots, y_{N-1}, 0_N, 0_{N+1}, \ldots, 0_{2N-1},$$

then the crosscorrelation result obtained via FFT's is

$$R_{yxN}, R_{yx(N-1)}, \ldots, R_{yx1}, R_{xy0}, R_{xy1}, \ldots, R_{xy(N-1)}.$$

If the arrangement of zeroes is modified so that the x sequence has trailing zeroes and the y sequence has leading zeroes, then the entire crosscorrelation function from R_{yxN} to $R_{xy(N-1)}$ is obtained. If the sequence is arranged as

$$x_0, x_1, x_2, \ldots, x_{N-1}, 0_N, 0_{N+1}, \ldots, 0_{2N-1},$$

$$0_0, 0_1, 0_2, \ldots, 0_{N-1}, y_0, y_1, y_2, \ldots, y_{N-1},$$

then the crosscorrelation result obtained is

$$R_{xy0}\, R_{xy1}, \ldots, R_{xy(N-1)}, R_{yx(N)}, R_{yx(N-1)}, \ldots, R_{yx1}.$$

Since $R_{yxr} = R_{xy(N-r)}$, the entire crosscorrelation function is obtained in proper sequence from $r = -N$ to $r = (N-1)$.

The crosscorrelation function is obtained from the cross spectral density function. Hence, two FFT's are involved rather than one. By applying Eqs. (4.50) and (4.51), two FFT's of real-valued time histories may be obtained simultaneously. The computational steps for the crosscorrelation are as follows:

1. Store x'_i in the real part and y'_i in the imaginary part, $z'_i = x'_i + jy'_i$; $i = 0, 1, \ldots, N-1$.
2. Augment both the real and imaginary parts with N zeroes to obtain $z_i = x_i + jy_i$; $i = 0, 1, \ldots, 2N-1$.
3. Compute the $2N$-point FFT Z_k; $k = 0, 1, \ldots, 2N-1$.
4. Compute X_k and Y_k from

$$X_k = \frac{Z_k + Z^*_{(N-k)}}{2}, \qquad k = 0, 1, \ldots, N-1$$

$$Y_k = \frac{Z_k - Z^*_{(N-k)}}{2j}.$$

5. Compute the raw cross spectrum

$$\tilde{S}_{xyk} = \frac{\Delta t}{N} X_k^* Y_k.$$

6. Compute the inverse FFT of $\tilde{S}_{xyk}$ and multiply by $N/(N-r)$ to obtain the correct divisor

$$\hat{R}_{xyr} = \frac{N}{N-r} \mathcal{F}^{-1} [\tilde{S}_{xyk}].$$

More complicated procedures may be employed and may be desirable depending on the final results wanted and the computer storage limitations. For example, in the previous computational sequence, if one autocorrelation function of length N were to be obtained, $4N$ words of storage in core memory would be required. The original sequence occupies N words, N zeroes are attached, and $2N$ words are then required for the imaginary part. If a cross-correlation is obtained $2N$ useful data values could be obtained since both the positive and negative delay portions would be available.

Only $2N$ storage words are necessary if the following procedure is used. First, note the two relations used in Section 4 to

1. compute two real transforms simultaneously and
2. compute a double length transform in two steps with use of a single length transform.

When a transform of a sequence of length N consisting of $N/2$ data values and $N/2$ zeroes is desired, this may be split into two sequences, each of length $N/2$. Then, the two-at-a-time procedure can be employed to obtain the two transforms of length $N/2$. These two are finally combined to obtain the single transform of N data points. The data must be arranged properly since the double length formulas (4.57a and b) require that the data be split into the even- and odd-indexed portions. Let the augmented sequence be

$$x_0, x_1, x_2, \ldots, x_{\frac{N}{2}-1}, \overbrace{0, \ldots, 0}^{N/2 \text{ times}}.$$

The arrangement necessary for the use of an $N/2$-point transform is shown in Fig. 5.4.

Let z_i be the complex sequence where

$$x_{ei} = \text{Re } z_i = \begin{cases} x_{2i} & i = 0, 1, \frac{N}{2} - 1 \\ 0 & i = \frac{N}{2}, \ldots, N, \end{cases} \tag{5.18a}$$

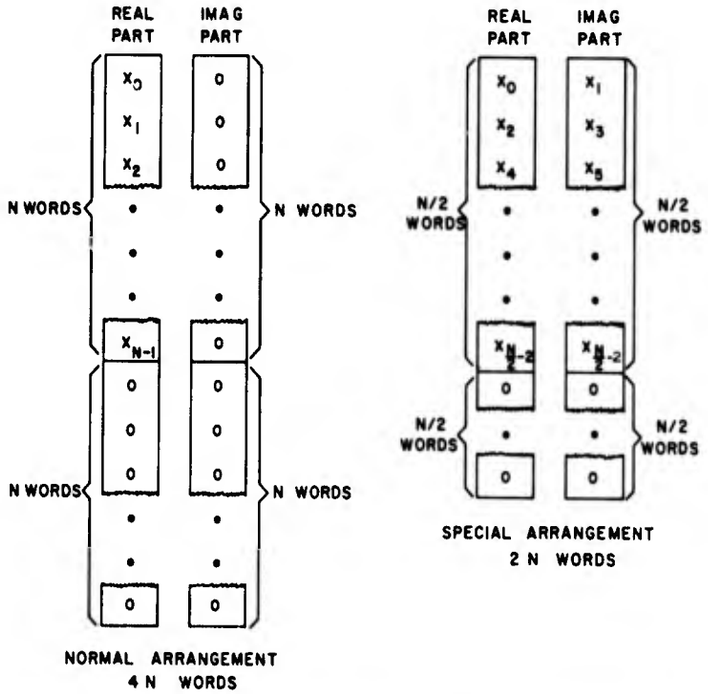


Fig 5.4. Normal and special arrangements for computing the single correlation function of an N-point sequence.

$$x_{oi} = \operatorname{Im} z_i = \begin{cases} x_{2i+1} & i = 0, 1, \ldots, \dfrac{N}{2} - 1 \\ 0 & i = \dfrac{N}{2}, \ldots, N. \end{cases} \tag{5.18b}$$

To maintain everything conveniently in storage, X_{ek} and X_{ok} can be arranged as in Fig. 5.5. If that is done, temporary extra storage locations are minimized. The Z_k are needed in sequence from each end toward the middle; thus it is necessary to store X_{ok} in the reverse manner indicated.

Now complete the finite, discrete transform of z_i

$$Z_k = \sum_{i=0}^{N-1} z_i W^{ik}, \qquad k = 0, 1, \ldots, N-1.$$

Use Eqs. (4.50) and (4.51) to obtain the individual transforms

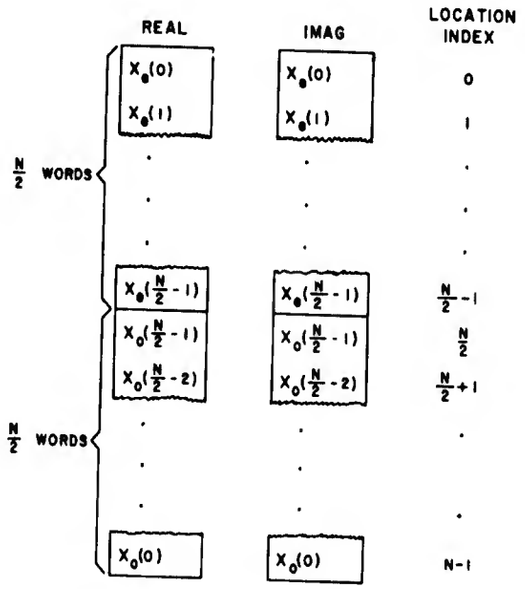


Fig. 5.5. Storage arrangement necessary when splitting out two transforms.

$$X_{ek} = \frac{Z_k + Z^*_{(N-k)}}{2},$$

$$X_{ok} = \frac{Z_k - Z^*_{(N-k)}}{2j}, \qquad k = 0, 1, \ldots, \frac{N}{2} - 1.$$

$$Z_N = Z_0$$

The functions X_{ek} and X_{ok} may be thought of as the first halves of N-point transforms. The entire transform is known since x_{ei} and x_{oi} are real functions and thus their transforms are Hermitian.

The $2N$-point transform can now be obtained from Eqs. (4.57a) and (4.57b)

$$X_k = X_{ek} + X_{ok} W^k, \qquad k = 0, 1, \ldots, \frac{N}{2} - 1. \tag{5.19}$$

So far, the first half of the N-point autocorrelation function has been obtained. It is now observed that

$$W^{k+\frac{N}{2}} = W^k W^{N/2} = -W^k,$$

and

$$X\left(k+\frac{N}{2}\right) = X\left(k-N+\frac{N}{2}\right) = X\left(k-\frac{N}{2}\right) = X^*\left(\frac{N}{2}-k\right),$$

where X_k is the transform of a real-valued function. The second half of the autocorrelation is obtained from

$$X\left(k+\frac{N}{2}\right) = X_e^*\left(\frac{N}{2}-k\right) + X_o^*\left(\frac{N}{2}-k\right) W^k, \quad k = 0, 1, \ldots, \frac{N}{2}-1. \tag{5.20}$$

Note that the implementation of these equations requires special procedures so that no additional storage is needed. Equations (5.19) and (5.20) can be performed simultaneously on the data arranged as indicated in Fig. 5.5. Storage is used as indicated in Fig. 5.6.

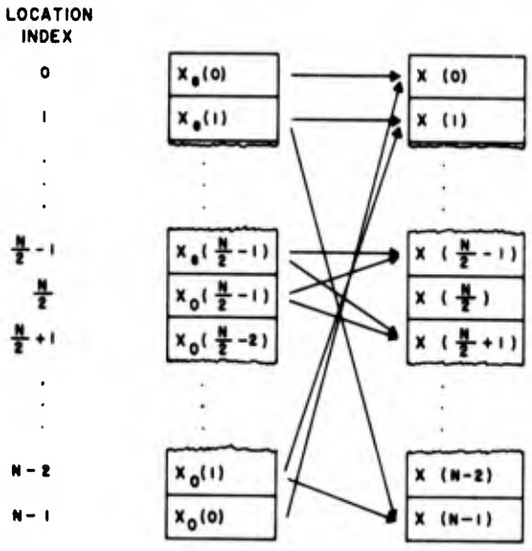


Fig. 5.6. Storage procedure for obtaining a noncircular correlation function.

The arrangement in Fig. 5.6 indicates that the storage is not perfectly symmetrical but lags behind by one for the top half. Thus, the computation of $X_{(N/2)+k}$ follows behind X_k. Absolute values squared can now be computed, and the N-point inverse transform taken to obtain the autocorrelation function. The autocorrelation will consist of only $N/2$ unique points in the usual order since only the even-indexed elements will be obtained. The last half will be a reversed order of the first half. The following discussion will clarify. If the full

$2N$-point transforms were being taken, the two parts of the circular correlation function would be obtained as pictured in Fig. 5.2. Since only an N-point transform is being computed, the even-indexed points are obtained:

Correlation Function Value

$$R_0,\ R_1,\ R_2,\ \ldots,\ R_{(N-2)},\ R_{(N-1)},\ R_N,\ R_{(N-1)},\ \ldots,\ R_1$$

Core Location Index

$$0 \qquad 1 \quad \ldots, \quad \frac{N}{2}-1, \quad \ldots, \quad \frac{N}{2}$$

The odd-indexed points can be obtained by a second transform.

5.8 Normalization of Correlation Functions

In classical statistics, it is common to describe correlation in terms of a normalized quantity $\rho(\tau)$ where

$$|\rho(\tau)| = \left|\frac{C(\tau)}{C(0)}\right| \leqslant 1.$$

The unnormalized quantity $C(\tau)$ is usually termed a covariance function with the tacit assumption that the mean value is zero. In the early development of statistical communication theory, the term correlation function was generally applied to the lagged cross product of two variables without regard to their mean values and without normalization. Special notation is therefore required, and the notation in this document is patterned after that of Ref. 5. The correlation function is defined without restriction on the mean value and without normalization:

$$\hat{R}_{xyk} = \frac{1}{N-r} \sum_{i=0}^{N-r-1} x_i\, y_{i+r}, \qquad r = 0, 1, \ldots, m.$$

The covariance function has the mean value removed

$$\hat{C}_{xyk} = \frac{1}{N-r} \sum_{i=0}^{N-r-1} (x_i - \bar{x})\,(y_{i+r} - \bar{y}). \tag{5.21}$$

In practice, the mean value usually has been subtracted from the data. Finally, the normalized quantity is termed the correlation coefficient function and is defined by

$$\rho_{xyk} = \frac{C_{xyk}}{\sigma_x \sigma_y}, \tag{5.22}$$

where

$$\sigma_x = \sqrt{C_x(0)},$$

and

$$\sigma_y = \sqrt{C_y(0)}.$$

The primary digital computer use of this quantity is as a plotting convenience. Also, in many cases it provides a convenient interpretation of the degree of (linear) correlation between two variables. The use of ρ_{xyk} simplifies a plotted output in that the scale of the function is guaranteed to be between minus one and plus one.

The normalization may be accomplished by an alternative method. If the sample mean $\bar{y}$ and sample variance s_y^2 have been computed, it is often convenient to transform the data to unit variance in addition to zero mean.

$$x_i = \frac{y_i - \bar{y}}{s_y}. \tag{5.23}$$

Then $\bar{x} = 0$ and $s_x^2 = 1$, which can simplify data handling when fixed point rather than floating point computations are employed.

5.9 Computational Time Comparisons

The straightforward implementation of Eq. (5.1) requires Nm (real) multiply-add operations for m lags of an autocorrelation function and $2Nm$ for a cross-correlation function. A fast Fourier transform of an $N = 2^p$ point sequence requires Np complex multiply-add operations, if it is assumed that the necessary complex exponentials have been computed in advance. This can be expedited by doing power of 4 type transforms. If the necessary sines and cosines are to be computed, $N/2$ of each are required, and a total of about 10 real multiply-add operations each. If two transforms are computed simultaneously, it requires an additional $2N$ real multiply-adds if one assumes multiplication by 1/2 is performed.

Obtaining a crosscorrelation requires

Computation	Number of Real Operations
1. One $2N$-point transform	$4 \cdot 2N \cdot 2p$
2. Sines and cosines	$10 \cdot 2N$
3. Split apart $X(k)$ and $Y(k)$	$4 \cdot N$
4. Cross product	N
5. Inverse $2N$-point transform	$4 \cdot 2N \cdot 2p$
TOTAL	$T_0 = (32p + 25)N$

Hence, if $32p + 25 < 2m$, then the FFT route is quickest and all possible $(2N - 1)$ lag values are always obtained. A typical specific example might be $N = 2^{12} = 4096$. Then

$$T_0 = (32 \cdot 12 + 25)\, N = 409\, N \text{ operations.}$$

A typical choice of m is $0.1\, N$, so that the direct route requires about $2 \cdot T_0$ operations.

CHAPTER 6
SPECTRAL DENSITY FUNCTION COMPUTATIONS

6.1 Power Spectra Calculations–Standard Method

There are three basic methods employed in computing power spectral densities. Each of these is based on different but equivalent definitions of the PSD. These are

1. The standard or Blackman-Tukey method based on the Wiener-Khinchin relations. This method involves computing the autocorrelation function of the time series and then taking its Fourier transform.

2. The direct Fourier transform method. The Fourier transform of the original time series is computed, from which the mean absolute value squared is determined.

3. The bandpass filter method. This involves the use of a bank of bandpass filters, one filter for each frequency value of interest. The output of the filter is squared and integrated (averaged). Either analog or digital bandpass filters can be employed.

The equivalence of the first two methods was demonstrated by Wiener [3] and Khinchin [29]. Other authors such as Magness [30] have shown the equivalence of all three procedures. It should be remembered that the equivalence exists only under limiting conditions, so that actual PSD's obtained by different techniques may not look exactly alike. PSD's obtained by the same method also may differ because the parameters used in the respective computations are not the same.

Historically, the second method was developed first. It was not widely used on actual data, however, because the required calculations increase in proportion to the square of the number of data points when the basic formula* is used. Rather, it was useful for examining theoretical functions such as sine waves, square waves, etc. For special cases such as these, the Fourier transform may be obtained readily in closed form even with the discrete representation.

The works of Wiener and Khinchin mentioned above showed that the autocorrelation transform method was equivalent to the Fourier transform method. It provided a firm mathematical foundation since transforms of correlation functions will usually exist mathematically where transforms of the time history do not. Bandwidth requirements permitting, the correlation PSD was much cheaper to obtain than the Fourier transform PSD. It should be noted, however, that correlations themselves were extremely difficult to compute with reasonable amounts of computer time until the mid-1950's. Computing power has dropped in price every year because of the successive improvements that have been made

*The formula developed before FFT's.

in computer hardware and software, steadily making correlation PSD calculations less expensive. The credit for developing the correlation PSD into a firmly established, practical, and available technique belongs to Blackman and Tukey [9], whose work is a milestone in the data analysis field. Their contributions will be referred to constantly in this chapter.

Analog mechanizations of the PSD techniques usually have been some form of the third method. This type of machine can be visualized as a bank of bandpass filters. The data to be analyzed is the input to each filter, and the output of each filter is squared and integrated (averaged). When the entire record has been processed in this manner, the machine stops and the PSD is read out as the final output of the integrators. Because of the expense of the filters, many machines use a single filter with an effectively variable center frequency. In this case, the data are fed through the filter many times while the center frequency of the filter is either slowly changed in a continuous fashion or stepped in discrete increments after each time the data pass through.

At one time the correlation PSD was the standard method if a digital computer was to be used, while filtering was the traditional analog technique. Times and equipment change, however. There now exist excellent analog and hybrid instruments for computing the correlation function, and digital bandpass filters suitable for PSD analysis have been derived. Thus, it is possible and practical to compute a correlation PSD on analog* equipment and a filter PSD on a general purpose digital computer.

With the advent of fast Fourier transform algorithms, PSD calculations using the second method have become practical. While not all problems inherent in this method have been solved, the procedure has been implemented both on general purpose, digital computers and in analog equipment. On the surface, the high speed of the FFT approach might seem to render all others obsolete. However, this is not always the case. For example, many special purpose high-speed, multiply-add units that permit very rapid correlation computations have been developed for digital computers.

All three methods have the same or related problems. These tend to fall into two distinct areas.

The first problem is that of leakage. The "true" PSD is seen through a "window." This window transmits unwanted power in addition to that which is desired. This extra power is the leakage. The basic process for obtaining the PSD is modified to reduce the leakage as much as possible.

The second difficulty has the nature of a dilemma: if one uses a wide "bandwidth" of analysis, the statistical reliability of the estimate is good, but the PSD is "smeared." On the other hand, if a narrow window is employed, thus making the view of the PSD sharper, or more highly resolved, the variability of the result

*It should be noted that the term *analog* as used herein really refers to a special purpose machine, that is, one designed mainly to provide PSD's. Internally, many of these devices are digital or pulse processing in nature.

increases. In the limiting case, the variance of the answer is the same size as the data themselves! This problem requires more background to discuss it properly. As it is common to all three of the methods, its discussion is postponed until Section 6.4. Leakage, on the other hand, varies somewhat from method to method and so must be accounted for specifically when implementing each of them. The remainder of this section will be spent on the correlation PSD and the problems peculiar to it.

The previous chapter showed various ways by which the correlation function can be obtained. Therefore, let it be assumed that the correlation function of the random variable x is available. Furthermore, for present purposes, suppose that $R_x(\tau)$, the correlation function, is continuous and defined over the interval $-\infty$ to ∞. The one-sided PSD* of x is

$$G_x(f) = 4 \int_0^\infty R_x(\tau) \cos(2\pi f\tau)\, d\tau$$

$$= 2 \int_{-\infty}^{\infty} R_x(\tau) \cos(2\pi f\tau)\, d\tau$$

$$= 2 \int_{-\infty}^{\infty} R_x(\tau)\, e^{-j2\pi f\tau}\, d\tau, \qquad f \geqslant 0. \tag{6.1}$$

All three of these formulations are equivalent. The complex part of the third form drops out because of the symmetry of the autocorrelation function and the antisymmetry of the imaginary part of the exponential.

It is physically impossible to compute this integral exactly from real data because only a finite rather than an infinite length of the sample correlation function may be obtained. Suppose that the maximum value for τ is T_m. Then the sample PSD takes the form

$$\hat{G}_x(f) = 4 \int_0^{T_m} R_x(\tau) \cos 2\pi f\tau\, d\tau$$

$$= 2 \int_{-\infty}^{\infty} u_{T_m}(\tau) R_x(\tau) \cos 2\pi f\tau\, d\tau. \tag{6.2}$$

In the last expression, $u_{T_m}(\tau)$, the boxcar function, is defined as in Eq. (1.28) of Chapter 1, by

*When the term PSD is used, it is implicit that it is one sided.

$$u_{T_m}(\tau) = \begin{cases} 0 & \tau < -T_m \\ 1 & -T_m \leq \tau \leq T_m \\ 0 & T_m < \tau. \end{cases} \tag{6.3}$$

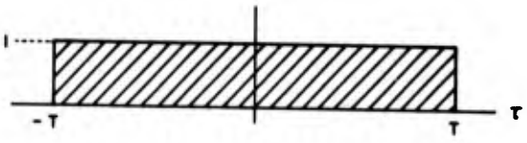


Fig. 6.1a. The boxcar function, $u_T(\tau)$.

The boxcar function is again pictured in Fig. 6.1a. The formulation involving u_{T_m} is clearly equivalent to the limited integration definition of G_x. The reason for employing it is so that the infinite integration range, which is mathematically convenient, may be retained. The Fourier transform of $u_{T_m}(\tau)$ is

$$U_{T_m}(f) = \int_{-\infty}^{\infty} u_{T_m}(\tau)\, e^{-j2\pi f\tau}\, d\tau$$

$$= 2T_m \frac{\sin(2\pi f\tau)}{(2\pi f)}. \tag{6.4}$$

$U_{T_m}(f)$ is shown again in Fig. 6.1b.

$\hat{G}_x(f)$ is the convolution of $G_x(f)$ and $U_{T_m}(f)$. This relationship is best stated using the two-sided form of the PSD:

$$\hat{S}_x(f) = \int_{-\infty}^{\infty} S_x(\phi)\, U_{T_m}(f-\phi)\, d\phi, \qquad f \geq 0. \tag{6.5}$$

S_x is defined by

$$S_x(f) = \int_{-\infty}^{\infty} R_x(\tau) \cos(2\pi f\tau)\, d\tau, \qquad -\infty < f < \infty. \tag{6.6}$$

S_x is symmetric. Its relation to G_x is very simple.

$$G_x(f) = \begin{cases} 2S_x(f) & f \geq 0 \\ 0 & f < 0. \end{cases} \tag{6.7}$$

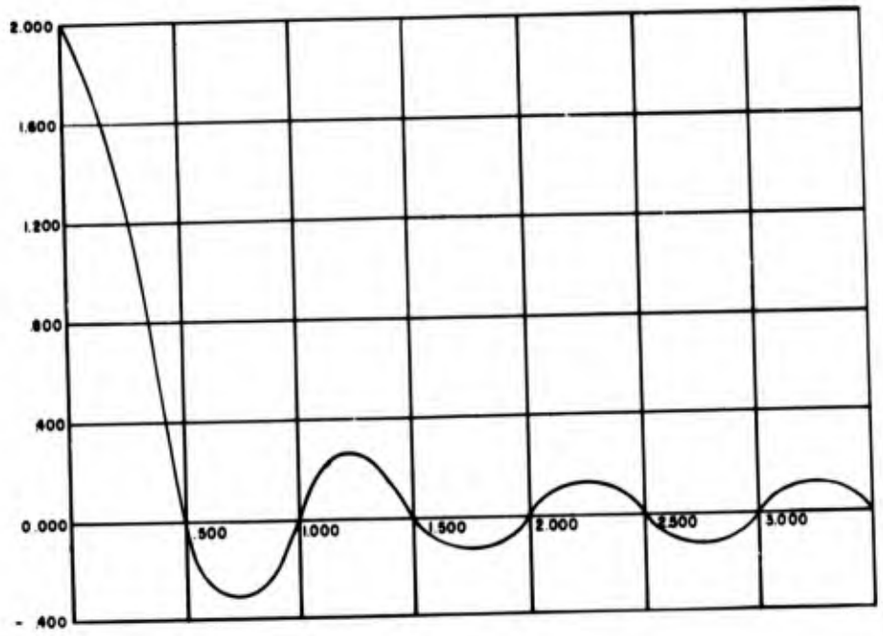


Fig. 6.1b. $U_{T_m}(f)$ versus frequency.

Similar definitions hold for $\hat{S}_x$. Most users prefer the G_x definition, as it eliminates the need for negative frequencies and is more compact. The two-sided definition has to be used in some cases or serious notational problems will arise.

The net effect of this convolution may be seen from the following: suppose that the power at the frequency f_0 is required. Computing $\hat{S}_x(f_0)$ from Eq. (6.5) may be interpreted as moving the main peak (at $f = 0$) of $U_{T_m}(f)$ down to f_0, multiplying the "true" PSD by the translated $U_{Tm}(f)$ function, and then integrating. This is shown in Fig. 6.2.

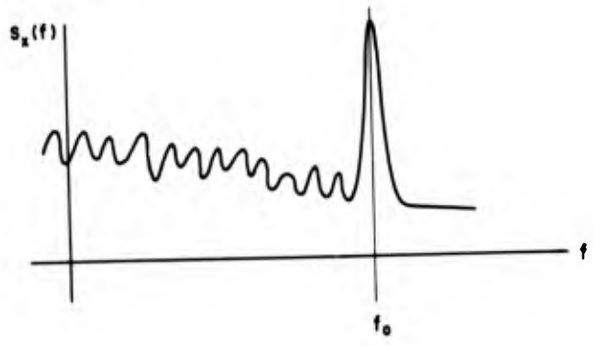


Fig. 6.2a. Theoretical $G_x(f)$

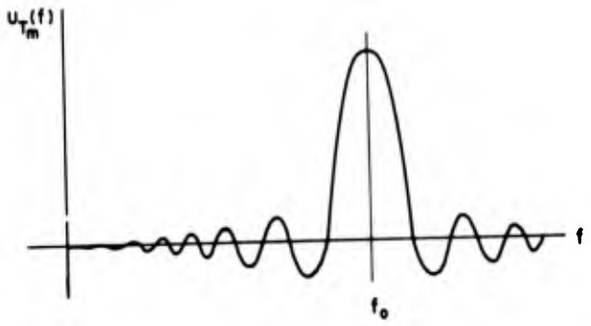


Fig. 6.2b. $U_{T_m}(f)$ translated to $f = f_0$.

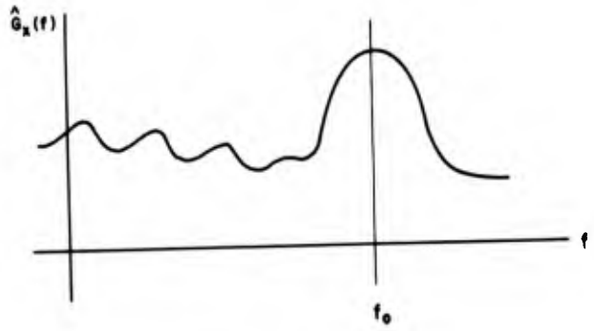


Fig. 6.2c. Resulting $\hat{G}_x(f)$.

As indicated in Fig. 6.2c, the sample PSD, $\hat{G}(f)$, will be smoothed.* The $U_{T_m}(f)$ function is usually referred to as the window. The true PSD is seen through it.

This window, as it stands, has some undesirable characteristics. This fact may readily be seen if $x(t)$ is a sine wave with frequency f_0 and amplitude A. Then R_x and S_x are

$$R_x(\tau) = \frac{A^2}{2} \cos 2\pi f_0 \tau,$$

$$S_x(f) = \frac{A^2}{4} [\delta(f - f_0) + \delta(f + f_0)]. \tag{6.8}$$

*It is understood that while the S_x functions are pictured, the reader can easily make the changeover to G_x.

The sample (two-sided) PSD is then

$$\hat{S}_x(f) = \frac{A^2 T_m}{2} \left\{ \frac{\sin [2\pi T_m(f+f_0)]}{[2\pi T_m(f-f_0)]} + \frac{\sin [2\pi T_m(f+f_0)]}{[2\pi T_m(f+f_0)]} \right\}. \quad (6.9)$$

A good approximation when f_0 is not near zero is that

$$\hat{G}_x(f) = A^2 T_m \frac{\sin [2\pi T_m(f-f_0)]}{[2\pi T_m(f+f_0)]}, \qquad f \geqslant 0. \quad (6.10)$$

Thus, instead of obtaining a single delta function as would be the case for the PSD of a sine wave, the truncation spreads the result considerably. The width of the basic lobe of $G_x(f)$ is $1/T_m$ between zero crossings. This is shown in Fig. 6.3.

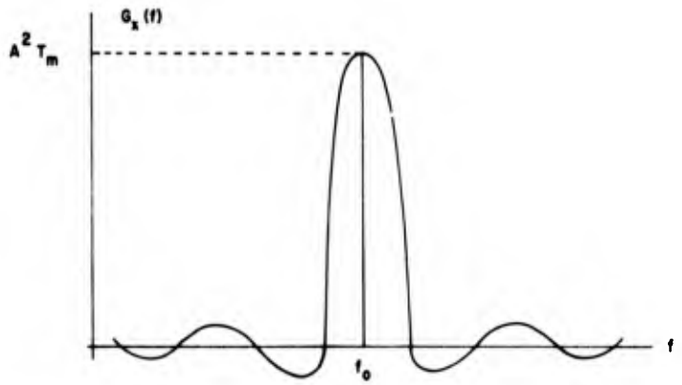


Fig. 6.3. PSD of a sine wave of frequency f_0.

Besides the spreading in the main lobe, there is an infinite number of smaller lobes added to the PSD by the truncation. Not only does the magnitude of these side lobes decrease slowly, but half of them are negative, a most displeasing result since average power is positive by definition. The height of the first two negative lobes is about one-fifth that of the main lobe.

A number of ideas has been suggested to alleviate this problem. The three procedures discussed below are all candidates for a solution to it.* None of the three stands out as being obviously the best. Their good and bad points must be considered by the user, and an engineering judgment made as to which is most suitable for a particular application.

*Several other less important ones are discussed in Appendix B, Section 5 of Ref. 9.

All three do have a common property: They obtain their improvement by modifying the boxcar function in the time domain (or by an equivalent operation in the frequency domain) and by so doing, broaden the principal lobe of the window function in the frequency domain.

The first of these modified spectral windows, or lag windows, is called the Hann* window and takes the form

$$u_{T_m}^{(1)}(\tau) = \begin{cases} 0 & \tau < -T_m \\ \frac{1}{2}\left[1 + \cos\left(\frac{\pi\tau}{T_m}\right)\right] & -T_m \leqslant \tau \leqslant T_m \\ 0 & \tau > T_m. \end{cases} \tag{6.11}$$

The Fourier transform of this, $U_{T_m}^{(1)}(f)$, is

$$\begin{aligned} U_{T_m}^{(1)}(f) &= \frac{1}{2} U_{T_m}(f) + \frac{T_m}{2}\left\{\frac{\sin\left[T_m\left(2\pi f - \frac{\pi}{T_m}\right)\right]}{T_m\left[2\pi f - \frac{\pi}{T_m}\right]} + \frac{\sin\left[T_m\left(2\pi f + \frac{\pi}{T_m}\right)\right]}{T_m\left(2\pi f + \frac{\pi}{T_m}\right)}\right\} \\ &= \frac{1}{2} U_{T_m}(f) + \frac{1}{4} U_{T_m}\left(f - \frac{1}{2T_m}\right) + \frac{1}{4} U_{T_m}\left(f + \frac{1}{2T_m}\right). \end{aligned} \tag{6.12}$$

The Hann window turns out to be the summation of three sin x/x functions! This function is shown in Fig. 6.4. As the lobes are spaced $1/(2T_m)$ Hz apart, this window has the effect of averaging three adjacent lobes, giving the center one twice as much weight as the side ones. Thus, there will be a tendency to greatly decrease the size of the side lobes. The main lobe is reduced to one-half of its previous height, and its width is doubled. The reduction in leakage therefore results in a corresponding widening of the bandwidth of analysis. The distance between the first zero crossings on either side of the main lobe is $2/T_m$ Hz. The distance between the half-power points, however, is $1/T_m$. This value usually is taken to be the resolution bandwidth B of the analysis. Note that T_m refers to the length of autocorrelation used rather than the length of the data record.

The second spectral window is called the Hamming† window and is given by

$$u_{T_m}^{(2)}(\tau) = \begin{cases} 0 & \tau < -T_m \\ 0.54 + 0.46\cos\left(\frac{\pi\tau}{T_m}\right) & -T_m \leqslant \tau \leqslant T_m \\ 0 & \tau > T_m. \end{cases} \tag{6.13}$$

*After Julius von Hann, according to Ref. 9.

†R. W. Hamming, according to Ref. 9.

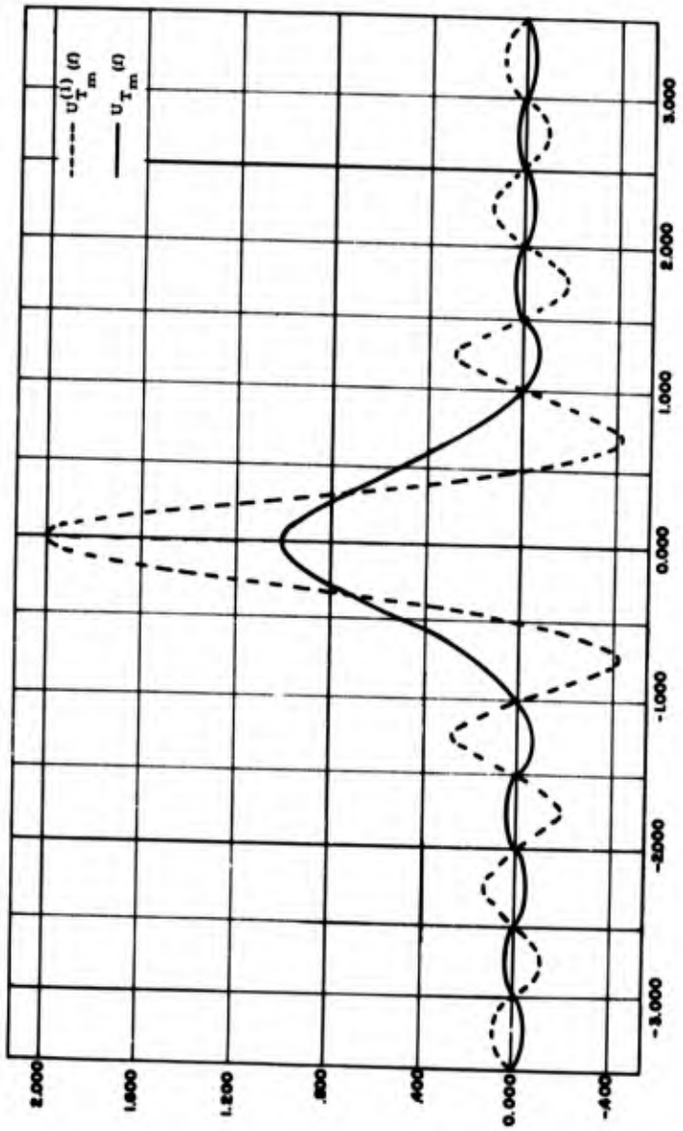


Fig. 6.4. $U_{T_m}(f)$ and $U_{T_m}^{(1)}(f)$ versus frequency.

The Fourier transform of that is

$$U_{T_m}^{(2)}(f) = 0.23\, U_{T_m}\left(f + \frac{1}{2T_m}\right) + 0.54\, U_{T_m}(f) + 0.23\, U_{T_m}\left(f - \frac{1}{2T_m}\right). \tag{6.14}$$

The Hamming window is similar to the Hann window. In detail, there are differences. A comparison of the two windows is made in Fig. 6.5. As indicated in Ref. 9, their differences can be summarized by the following two observations:

1. The height of the maximum side lobe for the Hamming window is approximately one-fifth that of the Hann window,
2. The height of the side lobes of the Hann window tend to drop more rapidly than in those of the Hamming window.

Both of these windows reduce the height of the side lobes after the first to less than 2 percent of the height of the main lobe, and even there the Hann window is almost within the 2-percent bound.

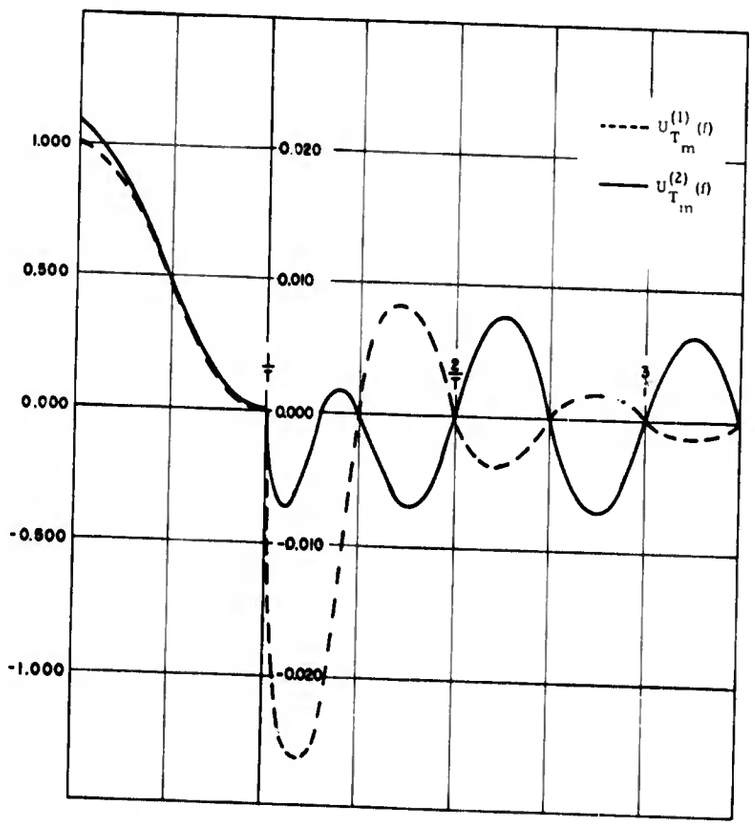


Fig. 6.5. $U_{T_m}^{(1)}(f)$ and $U_{T_m}^{(2)}(f)$ versus frequency.

The third window is one of several attributable to E. Parzen [31]. It takes the form

$$u_{T_m}^{(3)}(\tau) = \begin{cases} 1 - 6\left(\frac{|\tau|}{T_m}\right)^2 \left(1 - \frac{|\tau|}{T_m}\right) & \tau \leqslant T_m/2 \\ 2\left(1 - \frac{|\tau|}{T_m}\right)^3 & |\tau| > T_m/2. \end{cases} \tag{6.15}$$

It is shown in Fig. 6.6a along with the Hann window for comparison. The Fourier transform of the Parzen window is

$$U_{T_m}^{(3)}(f) = \frac{3}{4} T_m \left\{\frac{\sin(\pi f T_m/2)}{(\pi f T_m/2)}\right\}^4. \tag{6.16}$$

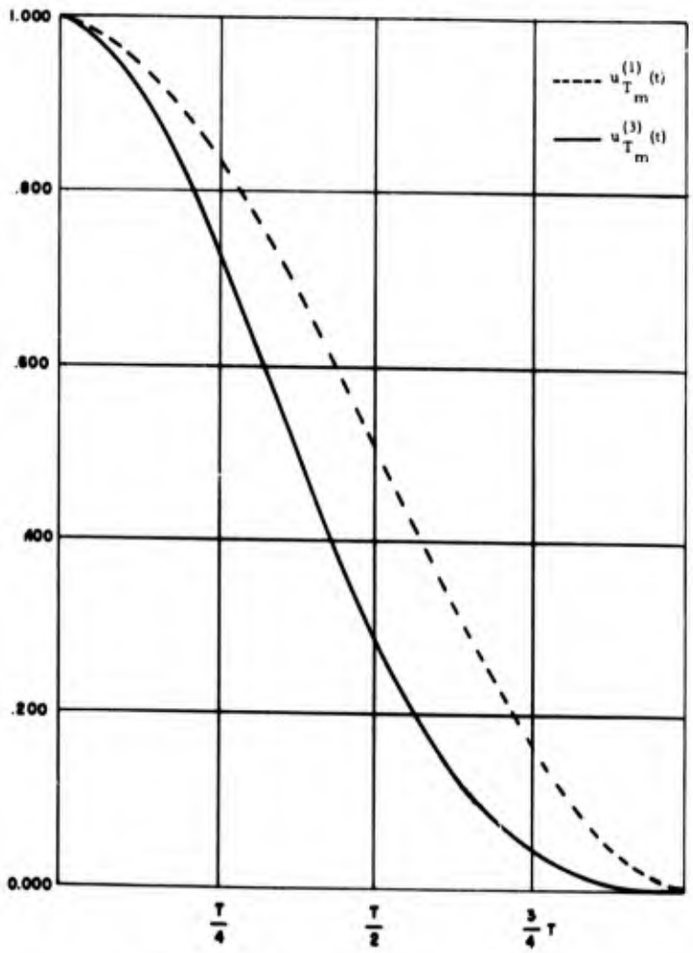


Fig. 6.6a. $u_{T_m}^{(1)}(t)$ and $u_{T_m}^{(3)}(t)$ versus time.

Thus, the Parzen window has a very simple relation to the basic sin x/x window, $U_{T_m}(f)$. The Parzen window is plotted in Fig. 6.6b, again with the Hann window for comparison. The Parzen is wider (about 25 percent) than either the Hann or the Hamming window. One feature of the Parzen window is that, as $U^{(3)}_{T_m}(f)$ is never negative, the PSD generated using it must also be non-negative. This is a great comfort for those who are squeamish about seeing negative power indicated on their PSD calculation printouts.

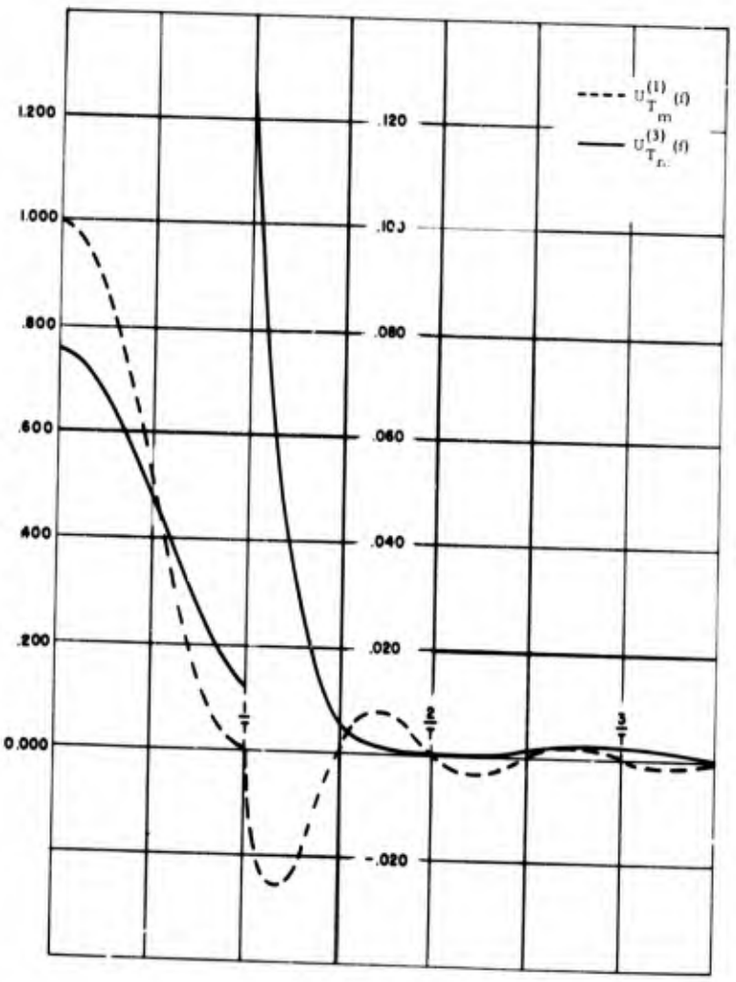


Fig. 6.6b. $U^{(1)}_{T_m}(f)$ and $U^{(3)}_{T_m}(f)$ versus frequency.

Because the main lobe of the Parzen window is wider, the variability of the corresponding spectral estimates is less for the same number of lags, under the assumption of fairly smooth spectra.

All three of these lag windows share an important property. The value of the correlation function at the zero lag is the variance, i.e., the total power. As all three windows are equal to unity at τ equal to zero, the variance is unchanged by any of the three operators, and they are therefore variance preserving operators. This means that the total amount of power shown by the PSD (the area under the PSD) is invariant. That is a desirable characteristic.

Up to this point in the discussion, only continuous correlation functions have been employed. When making the switch to discrete data, it is frequently assumed that the results are identical. As will be seen, the use of the continuous correlation function results in an approximation, though admittedly a very good one.

Suppose that R_{xr} is defined by

$$R_{xr} = \mathrm{E}\,[x_i\, x_{i+r}]. \tag{6.17}$$

The *two-sided* PSD is then

$$S_x(f) = \Delta t \sum_{r=-\infty}^{\infty} R_{xr} \cos(2\pi f r \Delta t). \tag{6.18}$$

Suppose that it is truncated in a manner similar to truncation of the continuous case. That can be accomplished by using only the values of r for $r = 0, 1, \ldots, m$,

$$\hat{S}_x(f) = \Delta t \sum_{r=-m}^{m} R_{xr} \cos(2\pi f r \Delta t). \tag{6.19}$$

As before, Eq. 6.19 could be written using a form of the boxcar function,

$$\hat{S}_x(f) = \Delta t \sum_{r=-\infty}^{\infty} u_{mr}\, R_{xr} \cos(2\pi f r \Delta t). \tag{6.20}$$

The definition given to u_{mr} is

$$u_{mr} = \begin{cases} 1 & |r| \leqslant m \\ 0 & |r| > m. \end{cases} \tag{6.21}$$

As before, $\hat{S}_x(f)$ can be interpreted as the convolution of $S_x(f)$ and $U_m(f)$, the Fourier transform of u_{mr}. Naturally, this implies that $U_m(f)$ should be calculated and examined. Its definition is

$$U_m(f) = \Delta t \sum_{r=-m}^{m} \cos(2\pi f \Delta t).$$

$$= \Delta t \frac{\sin[\pi f(2T_m + \Delta t)]}{\sin(\pi f \Delta t)}. \tag{6.22}$$

T_m here means $m\Delta t$. Recall that, for the continuous case,

$$U_{T_m}(f) = 2T_m \frac{\sin(2\pi f T_m)}{(2\pi f T_m)}. \tag{6.23}$$

The graphs of these two functions look very much alike for the range of values $0 \leqslant f \leqslant 1/2\Delta t$, the principal domain for the digital analysis. In that interval, $\sin(\pi f \Delta t)$ behaves very much like the function $(\pi f \Delta t)$, as shown in Fig. 6.7.

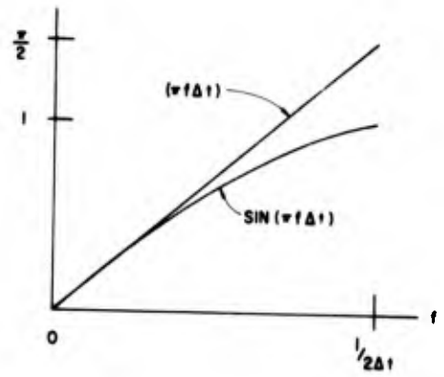


Fig. 6.7. Comparison of $(\pi f \Delta t)$ and $\sin(\pi f \Delta t)$.

If this approximation is made and the Δt within the sine function in the numerator is ignored, then Eq. (6.22) may be easily reduced to Eq. (6.23).

The term $U_m(f)$ is periodic. That would be expected, as $S_x(f)$ in the digital definition is also periodic. The carryover to the formulation for the various windows is similar. The discrete Hann window is

$$u_{mr}^{(1)} = \begin{cases} \frac{1}{2}\left[1 + \cos\frac{\pi r}{m}\right] & |r| \leqslant m \\ 0 & |r| > m. \end{cases} \tag{6.24}$$

Using summation techniques similar to those employed to find Eq. (6.22), we find the Fourier transform of $u_{mr}^{(1)}$ to be

$$U_m^{(1)}(f) = \frac{1}{2} U_m(f) + \frac{1}{4} U_m\left(f - \frac{1}{2m\Delta t}\right) + \frac{1}{4} U_m\left(f + \frac{1}{2m\Delta t}\right). \quad (6.25)$$

The Hamming window is

$$u_{mr}^{(2)} = \begin{cases} 0.54 + 0.46 \cos\left(\frac{\pi r}{m}\right) & |r| < m \\ 0 & |r| > m. \end{cases} \quad (6.26)$$

Its Fourier transform is

$$U_m^{(2)}(f) = 0.54\, U_m(f) + 0.23\, U_m\left(f - \frac{1}{2m\Delta t}\right) + 0.23\, U_m\left(f + \frac{1}{2m\Delta t}\right). \quad (6.27)$$

The results for the Parzen window are not so straightforward. Numerical quadrature indicates that the frequency weighting function has coefficients that are approximately given by the sequence 0.75, 0.493, 0.123, 0.006, 0.000, 0.000, 0.002, 0.000, ...

The question of which one of the three windows is most appropriate is a matter best decided by the user. The authors prefer the Hamming window for most purposes, and no smoothing at all in some special cases such as analyzing data known to be periodic. In any event, the general effect of the windows upon the PSD should be understood by the user, and their action upon the particular type of data being examined should be weighed carefully before a final selection is made.

In the case of the analysis of cross spectral density functions, a combination of windows might be best. The Hamming window can be justified as a good choice for the power spectrum, and the Parzen window, a good choice for the cross spectrum. In many types of data, the PSD is relatively smooth (nearly constant) over frequency intervals on the order of reasonable resolution bandwidths. Thus, the oscillating side lobes of the Hamming window will tend to give alternate signs and roughly equivalent weight to the nearby constant power. The contributions to the PSD estimate from the side lobes will therefore tend to cancel to zero, and leakage will be minimized.

On the other hand, the real and imaginary parts of cross spectra tend to oscillate. Thus, the oscillating side lobes of the Hamming or Hann window might resonate with the cross spectrum itself and cause excessive leakage. In this situation, the relatively constant, always positive, side lobes of the Parzen window will allow the oscillating cross spectrum to be given roughly equal and positive weight so that the average will tend to be zero and leakage will be minimized.

At this point, it is appropriate to sum up the formulas for computing the correlation PSD in the digital case:

1. Assume that x_i is a sequence of N points having a zero mean. (If the mean is not zero, it should be calculated and removed from the data. This step might include the removal of subharmonic terms.)

2. The sample autocorrelation function of x_i is computed for $(m + 1)$ values (the proper choice for m will be discussed in Section 6.4). One method is

$$\hat{R}_{xr} = \frac{1}{N - r} \sum_{i=1}^{N-r} x_i x_{i+r}, \qquad r = 0, \ldots, m. \tag{6.28}$$

3. A lag window is selected. Possible candidates are

i. Hann

$$u_{mr}^{(1)} = \frac{1}{2}\left[1 + \cos\left(\frac{\pi r}{m}\right)\right].$$

ii. Hamming

$$u_{mr}^{(2)} = 0.54 + 0.46 \cos\left(\frac{\pi r}{m}\right).$$

iii. Parzen

$$u_{mr}^{(3)} = \begin{cases} 1 - 6\left(\frac{r}{m}\right)^2 \left(1 - \frac{r}{m}\right) & r \leq m/2, \\ 2\left(1 - \frac{r}{m}\right)^3 & r > m/2. \end{cases} \tag{6.29}$$

One of these is applied to the correlation, resulting in a new correlation, $\tilde{R}_{xr}$:

$$\tilde{R}_{xr} = R_{xr}\, u_{mr}^{(\ell)}, \qquad r = 0, 1, \ldots. \tag{6.30}$$

4. The PSD is calculated for various frequencies using trapezoidal integration,

$$\hat{G}_x(f) = 2\Delta t \left[\tilde{R}_{xo} + 2 \sum_{r=1}^{m-1} \tilde{R}_{xr} \cos(2\pi f r \Delta t) + \tilde{R}_{xm} \cos(2\pi f T)\right]. \tag{6.31}$$

A "standard" set of frequencies is

$$f_k = \frac{k}{2m\Delta t}, \qquad k = 0, 1, \ldots, m. \tag{6.32}$$

This produces $(m+1)$ equally spaced, overlapping PSD estimates. Equation (6.31) can be rewritten as

$$\hat{G}_{xk} = 2\Delta t\left[\tilde{R}_{xo} + 2\sum_{r=1}^{m-1}\tilde{R}_{xr}\cos\left(\frac{\pi rk}{m}\right) + \tilde{R}_{xm}\cos(\pi r)\right],$$

$$k = 0, 1, \ldots, m. \qquad (6.33)$$

Any prewhitening would be done before Step 2, and the corresponding postdarkening accomplished after Step 4.

If the PSD is calculated at the frequencies $k/(2m\Delta t)$, $k = 0, 1, \ldots, m$, then it is possible to change the order of the computations. Step 3 in the above is deleted, and a new one is added at the end. It consists of weighting the PSD in the following manner:

$$\hat{G}_{xk} = \begin{cases} D_0\,\tilde{G}_{x(k-1)} + D_1\,\tilde{G}_{xk} + D_0\,\tilde{G}_{x(k+1)} & k \neq 0, m \\ D_1\,\tilde{G}_{x0} + 2D_0\,\tilde{G}_{x1} & k = 0 \\ D_1\,\tilde{G}_{xm} + 2D_0\,\tilde{G}_{x(m-1)} & k = m. \end{cases} \qquad (6.34)$$

The values for D_0 and D_1 are

$$D_0 = \begin{cases} \frac{1}{2} & \text{Hann} \\ 0.54 & \text{Hamming,} \end{cases}$$

$$D_1 \begin{cases} \frac{1}{4} & \text{Hann} \\ 0.23 & \text{Hamming.} \end{cases} \qquad (6.35)$$

There does not seem to be any real gain in doing the lag window operation in this manner. Compared with the expense of computing the autocorrelation and Fourier transform, the lag window computations require an insignificant amount of time. This formulation is therefore included only because the reader is likely to encounter it in the course of reviewing the computations as done by some organizations. That Eqs. (6.34) and (6.35) are equivalent to the time plane operation can be easily verified by referring to Eqs. (6.12) and (6.14).

Step 4, which is essentially that of computing the Fourier cosine transformation, could be accomplished using the fast Fourier transform techniques described in Chapter 4. On the other hand, if a small number of lag values is used, the complication of the FFT may not be merited. If that is the case, one simple cost-saving procedure is to compute the required cosines recursively. The recursive relation takes the form

$$c_i = hc_{i-1} - c_{i-2},$$

$$h = 2\cos(2\pi\Delta t\Delta f). \tag{6.36}$$

This relation will yield either sines or cosines, depending upon what is used to start the recursion. In particular,

$$c_0 = \begin{cases} 1 & \text{(cosine generation)} \\ 0 & \text{(sine generation)}, \end{cases}$$

$$c_{-1} = \begin{cases} \cos(2\pi\Delta t\Delta f) & \text{(cosine generation)} \\ -\sin(2\pi\Delta t\Delta f) & \text{(sine generation).} \end{cases} \tag{6.37}$$

This recursion is equivalent to Eq. (3.58) with the damping term ζ set to zero.

The last topic of this section is the calculation of cross spectral density. The lag window considerations for the CSD are identical to those for the PSD, so that only the definitions and computational procedures need be discussed. In the continuous case, the crosscorrelation function is

$$R_{xy}(\tau) = E\,[x(t)\,y(t+\tau)]. \tag{6.38}$$

The time average definition is

$$R_{xy}(\tau) = \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} x(t)\,y(t+\tau)\,d\tau. \tag{6.39}$$

The CSD is the Fourier transform of Eq. (6.39)

$$G_{xy}(f) = 2\int_{-\infty}^{\infty} R_{xy}(\tau)\,e^{-j2\pi f\tau}\,d\tau, \qquad f \geqslant 0. \tag{6.40}$$

Another formulation is to write

$$G_{xy}(f) = C_{xy}(f) - jQ_{xy}(f). \tag{6.41}$$

The C_{xy} function is the cospectral density or cospectrum and Q_{xy} is the quadrature spectral density or quadspectrum. The definitions of these two expressions are

$$C_{xy}(f) = 2\int_0^\infty [R_{xy}(\tau) + R_{xy}(-\tau)]\cos(2\pi f\tau)\,d\tau,$$

$$Q_{xy}(f) = 2\int_0^\infty [R_{xy}(\tau) - R_{xy}(-\tau)]\sin(2\pi f\tau)\,d\tau, \qquad f \geqslant 0. \tag{6.42}$$

The digital procedure would be much the same as for the PSD:

1. x_i and y_i are assumed to be zero mean sequences of N points. (If the means were not zero they would be calculated and removed.)
2. Sample correlation functions are computed for $(m+1)$ values:

$$\hat{R}_{xyr} = \frac{1}{N-r}\sum_{i=1}^{N-r} x_i\, y_{i+r},$$

$$\hat{R}_{yxr} = \frac{1}{N-r}\sum_{i=1}^{N-r} x_{i+r}\, y_i, \qquad r = 0, \ldots, m. \tag{6.43}$$

3. An appropriate lag window is selected as in Eq. (6.29), and two new correlation functions are computed:

$$\tilde{R}_{xyr} = u_{mr}^{(\ell)}\,\hat{R}_{xyr},$$

$$\tilde{R}_{yxr} = u_{mr}^{(\ell)}\,\hat{R}_{yxr}, \qquad r = 0, \ldots, m. \tag{6.44}$$

4. The co- and quadspectra are computed for various frequencies using trapezoidal integrations:

$$\hat{C}_{xy}(f) = \Delta t\left\{(\tilde{R}_{xy0} + \tilde{R}_{yx0}) + 2\sum_{r=1}^{m-1}(\tilde{R}_{xyr} + \tilde{R}_{yxr})\right.$$

$$\left.\cos(2\pi fr\Delta t) + (\tilde{R}_{xym} + \tilde{R}_{yxm})\cos(2\pi fm\Delta t)\right\},$$

$$\hat{Q}_{xy}(f) = \Delta t \left\{ (\tilde{R}_{xy0} - \tilde{R}_{xy0}) + \sum_{r-1}^{m-1} (\tilde{R}_{xyr} - \tilde{R}_{xyr}) \right.$$

$$\left. \sin(2\pi fr\Delta t) + (\tilde{R}_{xym} - \tilde{R}_{yxm}) \sin(2\pi fm\Delta t) \right\}. \tag{6.45}$$

As is the case for the PSD, a standard set of frequencies can be used:

$$f_k = \frac{k}{2m\Delta t}, \qquad k = 0, \ldots, m. \tag{6.46}$$

5. There are various ways of displaying or different forms of output for $\hat{C}_{xy}(f)$ and $\hat{Q}_{xy}(f)$. Some commonly calculated additional information is

a. The absolute value of the CSD;

$$|\hat{G}_{xy}(f)| = \sqrt{\hat{C}^2_{xy}(f) + \hat{Q}^2_{xy}(f)}. \tag{6.47}$$

b. The phase angle of the CSD;

$$\hat{\Phi}(f) = \frac{360}{2\pi} \arctan\left[\frac{\hat{Q}_{xy}(f)}{\hat{C}_{xy}(f)}\right]. \tag{6.48}$$

Note that the quadrant is always known, so that Φ ranges over 360 degrees. The most usual span is -180 to 180. A test using the signs of Q_{xy} and C_{xy} has to be made to determine the proper quadrant. Many routines that compute arctangents are designed to take care of this problem.

c. The transfer function between x and y, $H(f)$;

$$\hat{H}(f) = \frac{\hat{C}_{xy}(f) - j\hat{Q}_{xy}(f)}{\hat{G}_{xx}(f)}. \tag{6.49}$$

This is also usually rewritten in terms of the modulus and the phase angle. The modulus is

$$|\hat{H}(f)| = \sqrt{\frac{\hat{C}^2_{xy}(f) + \hat{Q}^2_{xy}(f)}{\hat{G}^2_{xx}(f)}}. \tag{6.50}$$

The phase angle is the same as for the CSD.

6.2 Direct Filtering Methods

The power spectral density can also be measured using filtering techniques. Each output value of the PSD is obtained in the following manner:

1. The data are filtered using a bandpass digital filter centered about the frequency of interest, producing a new, filtered sequence;
2. The variance of the filtered sequence is computed;
3. The variance is scaled so that it is in the correct units.

This procedure is shown schematically in Fig. 6.8. If the whole frequency interval $[0, 1/(2\Delta t)]$ were to be covered with $(m + 1)$ filters with equal bandwidths, one straightforward way of setting up the filters is as follows:

1. Have a low-pass filter with a half-power point at $1/(4m\Delta t)$ Hz.
2. Have a high-pass filter that has its half-power point at

$$\frac{1}{2\Delta t} - \frac{1}{4m\Delta t} = \frac{1}{2\Delta t}\left[1 - \frac{1}{2m}\right]. \tag{6.51}$$

This filter thus covers the interval

$$\left[\frac{1}{2\Delta t}\left(1 - \frac{1}{2m}\right), \frac{1}{2\Delta t}\right].$$

3. The remainder of the filters, a total of $(m - 1)$, could then be spaced with their right and left half-power points $1/(2m\Delta t)$ Hz apart. The half-power points of the ith filter would take the form

$$\text{Left half-power point} = \left(\frac{i - 1/2}{2m\Delta t}\right)$$

$$\text{Right half-power point} = \left(\frac{i + 1/2}{2m\Delta t}\right), \qquad i = 1, \ldots, m - 1. \tag{6.52}$$

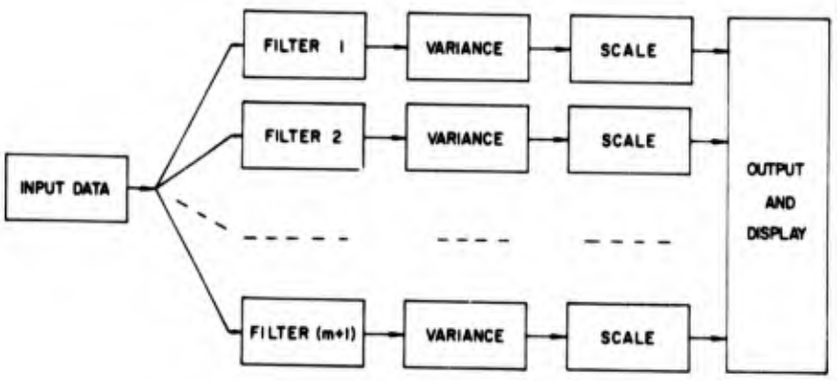


Fig. 6.8. PSD calculations using digital filters.

It is interesting to compare this procedure with the previous one. Computer running time is difficult to estimate because it involves many variables, including the individual skill of the programmer who writes the code, a factor which is difficult to evaluate. One thing that is known, however, is the approximate number of floating point arithmetic operations that will be required in each case. So, assume that

1. There are N data points and $(m + 1)$ equally spaced frequency values at which the PSD is to be estimated,

2. The correlation method uses the algorithm for computing the cosine terms recursively as described in the preceding section but does not use the fast Fourier transform,

3. The filter method uses second-order filters as described in Section 3.4. Under these conditions, the number of floating point operations required for the two cases is (approximately)

$$\begin{aligned} &\text{Floating point operations,}\\ &\text{correlation method} \end{aligned} = (m+1)\ [2N+5(m+1)]$$

$$\begin{aligned} &\text{Floating point operations,}\\ &\text{filter method} \end{aligned} = 7N(m+1). \tag{6.53}$$

The ratio of these two terms is

$$\left(\frac{\text{correlation}}{\text{filtering}}\right) = \frac{2N + 5(m+1)}{7N}. \tag{6.54}$$

Thus, for evenly spaced analyses, as $m < N$ (usually), the correlation method is the cheapest right up to the point where $(m + 1) = N$, which is where there are as many frequencies as there are data points. This analysis assumes the filter approach uses the simplest of all filters. If a more complicated filter were to be used, the cost would go up even more. For example, if a six-pole bandpass filter were employed, then Eq. (6.53) would be

$$\begin{aligned} &\text{Floating point operations,}\\ &\text{filter method, six-pole filters} \end{aligned} = 13N(m+1). \tag{6.55}$$

The filtering procedure has another serious drawback. The filters require warming up time, that is, time to go from zero to their steady state condition. The warmup time is roughly proportional to the reciprocal of the bandwidth. If the bandwidth is $1/T$, then the warmup time is approximately T seconds, which is the whole record length! Usually, a bandwidth of $1/(\Delta t)$ is required, which says that the first Δt seconds of filtered data must be discarded because the results could be biased (generally downward) if this were not done. A seemingly analogous situation in the correlation case is that the mth correlation function value,

$\hat{R}_{xm}$, is only averaged over $(N-m)$ lag products rather than the N total data values.

The picture changes considerably if equal bandwidth estimates are not required. Many users, for example, would like the bandwidths to increase with increasing frequency in some geometrical fashion. Suppose that the bandwidth of analysis at 0 Hz is $1/(m\Delta t)$, and that the PSD is subdivided into n intervals, each of which is increasing multiplicatively by a factor a. The total bandwidth will be

$$\frac{1}{m\Delta t}\,(1 + a + \ldots + a^{n-1}) = \frac{1}{2\Delta t}. \tag{6.56}$$

This can be reduced to

$$\frac{a^n - 1}{a - 1} = \frac{m}{2}. \tag{6.57}$$

Solving for n yields

$$n = \frac{\log\left[1 + (a-1)\,\frac{m}{2}\right]}{\log a}. \tag{6.58}$$

Thus, the cost of computing the variable bandwidth PSD using filters is

$$\text{Floating point operations} = N(3 + 2p)\,\frac{\log\left[1 + (a-1)\,\frac{m}{2}\right]}{\log a}. \tag{6.59}$$

In Eq. (6.59), p is the number of poles in the filters. Suppose that p and a are both set equal to two, and that m is equal to N (maximum resolution). Then,

$$\text{Floating point operations} \approx \frac{7N \log\left[1 + \frac{N}{2}\right]}{\log a}. \tag{6.60}$$

The ratio of this expression to the similar one for the correlation method yields

$$\frac{\text{correlation}}{\text{filtering}} \approx \frac{N \log a}{\log\left(1 + \frac{N}{2}\right)}. \tag{6.61}$$

An even less expensive method of implementing a third-octave type analysis is the following: A third-octave module is developed as shown in Fig. 6.9. The

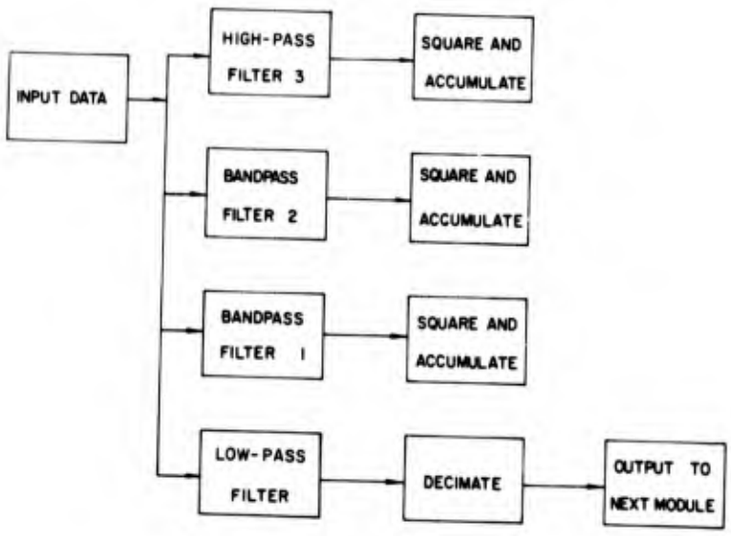


Fig. 6.9. Third-octave PSD module.

high-pass and two bandpass filters are designed to cover the upper one-half of the frequency interval in the following manner:

$$\text{hpp} = \text{half-power point}$$

$$\begin{aligned}
&\text{Filter 1} \begin{cases} \text{Left hpp} = 1/(4\Delta t) \\ \text{Right hpp} = \sqrt[3]{2}/(4\Delta t) \end{cases} \\
&\text{Filter 2} \begin{cases} \text{Left hpp} = \sqrt[3]{2}/(4\Delta t) \\ \text{Right hpp} = \sqrt[3]{4}/(4\Delta t) \end{cases} \\
&\text{Filter 3} \begin{cases} \text{Left hpp} = \sqrt[3]{4}/(4\Delta t) \\ \text{Right hpp} = 1/(2\Delta t). \end{cases}
\end{aligned} \tag{6.62}$$

The low-pass filter has its half-power point at $1/(4\Delta t)$. By the sampling theorem discussed in Chapter 1, every other data point is redundant and may be thrown away. This process of throwing away excess points, as described in Chapter 3, is known as decimation.

The process for doing the decimation is as follows: Label the original sampling interval Δt_0. Call the new sampling interval Δt_1. Then the relation is

$$2\Delta t_1 = \Delta t_0. \quad (6.63)$$

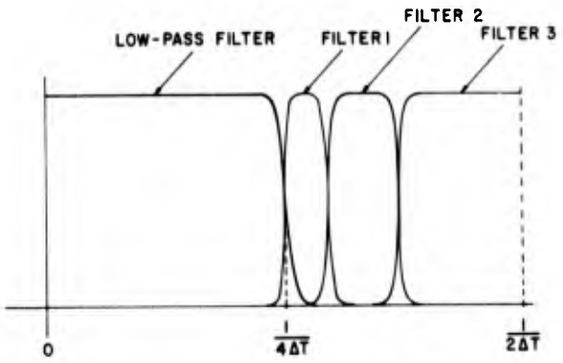


Fig. 6.10. Transfer functions of the filters in the third-octave module.

The decimated data output from the module is now the input into an identical module. The output of that is as shown in Fig. 6.10 except that the relative scale has changed. The combined coverage of these two modules is shown in Fig. 6.11. As that figure indicates, the procedure of putting the decimated output of a module into a new module can be carried on indefinitely. Suppose that the processing all of the data using the first module requires A seconds. As the first module puts out only half as many points as were continued in its input, the second module will require $A/2$ seconds. Similarly, the third module will require only $A/4$ seconds. Noting the fact that

$$\lim_{n\to\infty} \sum_{i=0}^{n} \frac{1}{2^i} = 2, \quad (6.64)$$

it is seen that a total of only $2A$ seconds is required for the whole analysis.

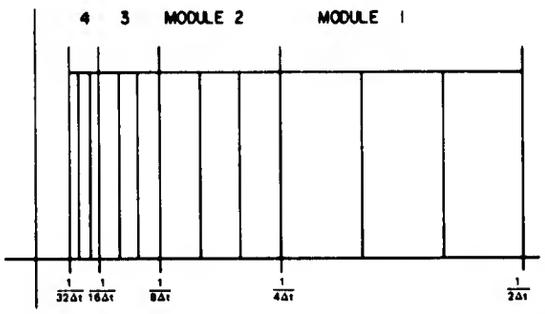


Fig. 6.11. Overall filter spacing of the third-octave analysis.

Insofar as the number of floating point operations is concerned, suppose that filters one, two, and three are six-pole filters (each requiring 13 floating point operations) and that the low-pass filter is a 12-pole filter (requiring 23 floating point operations). Then a total of 62 operations per data point per module will be required. The limiting operation of Eq. (6.64) shows that the total computation requires only twice as many operations. Thus, for N original data points, $124N$ floating point operations will be required. Comparing this with the correlation procedure yields

$$\left(\frac{\text{correlation}}{\text{3rd octave}}\right) = \frac{(m+1)\,[2N+5(m+1)]}{124N}$$

$$\approx \frac{m}{62}. \tag{6.65}$$

Thus, any time that more than 62 autocorrelation lags are required, the third-octave filtering method is cheaper. The above procedure could be recast using other divisions of the interval instead of 1/3 octaves. There could be a corresponding increase in computing cost, of course, but the cost increase would be linear. Filter weights for the above procedure are given in Table 6.1. The coefficients were obtained using the methods described in Chapter 3.

Table 6.1. Filter Weights for Recursive Filters for a Third-Octave Analysis

Filter Weight	Low-Pass with Cutoff at $f = \frac{1}{4\Delta t}$	Bandpass with Center at $f = 0.565\,\frac{1}{2\Delta t}$	Bandpass with Center at $f = 0.712\,\frac{1}{2\Delta t}$	High-Pass with Cutoff at $f = 0.793\,\frac{1}{2\Delta t}$
C	0.236568	0.041672	0.038602	0.019338
h_1	2.181491	−1.028436	−2.969963	−3.586045
h_2	−3.156543	−2.554924	−5.011952	−5.668159
h_3	3.300713	−1.585242	−5.164265	−4.974650
h_4	−2.634151	−1.958336	−3.598889	−2.535066
h_5	1.641908	−0.6001632	−1.521951	−0.7072489
h_6	−0.804673	−0.4433281	−0.3627491	−0.08405666
h_7	0.308445			
h_8	−0.09091991			
h_9	0.01995444			
h_{10}	−0.003077582			
h_{11}	0.0002981100			
h_{12}	−0.00001366317			

The last topic for this section is the effective window shape. Suppose that $y(t)$ is the truncated $x(t)$ function

$$y(t) = x(t)\, u_{T/2}(t). \tag{6.66}$$

Suppose that a bandpass filter with a center frequency of a Hz and a unit impulse response function of $h_a(t)$ is used in the analysis. Then the filter function $z(t)$ will have the form

$$z(t) = \int_{-\infty}^{\infty} h_a(t-\tau)\, y(\tau)\, d\tau$$

$$= \int_{-\infty}^{\infty} h_a(t-\tau)\, x(\tau)\, u_{T/2}(\tau)\, d\tau. \tag{6.67}$$

The Fourier transform is

$$Z(f) = H_a(f) \cdot [X(f) * U_{T/2}(f)]. \tag{6.68}$$

Note that the order of operations in the above cannot be changed. That is, in general

$$Z(f) \neq [H_a(f) \cdot X(f)] * U_{T/2}(f). \tag{6.69}$$

The power spectrum of z is

$$G_z(f) = \frac{2}{T} |H_a(f)|^2 \cdot \left|[X(f) * U_{T/2}(f)]\right|^2. \tag{6.70}$$

The estimated PSD centered at the frequency a is therefore

$$\hat{G}_x(a) = \int_0^{\infty} G_z(f)\, df. \tag{6.71}$$

The action of the bandpass filter center at a Hz is as expected; the absolute value squared of its transfer function multiplies a PSD. The PSD in question, however, is not of x but rather of y, the truncated x function.

One advantage of the filtering scheme is that the user can suppress leakage to a great extent by increasing the order of the filter. In general, for a given type of filter, the more poles that are employed, the greater the leakage suppression.

6.3 Fourier Transform Methods

The use of direct, finite, discrete Fourier transforms of time series gives rise to some relatively new problems. The Blackman-Tukey procedure of computing

the smoothed transform of a correlation function has been applied for a considerable time. As discussed in Section 6.1, the limitations of that method are well known and much experience on a variety of types of data has been obtained. Thus, most detailed practical problems have been encountered, and accepted solutions are available.

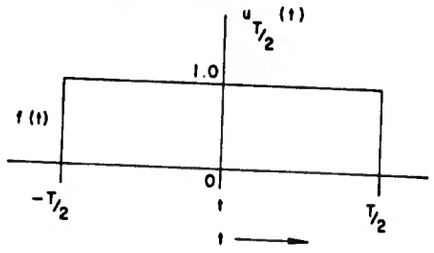
The FFT, on the other hand, is not so well established. In particular, there are questions with regard to good smoothing to obtain estimates with desired resolution, variability, and bias characteristics. Restrictions on the length of the sequence give rise to special problems. In principle, a program can be written to handle time series of arbitrary length N, and the computational speed will increase if N is any composite number not a prime. In practice, programs are usually written for series of length $N = 2^p$. Thus records of data of lengths that are not integral powers of two must be truncated to appropriate lengths or zeroes must be attached. An alternative approach is to subdivide the time history into shorter (possibly overlapping) time histories and average the final results. In many applications, such as vibration data analysis, the number of data points required presents no special problem. Digitizing rates and record lengths can often be juggled in advance to obtain the appropriate number of data points. In many other instances, however, digitized time histories of relatively inconvenient lengths might already be available and discarding data might amount to throwing away expensive information. Furthermore, the requirement of collecting a certain number of data points presents an additional constraint to the data analyst.

The following sections will indicate methods for smoothing raw spectra and also the effect of augmenting the series with zeroes. The problem of smoothing raw spectra via FFT computations has not been thoroughly studied at this time, and thus the discussion is incomplete. This is partly because the potential flexibility available via the FFT is much greater than with the correlation PSD function approach.

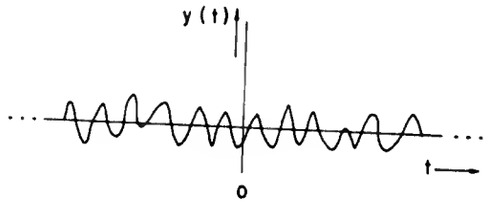
It is desirable to taper a random time series at each end to enhance the characteristics of the spectral estimates. Tapering is multiplying the time series by a "data window," analogous to multiplying the correlation function by a lag window. Thus, tapering the time series is equivalent to applying a convolution operation (see Section 6.1) to the "raw" Fourier transform. The purpose of tapering is to suppress large side lobes in the effective filter obtained with the raw transform.

As with the correlation function, one can view a finite length, random, time series as the product of a finite length boxcar $u_{T/2}(t)$ (Fig. 6.12a) and an infinitely long time history $y(t)$, as depicted in Fig. 6.12b. Thus, the finite transform of $x(t)$ may be considered as the transform

$$X(f) = \int_{-T/2}^{T/2} x(t) \exp(-j2\pi ft)\, dt = \int_{-\infty}^{\infty} y(t)\, u_{T/2} \exp(-j2\pi ft)\, dt. \qquad (6.72)$$



(a)



(b)

Fig. 6.12. (a) Boxcar function for the FFT; (b) Sample time history of length t.

Because products transform into convolutions, we have

$$X(f) = Y(f) * U_{T/2}(f)$$

where

$$U_{T/2}(f) = \int_{-T/2}^{T/2} u_{T/2}(t) \exp(-j2\pi ft)\, dt.$$

The effective filter shape with no tapering is illustrated in Fig. 6.13. Note that the width of the boxcar function is T, where T is the record length, rather than

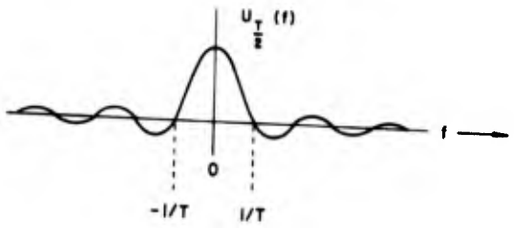


Fig. 6.13. Effective filter shape with no tapering.

the $2T_m$ as in Section 6.1, where T_m was the maximum lag value of the autocorrelation function. Correspondingly, the distance between zero crossings of the principal lobe in the frequency plane is $2/T$ rather than $1/T_m$.

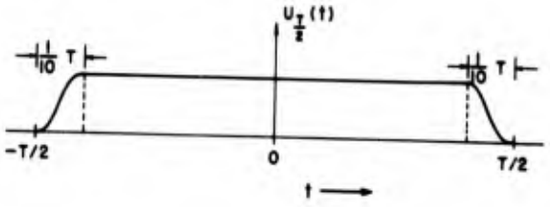


Fig. 6.14. Cosine taper data window $u_{T/2}(t)$.

Because of the large side lobes, power existing in the data at values other than at integral multiples of $1/T$ will be averaged in the value centered at $f = 0$. By sacrificing some resolution, one can improve the side lobe characteristics. Using a cosine taper over 1/10 of each end of the data rather than $u_{T/2}(t)$ is suggested in Ref. 32. Such a tapering procedure is shown in Fig. 6.14. In equation form, $u^{(4)}_{T/2}$ is

$$u^{(4)}_{T/2}(t) = \begin{cases} \cos^2 \dfrac{(5\pi t)}{T} & \left(-\dfrac{T}{2} \leq t \leq -\dfrac{4T}{10}\right) \\ 1 & \left(-\dfrac{4T}{10} \leq t \leq \dfrac{4T}{10}\right) \\ \cos^2 \dfrac{(5\pi t)}{T} & \left(\dfrac{4T}{10} \leq t \leq \dfrac{T}{2}\right) \\ 0 & \text{(otherwise).} \end{cases} \tag{6.73}$$

On the other hand, multiplying by a full cosine bell has the form

$$u^{(5)}_{T/2}(t) = \frac{1}{2}\left[1 + \cos\left(\frac{2\pi t}{T}\right)\right], \qquad \left(-\frac{T}{2} \leq t \leq \frac{T}{2}\right) \tag{6.74}$$

and can be shown to be equivalent to using discrete convolution weights (1/4, 1/2, 1/4). That is, $U^{(5)}_{T/2}(f)$

$$\left.\begin{aligned} U^{(5)}_{T/2}\left(-\frac{1}{T}\right) &= \frac{1}{4} \\ U^{(5)}_{T/2}(0) &= \frac{1}{2} \\ U^{(5)}_{T/2}\left(\frac{1}{T}\right) &= \frac{1}{4} \end{aligned}\right\} \text{ when } (0 < t < T)$$

and is zero at all other multiples of $1/T$, which are the positions at which the finite discrete Fourier transform is evaluated. Reducing the taper to only $(1/10)T$ from each end point changes this, however. The approximate shape of the effective filter is shown in Fig. 6.15.

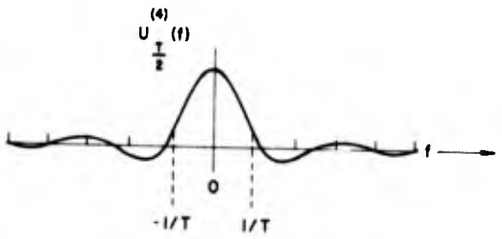


Fig. 6.15. Effective filter shape with cosine tapering.

Tapering reduces the amplitude of the time series, and in turn, the variance. Thus, a scale factor is necessary whenever tapering is performed. For $u^{(4)}_{T/2}(t)$, the spectrum estimates should be multiplied by (1/0.875). This quantity is the ratio of the areas of $u^{(2)}_{T/2}(t)$ and $u^{(4)}_{T/2}(t)$. Thus, one obtains modified spectral estimates of a roughly triangular shape. Note that the width of the main lobe is slightly increased for $U^{(4)}_{T/2}(f)$. It will be noted that $1/T$ will be very close to a half-power point bandwidth for $U^{(4)}_{T/2}(f)$.

The spacing of discrete Fourier transform values is

$$\Delta f = 1/T = 1/N\Delta t. \tag{6.75}$$

The spacing between the first zero crossings on both sides of the main lobe for $U_{T/2}$ is $B_e = 1/T$. When zeroes are attached to the sequence, nothing is contributed to the basic shape of $U_{T/2}(f)$ and hence the width of the main lobe is unchanged. However, because of the nature of the computational formula, the spacing of the estimates is based on the augmented record length and is

$$\Delta f' = \frac{1}{(N+N_z)\,\Delta t}, \tag{6.76}$$

where N_z is the number of zeroes attached. For example, if an equal number of zeroes, $N_z = N$, is attached, the spacing is halved and appears as in Fig. 6.16. This change in spacing leads to problems, since the effective convolution applied to the raw Fourier transform may not lead to the desired side lobe cancellation. In fact, one could enlarge the side lobes. The problems of modifying Fourier transforms when arbitrary numbers of zeroes must be added is under study. The problem is essentially that of empirical filter design. Another approach is discussed further here.

Power spectra are obtained from the Fourier transform by the formula

$$\begin{aligned}\tilde{G}_{xk} &= \frac{2\Delta t}{N}\,|X_k|^2 \\ &= 2\frac{\Delta t}{N}([\mathrm{Re}\,X_k]^2 + [\mathrm{Im}\,X_k]^2).\end{aligned} \tag{6.77}$$

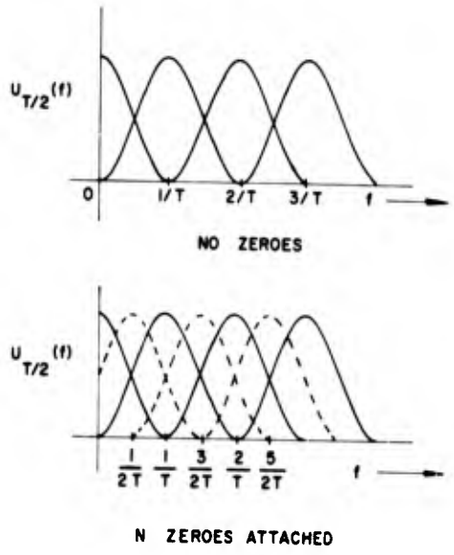


Fig. 6.16. Effect on spacing of spectral averages when zeroes are added.

Cross spectra are obtained from the more general formula

$$\tilde{G}_{xyk} = \frac{2\Delta t}{N} [X_k^* Y_k]. \tag{6.78}$$

If the number of data points $N = 2^p$, then the spacing of the raw estimates is illustrated in Fig. 6.16. In statistical terms, the raw power spectra $\tilde{G}_{xk}$ can be shown (as discussed in Section 6.5) to be approximately χ^2 variables with two degrees of freedom (d.f.). That is, if the data are Gaussian, then each spectrum point is the sum of two squared Gaussian variables. The standard error of the unsmoothed spectrum estimates is shown in Section 6.5 to be

$$\epsilon = \left[\frac{2n}{n^2}\right]^{1/2} = \left[\frac{2 \cdot 2}{2 \cdot 2}\right]^{1/2} = 1$$

or 100 percent. This is not satisfactory for most purposes. If the spectrum is smooth, the estimates at a spacing of $1/T$ are approximately uncorrelated (the correlation is the overlap between neighboring estimates as indicated in Fig. 6.16). Hence if ℓ neighboring estimates are averaged,

$$\hat{G}_k = \frac{1}{\ell} [\tilde{G}_k + \tilde{G}_{k+1} + \ldots + \tilde{G}_{k+\ell-1}] \tag{6.79}$$

a χ^2 variable with roughly $2\ell n$ d.f. is obtained by the χ^2 addition theorem. The effective filter shape is then roughly trapezoidal, since adding together triangles that overlap at half-power points gives a trapezoid as indicated in Fig.6.17.

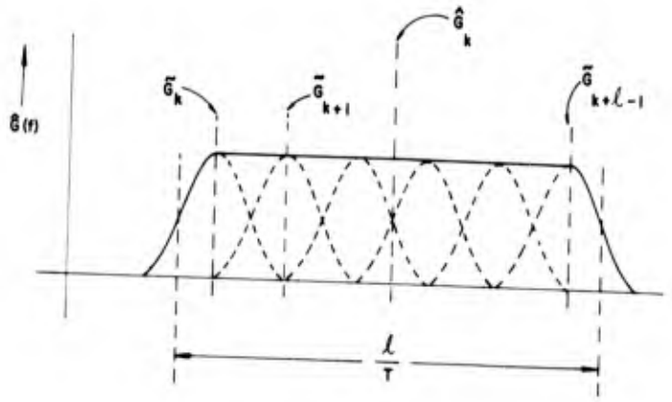


Fig. 6.17. Effective filter shape after averaging.

Note that the effective resolution bandwidth is now approximately

$$B_e = \ell \Delta f = \frac{\ell}{T}. \tag{6.80}$$

The estimate $\hat{G}_k$ may be considered as representing the midpoint of the frequency interval from $k\Delta f$ to $(k+\ell-1)\,\Delta f$.

The final estimates $\hat{G}_k$ can be spaced in any manner desired. If it is satisfactory to have final estimates that overlap at half-power points, then one would average and decimate to obtain N/ℓ final spectrum estimates that are approximately uncorrelated. In most B-T computer programs, the spacing is $B_e/2$ so that $2N/\ell$ final estimates are obtained that contain considerable correlation between any contiguous pair of estimates.

Spectrum estimates identical to the B-T method can be obtained using the direct Fourier transform approach (although this is by no means necessarily desirable from a statistical standpoint). A seemingly long way around is taken which, in fact, is usually faster than the direct correlation computation. The procedure is as follows:

1. Compute the raw transform X_k.
2. Compute the raw spectrum $\tilde{G}_{xk} = (2\Delta t/N)\,|X_k|^2$.
3. Compute the inverse FFT to obtain the autocorrelation function $\hat{R}_{xr} = \mathcal{F}^{-1}\,[\tilde{G}_{xk}]$.
4. Multiply $\hat{R}_{xr}$ by lag window u_{mr}.

5. Compute the final smoothed spectrum $\hat{G}_k$ from the direct FFT of the weighted autocorrelation $\hat{G}_{xk} = \mathcal{F}[u_{mr}\hat{R}_{xr}]$.

As discussed in Section 5.5, this computational procedure must be modified since the usual correlation function is not obtained at Step 3. Zeroes must be attached to the original data sequence (or special computational tricks applied) to obtain the desired correlation function. When this is done, the right-hand half of the circular correlation can be discarded as discussed in Section 5.5 and the remainder used in Steps 3, 4, and 5.

The modified computational procedure is

1. Augment sequence x_i with N zeroes to obtain sequence of length $2N$.
2. Compute the raw transform $X_k, k = 0, 1, \ldots, 2N-1$.
3. Compute the raw spectrum

$$\tilde{G}_{xk} = \frac{2\Delta t}{N}|X_k|^2, \qquad k = 0, 1, \ldots, 2N-1. \tag{6.81}$$

4. Compute the inverse transform of G_{xk},

$$\tilde{R}_{xr} = \mathcal{F}^{-1}[\tilde{G}_{xk}], \qquad r = 0, 1, \ldots, 2N-1. \tag{6.82}$$

5. Discard the last N values of $\hat{R}_{xr}$ and retain $\hat{R}_{xr}, r = 0, 1, \ldots, N$.
6. Multiply $\hat{R}_{xr}$ by lag window u_{mr}.
7. Compute the direct transform of $\hat{R}_{xr}$

$$\hat{G}_{xk} = \mathcal{F}[u_{mr}^{i}\hat{R}_{xr}], \qquad k = 0, 1, \ldots, N. \tag{6.83}$$

Note that a given sequence need be augmented with only as many zeroes as lags. The number of zeroes necessary to obtain sequence lengths which are a power of 2 and the number of lags needed or desired will dictate the final sequence length. The amount of correlation function retained need only be the smallest power of 2 which exceeds the number of lags to be used. Step 5 can be modified to discard as many correlation values as possible to obtain the minimum power of 2 larger than the number of lags desired.

This procedure will provide B-T type estimates. Whether or not B-T results are duplicated exactly will depend on the relation of the point at which the desired lag window goes to zero and the number of correlation values that must be used for the FFT. The lag window dictates the bandwidth, so the statistical characteristics will be those of B-T estimates, although the spacing may not be duplicated.

Reasonable spectral estimates can be obtained for sequences of length $N = 2^p$ by the following procedure:

1. Taper the original sequence $\{x_i\}$ using Eq. (6.72).
2. First fill out the data sequence with zero data points to obtain 2^{p+1} total data points if this spectrum is later to be inverse transformed to obtain a correlation function.

3. Compute the finite Fourier transform of this augmented data sequence:

$$X_k = \sum_{i=0}^{N-1} x_i^{-j2\pi \frac{ik}{N}}, \qquad k = 0, 1, \ldots, N-1. \tag{6.84}$$

4. Compute the absolute value squared, scaled appropriately to obtain the "raw" power spectral estimates.

$$\tilde{G}_{xk} = \frac{2\Delta t}{N} |X_k|^2, \qquad k = 0, 1, \ldots, N-1. \tag{6.85}$$

5. Adjust the estimates for the scale factor due to tapering

$$(1/0.875)\, \tilde{G}_x(k) \rightarrow \tilde{G}_x(k), \qquad k = 0, 1, \ldots, N-1.$$

6. If cross spectral density functions are desired, a second Fourier transform Y_k is obtained. Then the "raw" cross spectral density estimate is obtained from the equation

$$\begin{aligned} \tilde{G}_{xyk} &= \frac{2\Delta t}{N} [X_k^* Y_k], \\ &= \tilde{C}_{xyk} - j\tilde{Q}_{xyk}, \qquad k = 0, 1, 2, \ldots, N-1. \end{aligned} \tag{6.86}$$

7. Smoothed estimates are then obtained by averaging ℓ contiguous raw estimates to yield

$$\hat{G}_{xk} = \frac{1}{\ell} \sum_{j=1}^{\ell} \tilde{G}_{xx(k+j)}, \tag{6.87}$$

or

$$\hat{G}_{xyk} = \frac{1}{\ell} \sum_{j=1}^{\ell} \tilde{G}_{xy(k+j)}, \qquad k = \ell, 2\ell, 3\ell, \ldots, m. \tag{6.88}$$

8. For cross spectra, the squared absolute value, $|\hat{G}_{xyk}|^2$, and the phase, $\hat{\theta}_{xyk}$, will normally be the final results given by

$$\begin{aligned} |\hat{G}_{xyk}|^2 &= \hat{C}_{xyk}^2 + \hat{Q}_{xyk}^2, \\ \hat{\theta}_{xyk} &= \frac{360}{2\pi} \arctan [\hat{Q}_{xyk}/\hat{C}_{xyk}]. \end{aligned} \tag{6.89}$$

Computational time for a correlation PSD is usually dictated almost entirely by the correlation computational time. This is about $2N \cdot m$ multiply and add operations. The necessary Fourier transform time is usually not significant because the correlation function length is so much less than the time history length. Similarly, the majority of computational time for spectra, calculated directly by FFT procedures, is the basic FFT time. The additional time for absolute value squared, smoothing, and sine-cosine generation will usually not be crucial. Thus, a fairly direct time comparison can be made, assuming $8Np$ operations as required for the $N = 2^p$ type FFT. It must be kept in mind that special procedures and four-point transform techniques can reduce this time. The basic speed ratio for FFT versus B-T for a single PSD is roughly

$$\text{SR} = \text{Speed Ratio} = \frac{2Nm}{8Np} = \frac{m}{4p}. \tag{6.90}$$

Consider $N = 8000$ and $p = 13$. Typically, $m = 400$ lags might be selected. Then the speed ratio is

$$\text{SR} = \frac{400}{(4)\ (13)} = 7.7. \tag{6.91}$$

The speed advantage of the FFT approach increases when many time series are involved and cross spectra are required. If two PSD's and a cross spectrum are necessary, four times the number of operations in the B-T method will be required since two halves of the crosscorrelation must be computed. The FFT effort only doubles, however, since only two FFT's are required. The only new computation is the cross product of the two transforms. The speed ratio therefore becomes

$$\text{SR} = \frac{2\,m}{4\,(4p)} = 15.4. \tag{6.92}$$

The third-octave procedure discussed in Section 6.2 may be faster for large N. The ratio is

$$\text{SR} = \frac{124N}{8Np} = \frac{15.5}{p}. \tag{6.93}$$

Thus for large N, say N greater than 45,000, the third-octave analysis is cheaper than the FFT PSD.

These time comparisons are only guidelines, however. A complete program includes significant amounts of data input and manipulation, in addition to preparation of output plots. Final speed ratios for an entire program might typically range from 5 to 10.

6.4 Special Methods

Various special procedures for reducing computer time have been suggested from time to time. These tend to take the form of variable-bandwidth analyses.

A cascade method for computing PSD's is presented in Ref. 9. A somewhat more elaborate method was implemented and tested at Douglas Aircraft Company and reported in Appendix VII of Ref. 33. The Douglas method is a modification of the cascade procedure as outlined in Ref. 9. The resulting PSD differs from the standard one in that the bandwidth of analysis increases with increasing frequency. This increase is neither linear nor geometric but, roughly speaking, equally spaced through each octave.

The Douglas procedure begins with computing an autocorrelation of the function with $m = 12$, weighting the correlation function with the Hamming lag window, and then computing only the eight values of the PSD that correspond to the eight highest frequencies in an equally spaced analysis. The time sequence is then filtered with a low-pass numerical filter, and every other point discarded. An autocorrelation function of the filtered data is computed with $m = 12$, and is then weighted as before. Only the central four values of the PSD are calculated. The center frequencies of these four estimates fit in directly below the eight estimates calculated in the previous stage.

The cycle of filtering, discarding, correlating, smoothing, and computing the central four estimates is repeated until the data sequence contains fewer than forty points. Finally, the resulting spectrum is adjusted to compensate for the low-pass filtering. This had to be done because the filter used by Douglas was of the binomial type discussed in Section 3.2, which does not have sharp cutoffs. To eliminate foldback, the filter was chosen so that the power was negligible midway between 0 Hz and the folding frequency. The result of this is that the power at the frequencies of interest has been altered and thus requires compensation.

Test cases were run at Douglas using random numbers with a flat power spectral density. A typical pair of results is shown in Fig. 6.18. The standard PSD was computed with $m = 125$, $N = 6000$, and $\Delta t = 0.04$, so that $B = 0.2$ Hz. It required a total of seven minutes of IBM 704 time. The PSD computed using the cascade method required only one minute of computer time. The agreement seems good. The bandwidths match at approximately 0.7 Hz.

Although considerable savings may be realized with the cascade procedure, it lacks flexibility because only the preset bandwidths are available and the span of data is limited by the core storage of the computer. The Douglas program for computing the cascade PSD is no longer extant. If it were to be recoded, no doubt many changes would be made to modernize it, particularly in the numerical filtering.

The *modified autocorrelation method* (MAM) is basically a more flexible version of the cascade procedure, extended to take in larger spans of data and to take advantage of a geometrical or other unequal frequency subdivision

scheme. The difference between MAM and the cascade method is that MAM is designed to accommodate any frequency division scheme that subdivides the frequency range so that the analysis bandwidth does not decrease with increasing frequency. That is accomplished by computing autocorrelations for varying

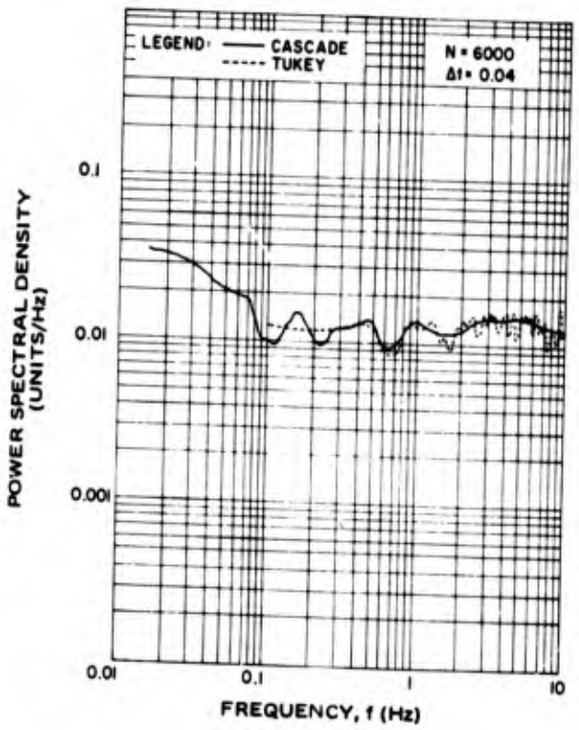


Fig. 6.18. PSD of random noise using standard and cascade methods.

sizes of sampling increments and using portions of these autocorrelation functions to suit the bandwidth requirements. In detail, the three phases required by this type of procedure are as follows:

1. The first phase generates the correlation function. These are computed on an input data frame by data frame basis. The program can be made to compute as long as frames are supplied to it; hence, the ability to handle arbitrarily long sequences. Between each two levels of correlation there is filtering and decimation so that the lower levels of correlation use progressively fewer data points. The time span for each level stays approximately the same since the sampling interval is increasing.

2. The second phase computes the spectral density. There are two differences at this step; more than one correlation function has been computed, and more than one weighting function is used with each frequency. It is this

last option that permits variation of bandwidth within a frequency range corresponding to one autocorrelation function.

3. The third and last phase combines the results and puts out a combined printout, plot, and punched cards, as required.

An example will clarify the above. Suppose it is desired to compute a power spectral density of a set of 8000 data points that were obtained by digitizing the function $\{x_t\}$ at a 2000-sample per second (sps) rate, and to compute the power according to a progression as given in Table 6.2.* There are a number of ways in which the procedure could be set up, but suppose that three correlation functions are to be computed with a resulting flow of information as shown schematically in Fig. 6.19. Table 6.2 shows one choice of parameters for an analysis scheme of three bandwidths' resolution.

Table 6.2. MAM PSD Parameters†

Band p	Range	i	Δt_p	Narrowest B	M_p	N_p	M_pN_p
1	500-1000 (high)	15-20	0.0005 (2000 sps)	71.43	28	8000	224,000
2	250-500 (middle)	10-14	0.001 (1000 sps)	47.62	21	4000	84,000
3	0-250 (low)	1-9	0.002 (500 sps)	4.76	105	2000	210,000
							518,000

†$MN = (420)(8000) = 3{,}360{,}000$, $MN/\Sigma M_pN_p = 6.48$.

Before passing on to the details of calculations, consider some of the economic advantages of the MAM method over the equal-interval method. The equal-interval method would require the use of an m that corresponds to the narrowest bandwidth to be analyzed. In this case,

$$m = \frac{2000}{4.76} \approx 420. \tag{6.94}$$

The product of m and N is therefore about 3.36×10^9. As noted before, cost is proportional to this product. The mN product is tabulated in Table 6.2 for each of the three levels. The sum of the three products is 0.52×10^6, so that

*This frequency scheme is arithmetical.

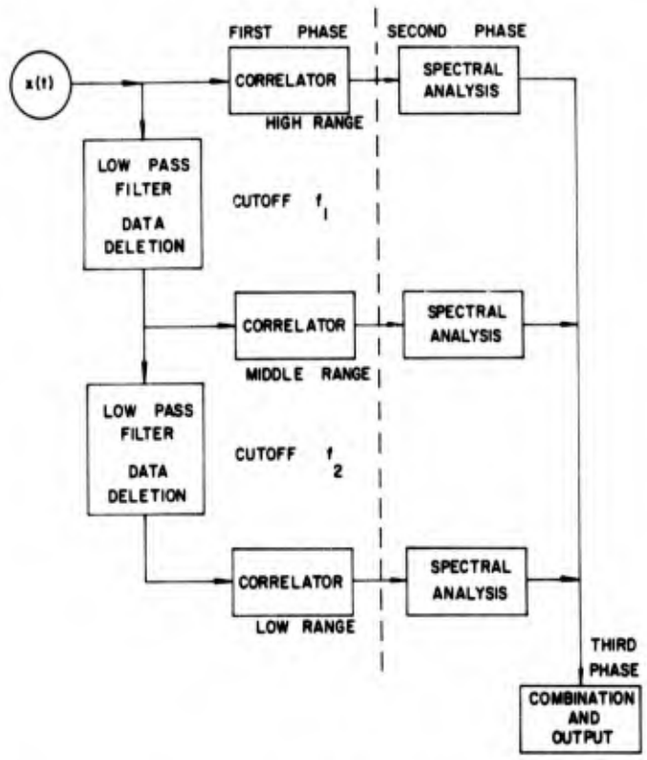


Fig. 6.19. Schematic of a typical setup of a MAM PSD program.

$$\frac{mN}{\sum_{p=1}^{3} m_p N_p} = 6.48, \tag{6.95}$$

where m_p and N_p are the different values used for the three different correlation functions. That is, the cost of the standard correlation function is nearly six and one-half times the cost of the MAM correlations, not counting the cost of the filtering. Empirical information on the cost of filtering as well as the cost of computer bookkeeping would indicate that the actual figure is closer to three to one. Therefore, for the above configuration, the MAM power spectral density would cost one-third the price of an equal interval analysis.

Two of the specialized numerical techniques needed for this analysis are numerical filtering and a variable-bandwidth Fourier transformation. Numerical low-pass filtering has been discussed in Section 3.6. The expression required for the Fourier transformation is

$$G_k = 2\Delta t_p \left\{ R_{p0} + 2 \left[\sum_{j=1}^{m_k - 1} \left(\frac{1}{2} + \frac{1}{2} \cos \left[\frac{\pi j}{m_k} \right] \right) \cos (2\pi f_k j \Delta t) R_{pj} \right] \right\}, \tag{6.96}$$

where the p subscript indicates the different correlation functions. Note that a separate and not necessarily integral number m_k is required for each frequency f_k so that the autocorrelation points are not necessarily summed over the entire length of the correlation function computed for each level. The bandwidth corresponding to the frequency f_k is $1/(\Delta t_p m_k)$. A computer program written to implement this process would probably be general, with an arbitrary number of levels and sets of frequencies. Specific tables of frequencies, bandwidths, etc., could either be made available on a semipermanent basis or be given to the program as specific input.

While the procedures discussed in this section have been implemented and found to achieve savings over the basic method, these savings do not compare with those obtainable using the FFT or numerical filtering PSD techniques. It is more from a standpoint of historical interest than current usage that a discussion of them is included.

6.5 Statistical Error

The purpose of this section is to outline the derivations used to obtain error parameters. An understanding of the material would be helpful, but it is not entirely necessary when applying the results. The casual reader may, therefore, skip over this section. The principal results are summarized in the section that follows.

Statistical error, as opposed to instrumentation error, is the uncertainty in PSD measurements due to the amount of data gathered, the underlying probabilistic nature of the data, and the method used in deriving the desired parameter. Suppose that the parameter Φ is being estimated and that the estimate of Φ is $\hat{\Phi}$. Such an estimate is unbiased if

$$\mathbf{E}\,[\hat{\Phi}] = \Phi. \tag{6.97}$$

The mean square error of the estimate is defined by

$$\text{mean square error} = \mathbf{E}\,[(\hat{\Phi} - \Phi)^2]. \tag{6.98}$$

The estimate $\hat{\Phi}$ will usually be a function of the record length T. If $\hat{\Phi}$ is a consistent estimate of Φ, then

$$\lim_{T \to \infty} \mathbf{E}\,[(\hat{\Phi} - \Phi)^2] = 0. \tag{6.99}$$

This implies that as the record length becomes larger, the mean square error tends to decrease.

Consider the estimate of the PSD of x at the frequency f using the discrete Fourier transform of x

$$\hat{G}_x(f) = \frac{2}{N\Delta t}\left(\Delta t \sum_{i=1}^{N} x_i\, e^{j2\pi fi\Delta t}\right)\left(\Delta t \sum_{k=1}^{N} x_k\, e^{-2\pi fk\Delta t}\right)$$

$$= \frac{2\Delta t}{N}\left(\sum_{i=1}^{N}\sum_{k=1}^{N} x_i x_k\, e^{-j2\pi f\Delta t(k-i)}\right). \tag{6.100}$$

By defining r to be equal to $(k - i)$ and introducing a new variable ℓ, the above can be rearranged to yield

$$\hat{G}_x(f) = 2\Delta t \sum_{r=-(N-1)}^{N-1} \cos(2\pi fr\Delta t)\left(\frac{1}{N}\sum_{\ell=1}^{N-|r|} x_\ell\, x_{\ell+|r|}\right). \tag{6.101}$$

Expectations may be taken of this, resulting in

$$\mathbf{E}\,[\hat{G}_x(f)] = 2\Delta t \sum_{r=-(N-1)}^{N-1} \cos(2\pi fr\Delta t)\left[\frac{N-r}{N} R_{xr}\right]. \tag{6.102}$$

Suppose that $\{x_i\}$ is uncorrelated white noise. That is

$$R_{xr} = \begin{cases} \sigma^2 & r = 0 \\ 0 & \text{otherwise.} \end{cases} \tag{6.103}$$

Then

$$\mathbf{E}\,[\hat{G}_x(f)] = 2\Delta t\,\sigma^2. \tag{6.104}$$

Note that this evaluation corresponds to a bandwidth of $(1/N\Delta t)$ Hz. If $N/2$ bands that do not overlap are formed, they will completely cover the frequency range. Thus, the integral over f of $\mathbf{E}\,[\hat{G}_x(F)]$ is σ^2. For white noise at least, the estimate is therefore unbiased.

Next consider the mean square error of $\hat{G}_x(f)$ given by

$$\epsilon^2 \equiv \text{mean square error}$$

$$= \mathbf{E}\,[(\hat{G}_x(f) - G_x(f))^2]/G_x^2(f)$$

$$= \mathbf{E}\,[\hat{G}_x^2(f) - 2\hat{G}_x(f)\,G_x(f) + G_x^2(f)]/G_x^2(f)$$

$$= \{\mathbf{E}\,[\hat{G}_x^2(F)] - 2G_x(f)\,\mathbf{E}\,[\hat{G}_x(f)] + G_x^2(f)\}/G_x^2(f). \tag{6.105}$$

Suppose that white noise is again being discussed. Then

$$\epsilon^2 = \{\mathbf{E}\,[\hat{G}_x{}^2(F)] - 4\Delta t\,\sigma^2\,[2\Delta t\sigma^2] + [2\Delta t\sigma^2]\}/G_x{}^2(f)$$

$$= \{\mathbf{E}\,[\hat{G}_x{}^2(f)] - 4(\Delta t)^2\,\sigma^4\}/(4\Delta t^2\,\sigma^4). \tag{6.106}$$

Before this can be evaluated, another assumption about $\{x_l\}$ must be made. Up to now, the only assumptions required were that $\{x_l\}$ have a zero mean and be uncorrelated. The new assumption is that $\{x_l\}$ is Gaussian. The reason for making this assumption is to be able to employ a very useful feature of the Gaussian distribution; namely, that if x_a, x_b, x_c, x_d are any four samples from $\{x_l\}$ then

$$\begin{aligned}\mathbf{E}\,[x_a\,x_b\,x_c\,x_d] &= R_{x_a\,x_b}\,R_{x_c\,x_d}\\ &= R_{x_a\,x_c}\,R_{x_b\,x_d}\\ &= R_{x_a\,x_d}\,R_{x_c\,x_b}.\end{aligned} \tag{6.107}$$

This property is discussed (as a problem) in Ref. 34. If $\{x_l\}$ is uncorrelated, then

$$\mathbf{E}\,[x_a\,x_b\,x_c\,x_d] = \begin{cases} 3\sigma^4 & a = b = c = d \\ \sigma^4 & a = b,\ c = d,\ a \neq c \\ \sigma^4 & a = c,\ b = d,\ a \neq d \\ \sigma^4 & a = d,\ b = c,\ a \neq b \\ 0 & \text{otherwise.} \end{cases} \tag{6.108}$$

Thus,

$$\mathbf{E}\,[\hat{G}_x{}^2(f)] = \left(\frac{2\Delta t}{N}\right)^2 \mathbf{E}\left\{\sum_{a=1}^{N}\sum_{b=1}^{N}\sum_{c=1}^{N}\sum_{d=1}^{N} x_a\,x_b\,x_c\,x_d \cdot \exp\left[-j2\pi f\Delta t\,(a-b+c-d)\right]\right\}.$$

This must be evaluated for each of the four cases in which the expectation is not zero.

Case 1. $(a = b = c = d)$ This holds in exactly N places. Therefore, the contribution is $3N\sigma^4$.

Case 2. ($a = b$, $c = d$, $a \neq c$) There are $N^2 - N$ places where this occurs. As the exponent is zero, the contribution is $(N^2 - N)\,\sigma^4$.

Case 3. Same as Case 2. The contribution is $(N^2 - N)\,\sigma^4$.

Case 4. This case is more complex. Its contribution is

$$\sigma^4 \sum_a \sum_b \exp\left[-j4\pi f\Delta t\,(a-b)\right] - N\sigma^4 = \sigma^4 \left(e^{-j4\pi f\Delta t}\,\frac{1-e^{-j4\pi f\Delta tN}}{1-e^{-j4\pi f\Delta t}}\right)$$

$$\cdot \left(e^{+j\pi f\Delta t}\,\frac{1-e^{+j4\pi f\Delta tN}}{1-e^{+j4\pi f\Delta t}}\right) - N\sigma^4$$

$$= \sigma^4\,\frac{2-2\cos(4\pi f\Delta tN)}{2-2\cos(4\pi f\Delta t)} - N\sigma^4$$

$$= \sigma^4 \left(\frac{\sin(2\pi f\Delta tN)}{\sin(2\pi f\Delta t)}\right)^2 - N\sigma^4.$$

In summary,

$$\mathbf{E}\left[\hat{G}_x{}^2(f)\right] = \left(\frac{2\Delta t}{N}\right)^2$$

$$\cdot \left[3N\sigma^4 + 2\,(N^2-N)\,\sigma^4 + \sigma^4 \left(\frac{\sin(2\pi f\Delta tN)}{\sin(2\pi f\Delta t)}\right)^2 - N\sigma^4\right]$$

$$= \left(\frac{2\Delta t}{N}\right)^2 \sigma^4 \left[2N^2 + \left(\frac{\sin(2\pi f\Delta tN)}{\sin(2\pi f\Delta t)}\right)^2\right]. \tag{6.109}$$

The mean square error therefore is

$$\epsilon^2 = \left(\frac{2\Delta t}{N}\right)^2 \sigma^4 \left[2N^2 + \left(\frac{\sin(2\pi f\Delta tN)}{\sin(2\pi f\Delta t)}\right)^2 - N^2\right] \Big/ (4\Delta t^2\,\sigma^4)$$

$$= 1 + \frac{1}{N^2}\left(\frac{\sin(2\pi f\Delta tN)}{\sin(2\pi f\Delta t)}\right)^2. \tag{6.110}$$

Note that if $f = p/(N\Delta t)$, then

$$\epsilon^2 = 1. \tag{6.111}$$

Thus, if $\{x_i\}$ is uncorrelated Gaussian noise, then the value of the normalized standard error for the estimate is greater than or equal to unity (the standard deviation of the estimate is greater than or equal to the quantity being estimated).

Furthermore, while it does decrease with increasing N, it does not vanish in the limit but goes to unity. Therefore, $\hat{G}_x(f)$ is not a consistent estimate of $G(f)$. This result has been shown by many authors such as Hannan [1], and is a fundamental limitation upon the PSD estimation process. As will be seen in the next section, this problem is overcome in practice if a sacrifice in resolving power is made.

As a final topic for this section, consider the random variable X_k where, as usual

$$X_k = \Delta t \sum_{i=1}^{N} x_i \exp\left[-j(2\pi ki)/N\right], \qquad k = 1, \ldots, N/2. \quad (6.112)$$

If $\{x_i\}$ is Gaussian, then by a well-known theorem in probability theory, so is X_k. Assuming $\{x_i\}$ to have a zero mean and be uncorrelated, then

$$\begin{aligned} \mathrm{E}[X_k] &= \mathrm{E}\left\{\Delta t \sum_{i=1}^{N} x_i \exp\left[-j(2\pi ki)/N\right]\right\} \\ &= \Delta t \sum_{i=1}^{N} \mathrm{E}(x_i) \exp\left[-j(2\pi ki)/N\right] \\ &= 0 \\ \mathrm{Var}[X_k] &= \mathrm{E}[X_k X_k^*] \\ &= N \Delta t^2 \sigma^2. \end{aligned} \quad (6.113)$$

Furthermore, if $p \neq q$, then X_p and X_q are uncorrelated:

$$\mathrm{E}[X_p X_q^*] = \mathrm{E}\left[\Delta t \sum_{i=1}^{N} x_i \exp\left[-j(2\pi ki)/N\right]\right] \times$$

$$\left[\Delta t \sum_{k=1}^{N} x_k \exp\left[j(2\pi kq)/N\right]\right]$$

$$= (\Delta t)^2 \, \mathrm{E}\left\{\sum_{i=1}^{N} \sum_{k=1}^{N} x_i x_k \exp\left[-j(2\pi(ip-kq))/N\right]\right\}$$

$$\mathbf{E}[X_p X_q^*] = (\Delta t)^2 \sum_{i=1}^{N} \sum_{k=1}^{N} \mathbf{E}(x_i x_k) \exp[-j(2\pi(ip-kq)),N]$$

$$= (\Delta t)^2 \sigma^2 \sum_{i=1}^{N} \exp[-j(2\pi(ip-kq))/N]$$

$$= (\Delta t)^2 \sigma^2 \exp[-j(2\pi(p-q))/N] \quad \frac{1-\exp[-j2\pi(p-q)]}{1-\exp\{[-j(2\pi(p-q))]/N\}}$$

$$= \begin{cases} (\Delta t)^2 N \sigma^2 & p = q \\ 0 & \text{otherwise.} \end{cases} \tag{6.114}$$

The complex sequence $X_k, k = 1, \ldots, N/2$ can be broken down into real and imaginary parts. Define

$$\mathrm{Re}\, X_k = \text{Real part of } [X_k]$$

$$= \Delta t \sum_{i=1}^{N} x_i \cos\left(\frac{2\pi ik}{N}\right),$$

$$\mathrm{Im}\, X_k = \text{Imaginary part of } [X_k]$$

$$= \Delta t \sum_{i=1}^{N} x_i \sin\left(\frac{2\pi ik}{N}\right),$$

$$X_k = \mathrm{Re}\, X_k + j\, \mathrm{Im}\, X_k, \qquad k = 1, \ldots, N/2. \tag{6.115}$$

The expected value of the product of the real and imaginary parts may be shown to be zero for any p:

$$\mathbf{E}[\mathrm{Re}\, X_k\, \mathrm{Im}\, X_k] = \mathbf{E}\left[(\Delta t)^2 \sum_{j=1}^{N} \sum_{\ell=1}^{N} X_j X_\ell \cos\left(\frac{2\pi jk}{N}\right) \sin\left(\frac{2\pi \ell k}{N}\right)\right]$$

$$= (\Delta t)^2 \sigma^2 \sum_{j=1}^{N} \cos\left(\frac{2\pi jk}{N}\right) \sin\left(\frac{2\pi jk}{N}\right)$$

$$\mathbf{E}\,[\mathrm{Re}\,X_k\,\mathrm{Im}\,X_k] = \frac{(\Delta t)^2}{2}\,\sigma^2 \sum_{j=1}^{N} \sin\left(\frac{4\pi jk}{N}\right)$$

$$= \frac{(\Delta t)^2\,\sigma^2}{2}\,\frac{\sin(2\pi k)\,\sin\left(\dfrac{N+1}{N}\,\pi k\right)}{\sin\left(\dfrac{\pi k}{N}\right)}$$

$$= 0. \tag{6.116}$$

Thus, the $N/2$ values each of $\{\mathrm{Re}\,X_p\}$ and $\{\mathrm{Im}\,X_p\}$ are independent Gaussian random variables with zero mean and with variance $(\Delta t\sigma^2)$. The importance of this lies in the following: An estimate of the power around the frequency f, where

$$f = \frac{k}{N\Delta t}, \qquad k = 1, \ldots, N/2, \tag{6.117}$$

can be made by taking an average of $2(2M + 1)$ values

$$\hat{G}_x(f) = \frac{1}{(2M+1)}\left(\frac{2}{N\Delta t}\right)\sum_{q=-M}^{k+M}\left[(\mathrm{Re}\,X_q)^2 + \mathrm{Im}\,X_q)^2\right]$$

$$f = \frac{k}{N\Delta t}. \tag{6.118}$$

The expectation of this estimate is

$$\mathbf{E}\,[\hat{G}_x(f)] = 2\Delta t\,\sigma^2, \tag{6.119}$$

the same as that obtained for the previous formulation of $\hat{G}_x(f)$ in Eq. (6.104). The bandwidth is broader because it now covers $(2M + 1)/(N\Delta t)$ Hz. On the other hand, because it is estimated using $2(2M + 1)$ rather than the basic two estimates, the variability is much less. In particular, the normalized standard error is

$$\epsilon^2 = \frac{1}{2M + 1},$$

or

$$\epsilon = \sqrt{\frac{1}{2M + 1}}. \tag{6.120}$$

This result uses another well-known property of uncorrelated, identically distributed, Gaussian random variables, namely, that the variance of their sum is equal to their common variance divided by the number of observations.

Define B_e as the bandwidth of the estimate. Then

$$B_e = (2M + 1)\,\Delta t$$

$$= (2M + 1)\left(\frac{1}{N\Delta t}\right)$$

$$= \frac{(2M + 1)}{T}. \tag{6.121}$$

Thus,

$$(2M + 1) = B_e\,T. \tag{6.122}$$

The expression for the normalized standard error can therefore be put in the form

$$\epsilon^2 = \frac{1}{B_e\,T}. \tag{6.123}$$

While this result was derived by employing the Fourier transform, it may also be used when discussing PSD's obtained by the other methods. In that case, the value B_e is taken to be the distance between the half-power points of the filter or lag window. With a variable bandwidth procedure, the B_e varies, and therefore the standard error must be computed for each value of the PSD. Note that this expression works only for averaging power above 0 Hz. If an estimate is made at zero, care must be taken not to count components twice in determining the number of basic estimates averaged to make the final estimate.

Another way of discussing the error is through the use of the χ^2 distribution. As discussed in Ref. 9, if $z_1, z_2, \ldots, z_n$ are independent random variables with zero mean and unity variance, then the random variable ${\chi_n}^2$ defined by

$${\chi_n}^2 = \sum_{i=1}^{N} {z_i}^2 \tag{6.124}$$

is the χ^2 variable with n d.f. The number of d.f., n, is the number of independent or "free" squares entering into the expression. In a sense, it characterizes the amount of information available in the data. The probability density function for ${\chi_n}^2$ is

$$p({\chi_n}^2) = \frac{[(\chi^2)^{(n/2)-1} - 1]\; e^{-\chi^2_{n/2}}}{2^{n/2}\,\Gamma(n/2)}, \tag{6.125}$$

where

$$\Gamma(y) = \int_0^\infty e^{-x}\, x^{y-1}\, dx = (y-1)!$$

if y is a positive integer. The distribution is used to compute confidence bands on the PSD. If $\hat{G}(f)$ is an estimate of the PSD around the frequency f, then the confidence limits take the form

$$\text{Prob}\,[A < G(f) \leqslant B] = p. \tag{6.126}$$

The parameter p is a fixed probability, which commonly is either 0.80, 0.90, or 0.95. Equation (6.126) may be interpreted as, "with $100p$ confidence, the true value of $G(f)$ lies between A and B."

The value for p is chosen before tests are made. A related parameter a, where

$$a = 1 - p, \tag{6.127}$$

is sometimes also used instead of p. Having obtained $\hat{G}(f)$, the question arises as to how to compute the A and B parameters, known as the confidence limits. These may be shown [1] to be

$$A = \frac{n\hat{G}(f)}{\chi^2_{n;1-a/2}},$$

$$B = \frac{n\hat{G}(f)}{\chi^2_{n;a/2}}. \tag{6.128}$$

The $\chi^2_{n;a}$ parameter is based on the χ^2 distribution. Its definition is

$$\chi^2_{n;a} = \left[b \text{ such that } \int_b^\infty p(\chi_n{}^2)\, d\, \chi_n{}^2 = a\right]. \tag{6.129}$$

The number of degrees of freedom is twice the number of observations that appear within the bandwidth of the estimate. Thus,

$$n = 2B_e T$$

$$= \frac{2N}{m}, \tag{6.130}$$

where m is the number of autocorrelation lags if the correlation procedure is used. Adjustments might have to be made depending on the effective bandwidth

of the lag window. The bandwidth of the Parzen window is slightly greater, and Eq. (6.13) should be multiplied by 1.3.

6.6 Statistical Error and Planning

The preceding section introduced three important terms commonly used when discussing the variability of PSD's. They are

B_e = Bandwidth of analysis; usually taken to be the distance between the half-power points of whatever filter or window is used to view the PSD

ϵ = the normalized standard error of the PSD

n = the number of degrees of freedom.

Based on the assumptions that $\{x_i\}$ has a zero mean, is normally distributed, and each value is independent of the others, it was shown somewhat heuristically that the following relations hold

$$\epsilon = \sqrt{\frac{1}{B_e T}} = \sqrt{\frac{2}{n}},$$

$$n = 2B_e T,$$

where

$$T = N\Delta t. \tag{6.131}$$

If the PSD is computed using the correlation procedure, then the number of lags m is related to the above parameters in the following manner:

$$B_e = \frac{1}{m\Delta t},$$

$$n = \frac{2N}{m},$$

$$\epsilon = \sqrt{\frac{m}{N}}. \tag{6.132}$$

Some examples will help illustrate the interactions of these terms. The following ones were taken from Ref. 5.

Example 6.1 Suppose it is desired that $\epsilon = 0.10$, the maximum frequency (the folding frequency $f_n = 1/2\Delta t$) is 2000 Hz, and $m = 50$, i.e., the correlation procedure is employed. Compute values for Δt, B_e, N, and T.

$$\Delta t = \frac{1}{2f_n} = 0.25 \text{ msec},$$

$$B_e = \frac{1}{m\Delta t} = 80 \text{ Hz},$$

$$N = \frac{m}{\epsilon^2} = 5000,$$

$$T = N\Delta t = 1.25 \text{ seconds}. \tag{6.133}$$

Example 6.2 Suppose it is desired that $B_e = 20$ Hz when $f = 1000$ Hz and $\epsilon = 0.10$. Compute values for Δt, m, N, and T.

$$\Delta t = \frac{1}{2f_n} = 0.50 \text{ msec},$$

$$m = \frac{1}{B_e\Delta t} = 100,$$

$$N = \frac{m}{\epsilon^2} = 10{,}000,$$

$$T_r = N\Delta t = 5.0 \text{ seconds}. \tag{6.134}$$

Example 6.3 Suppose that $B_e = 25$ Hz when $f_c = 500$ Hz and $T = 20$. Compute values for Δt, m, N, and ϵ.

$$\Delta t = \frac{1}{2f_n} = 1.0 \text{ msec},$$

$$m = \frac{1}{B_e\Delta t} = 40,$$

$$N = \frac{T}{\Delta t} = 20{,}000,$$

$$\epsilon = \sqrt{\frac{m}{N}} = 0.045. \tag{6.135}$$

The above examples all used the normalized standard error, ϵ. Confidence bands could also have been used. In that case,

$$n = 2B_eT = \begin{cases} 200 & \text{Example 6.1} \\ 200 & \text{Example 6.2} \\ 1000 & \text{Example 6.3.} \end{cases} \tag{6.136}$$

Table 6.3. Table of $\chi_{n;1-a/2}$

Degrees of Freedom	Exceeded by 90% of all Values	Exceeded by 50% of all Values	Exceeded by 10% of all Values
1	0.016	0.46	2.71
2	0.10	0.70	2.30
3	0.19	0.79	2.08
4	0.26	0.84	1.94
5	0.32	0.87	1.85
10	0.49	0.93	1.60
20	0.62	0.96	1.42
30	0.69	0.98	1.34
40	0.73	0.98	1.30
50	0.75	0.99	1.26
100	0.82	0.99	1.18
200	0.873	1.00	1.139
500	0.920	1.00	1.081
1000	0.943	1.00	1.057

Confidence bands for various values of n and levels of confidence are given in Table 6.3. A confidence interval should be selected before the test is made and the PSD computed.

CHAPTER 7
FREQUENCY RESPONSE FUNCTION AND COHERENCE FUNCTION COMPUTATIONS

7.1 Properties of Frequency Response Functions

A discussion of linear systems and the computational procedures necessary for obtaining estimates of them from measured time histories is presented in this chapter. The simplest case, which has a single input and a single output, is discussed first. A generalization to the multiple-input, single-output system is then presented. Computational requirements and procedures for confidence limit evaluations are given in the final section. The application of such techniques to shock and vibration analysis is quite broad. Many structures can be approximated to a useful degree of accuracy by ideal linear systems. The ideas of impedance and transmissibility as functions of frequency are derived directly as frequency response functions relating a specific type of input and output.

Consider a physically realizable linear system that does not have any time varying parameters. As discussed in Section 3.1, the weighting function $h(\tau)$ associated with this system is defined as the response (the output) function of the system to a unit impulse input function as a function of the time τ from the occurrence of the impulse. For physically realizable systems, it is necessary that $h(\tau) = 0$ for $\tau < 0$ since the response must follow the input. The weighting function concept is useful because, for an arbitrary input $x(t)$, the system output $y(t)$ is given by the convolution integral

$$y(t) = \int_0^\infty h(\tau)\, x(t-\tau)\, d\tau. \tag{7.1}$$

That is, the value of the output $y(t)$ at any time t is given as a weighted linear (infinite) sum over the entire past history of the input $x(t)$.

The linear system may alternatively be characterized by its frequency response or transfer function $H(f)$ which is defined as the Fourier transform of $h(\tau)$. That is,

$$H(f) = \int_0^\infty h(\tau)\, e^{-j2\pi f\tau}\, d\tau. \tag{7.2}$$

The lower limit is zero instead of $-\infty$ since $h(\tau) = 0$ for $\tau < 0$.

The idea of physical realizability is important from the standpoint of the engineering analysis of real systems. However, from a mathematical, and

sometimes also computational, viewpoint, unrealizable versions of Eqs. (7.1) and (7.2) are most useful. Instead of a finite lower limit, $-\infty$ is used. Thus,

$$y(t) = \int_{-\infty}^{\infty} h(\tau)\, x(t-\tau)\, d\tau, \tag{7.3}$$

$$H(f) = \int_{-\infty}^{\infty} h(\tau)\, e^{-j2\pi f\tau}\, d\tau. \tag{7.4}$$

For example, numerical filters used in a digital computer need not be realizable in this sense. It is perfectly correct to use symmetrical weighting functions and have phaseless (zero phase shift) filters (see Chapter 3). The frequency response function relates the input and output variables by the formula

$$Y(f) = H(f)\, X(f). \tag{7.5}$$

This is obtained by taking Fourier transforms of both sides of Eq. (7.3). The frequency response function is of great interest because it contains both amplitude magnification and phase shift information. Since $H(f)$ is complex valued, the complex exponential (polar) notation may be used. That is,

$$H(f) = |H(f)|\, e^{-j\phi(f)}, \tag{7.6}$$

where the absolute value $|H(f)|$ is the *gain factor* and the associated phase angle $\phi(f)$ is the *phase factor.*

7.2 Relationships for Single-Input Linear Systems

Assume that a linear system with a clearly defined, single input and single output is subjected to a random input x, which is a representative member from a stationary random process with a zero mean value. Then, the output y will have the same properties as shown in Ref. 5. Two relations between the ordinary one-sided power and the cross spectral density functions $G_x(f)$, $G_y(f)$, and $G_{xy}(f)$, defined for $f \geq 0$, are

$$G_y(f) = |H(f)|^2\, G_x(f), \tag{7.7}$$

$$G_{xy}(f) = H(f)\, G_x(f). \tag{7.8}$$

Therefore, with knowledge of the input power spectrum and the spectrum cross power, the frequency response function for a linear system is completely determined including both gain factor and phase factor.

The coherence function $\gamma_{xy}^2(f)$ is a real-valued quantity defined as

$$\gamma_{xy}^2(f) = \frac{|G_{xy}(f)|^2}{G_x(f)\, G_y(f)}. \tag{7.9}$$

The power cross spectral density function $G_{xy}(f)$ may be shown to satisfy the inequality

$$|G_{xy}(f)|^2 \leq G_x(f)\, G_y(f), \tag{7.10}$$

which implies that

$$0 \leq \gamma_{xy}^2(f) \leq 1. \tag{7.11}$$

Now, consider the measurement of the power spectral density for linear systems. For this case, $\gamma_{xy}^2 = 1$. Hence the coherence function attains a theoretical maximum of unity at all frequencies for the case of linear systems. If the coherence function is less than unity, one possible cause may be the lack of complete linear dependence between the input and the output for the system in question.

Given discrete time histories x_i and y_i, $i = 0, 1, \ldots, N-1$, Eqs. (7.8) and (7.9) are directly applied to compute transfer and coherence function estimates. Complex quantities must be manipulated, which is the major difference when compared with previous computations. Smoothed estimates of the power spectra $\hat{G}_{xk}$, $\hat{G}_{yk}$ and the cross spectrum $\hat{G}_{xyk}$ must first be obtained. The techniques of Chapter 6 would be applied to obtain these estimates. Then if complex arithmetical operations are available in the computer software, the transfer function is obtained directly from Eq. (7.8);

$$\hat{H}_{xyk} = \frac{\hat{G}_{xyk}}{\hat{G}_{xk}}, \qquad k = 0, 1, \ldots, m. \tag{7.12}$$

The real and imaginary parts, the co- and quadspectrum, can be used in the computations individually if complex arithmetical operations are not available.

$$\text{Re}\,[\hat{H}_{xyk}] = \hat{C}_{xyk}/\hat{G}_{xk}, \tag{7.13a}$$

$$\text{Im}\,[\hat{H}_{xyk}] = \hat{Q}_{xyk}/\hat{G}_{xk}, \qquad k = 0, 1, \ldots, m. \tag{7.13b}$$

The final transfer function output usually desired will be gain squared $|\hat{H}_{xyk}|^2$ and phase ϕ_{xyk};

$$|\hat{H}_{xyk}|^2 = \text{Re}^2\,[\hat{H}_{xyk}] + \text{Im}^2\,[\hat{H}_{xyk}], \tag{7.14}$$

$$\hat{\phi}_{xyk} = \frac{180}{\pi} \tan^{-1} \frac{\hat{Q}_{xyk}}{\hat{C}_{xyk}}, \qquad k = 0, 1, \ldots, m. \tag{7.15}$$

Many possible forms of plotted and printed outputs might be desired from the transfer function computations. For all of the forms given here, $f = k\Delta f$, $k = 0, 1, \ldots, m$:

1. Re $[\hat{H}_{xyk}]$ vs Im $[\hat{H}_{xyk}]$,
2. Re $[\hat{H}_{xyk}]$ vs f,
   Im $[\hat{H}_{xyk}]$ vs f,
3. $|\hat{H}_{xyk}|^2$ vs f,
   $\hat{\phi}_{xyk}$ vs f,
4. $\log |\hat{H}_{xyk}|$ vs $\log f$.

The ordinary coherence function is also obtained directly:

$$\hat{\gamma}^2_{xyk} = \frac{|\hat{G}_{xyk}|^2}{\hat{G}_{xk}\,\hat{G}_{yk}} = \frac{C^2_{xyk} + Q^2_{xyk}}{\hat{G}_{xk}\,\hat{G}_{yk}}, \qquad k = 0, 1, \ldots, m. \tag{7.16}$$

The use of FFT's to obtain spectra suggests the application of Eq. (7.5) for transfer function estimation. The transforms of the original data are available at an intermediate stage of the spectrum computations and thus $\tilde{H}(f)$ can be obtained from the relation

$$\tilde{H}_k = \frac{\tilde{Y}_k}{\tilde{X}_k}, \qquad k = 0, 1, \ldots, m. \tag{7.17}$$

It would seem that the coherence function could be obtained directly also.

$$\tilde{\gamma}^2_{xyk} = \frac{|\tilde{X}^*_k \tilde{Y}_k|^2}{|\tilde{X}_k|^2\,|\tilde{Y}_k|^2}, \quad k = 0, 1, \ldots, m. \tag{7.18}$$

Problems arise, however, since a result of unity is always obtained, as shown by

$$\begin{aligned} \tilde{\gamma}^2_{xyk} &= \frac{[\tilde{X}^*_k \tilde{Y}_k]\,[\tilde{X}^*_k \tilde{Y}_k]^*}{\tilde{X}^*_k \tilde{X}_k \tilde{Y}^*_k \tilde{Y}_k} \\ &= \frac{\tilde{X}^*_k \tilde{Y}_k \tilde{X}_k \tilde{Y}^*_k}{\tilde{X}^*_k \tilde{X}_k \tilde{Y}^*_k \tilde{Y}_k} = 1, \quad k = 0, 1, \ldots, m. \end{aligned} \tag{7.19}$$

When the time histories are considered as samples of random processes, it reflects the fact that sample coherence is a highly biased estimator of true coherence for small d.f. The d.f. in $\widetilde{\gamma}^2_{xyk}$ computed by Eq. (7.19) is $n = 2$. Thus, one must obtain smoothed spectra of proper statistical variability before taking ratios to obtain coherence estimates.

The result regarding coherence functions tends to cast doubt on the use of Eq. (7.17) for transfer function estimates when x_i and y_i are samples of a random process. Clearly, it is proper to use the formula when x_i and y_i are deterministic data passed through a truly linear system. There is considerable question as to what is a proper estimation procedure for random data. At least three possibilities exist:

1. Compute $\widetilde{Y}_k$ and $\widetilde{X}_k$. Then compute $\widetilde{H}_{xyk}$ from Eq. (7.17). Finally, smooth the real and imaginary parts of $\widetilde{H}_{xyk}$ individually to obtain an estimate $\hat{H}_{xyk}$ of appropriate statistical reliability.
2. Compute $\widetilde{Y}_k$ and $\widetilde{X}_k$. Smooth $\widetilde{Y}_k$ and $\widetilde{X}_k$ to obtain $\hat{Y}_k$ and $\hat{X}_k$ of appropriate statistical reliability. Compute $\hat{H}_{xyk}$ from Eq. (7.17).
3. Compute $\widetilde{Y}_k$ and $\widetilde{X}_k$. Compute the smoothed $\hat{G}_{xyk}$ and $\hat{G}_{xk}$ of appropriate statistical reliability. Compute $\hat{H}_{xyk}$ from Eq. (7.12).

All of these procedures can give different results since the linear smoothing operation, the nonlinear operations of absolute value squared, and division are not commutative.

The first procedure can usually be eliminated from consideration for random data. More precisely, when low coherence exists because of extraneous noise in the output, then the first procedure must not be used. Nonlinearities in a system cannot be distinguished from extraneous noise in the output as far as the computation procedures are concerned. To prove the above statement, consider the following:

$$\left|\widetilde{H}_{xyk}\right|^2 = \frac{\left|\widetilde{Y}_k\right|^2}{\left|\widetilde{X}_k\right|^2} = \frac{\frac{1}{T}\left|\widetilde{Y}_k\right|^2}{\frac{1}{T}\left|\widetilde{X}_k\right|^2} = \frac{\widetilde{G}_{xk}}{\widetilde{G}_{yk}}. \tag{7.20}$$

Thus, the absolute value squared obtained from the ratio of raw transforms is equivalent to that obtained from the ratio of raw (unsmoothed) power spectra. When extraneous noise in the output is of significant magnitude, the ratios of power spectra can be shown (see Ref. 5) to give highly biased results for gain factor estimates.

A different type of bias, caused by smearing peaks in spectra, can also exist. It occurs when too wide a resolution bandwidth is employed. When an analysis bandwidth sufficiently narrow is used, additional resolution will not provide any additional help. However, it is conceivable that the smearing bias from using an excessively wide analysis bandwidth is of greater magnitude than the bias from low coherence when estimates are computed (Eq. (7.20) is used).

The recommended estimation procedure for input/output system data that recurs most typically is to use Eq. (7.12) or Eq. (7.17), in which some type of smoothing has been done on the raw Fourier transforms. Ratios of raw Fourier transforms should be avoided except when deterministic data are involved. Another exceptional case would be that of random data passed through a perfect linear system such as a linear numerical filter. The frequency response characteristics of a numerical filter can be conveniently evaluated by generating pseudo-random noise and operating on it with the numerical filter. Any of the frequency response estimation methods will give suitable results.

7.3 Relationships for Multiple-Input Linear Systems

A model of a linear system responding to multiple inputs will now be considered. It will be assumed that p inputs exist, and a single output is measured. Three types of coherence functions play an important role in this analysis, and their evaluation is discussed.

Consider a constant-parameter, linear system with p inputs $x_\ell(t)$, $\ell = 1, 2, \ldots, p$, and one measured output $y(t)$. The assumption is made that the output may be considered as the sum of the p individual output components $y_\ell(t)$, $\ell = 1, 2, \ldots, p$. That is,

$$y(t) = \sum_{\ell=1}^{p} y_\ell(t), \tag{7.21}$$

where $y_\ell(t)$ is defined as that part of the output produced by the ith input, $x_\ell(t)$, when all the other inputs are zero (see Fig. 7.1). The function $h_{\ell y}$ is

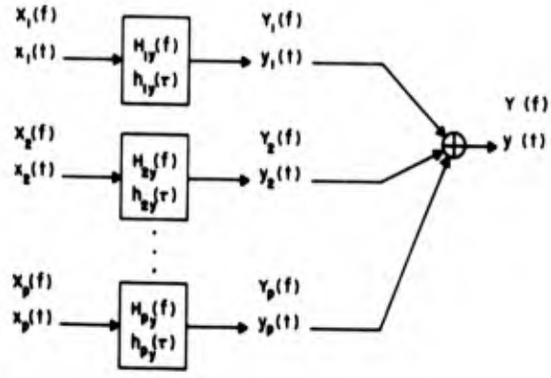


Fig. 7.1. Multiple-input linear system.

defined as the weighting function associated with the linear system between the input $x_\ell(t)$ and the partial output $y_\ell(t)$. Hence, $y_\ell(t)$ is given as follows:

$$y_\ell(t) = \int_{-\infty}^{\infty} h_{\ell y}(\tau)\, x_\ell(t-\tau)\, d\tau. \tag{7.22}$$

The Fourier transform of Eq. (7.22) gives

$$Y_\ell(f) = H_{\ell y}(f)\, X_\ell(f), \tag{7.23}$$

where $Y_\ell(f)$ and $X_\ell(f)$ are the Fourier transforms of $y_\ell(t)$ and $x_\ell(t)$, respectively. Then the Fourier transform $Y(f)$ for the total output is

$$Y(f) = \sum_{\ell=1}^{p} Y_\ell(f) = \sum_{\ell=1}^{p} H_{\ell y}(f)\, X_\ell(f). \tag{7.24}$$

The preceding relations can be expressed more concisely in matrix notation, and many results become more readily apparent. First, define a p-dimensional input vector

$$\mathbf{x}(t) = [x_1(t), x_2(t), \ldots, x_p(t)]'. \tag{7.25}$$

Also define a p-dimensional frequency response function vector

$$\mathbf{H}(f) = [H_{1y}(f), H_{2y}(f), \ldots, H_{py}(f)]'. \tag{7.26}$$

Next, define a p-dimensional cross power spectrum vector of the output $y(t)$ with the inputs $x_\ell(t)$,

$$\mathbf{G}_{xy}(f) = [G_{1y}(f), G_{2y}(f), \ldots, G_{py}(f)]', \tag{7.27}$$

where

$$G_{\ell y}(f) \equiv G_{x_\ell y}(f), \qquad \ell = 1, 2, \ldots, p. \tag{7.28}$$

Finally, define the $p \times p$ matrix of the power and cross spectra of all the inputs $x_\ell(t)$ by

$$\mathbf{G}_{xx}(f) = \begin{bmatrix} G_{11}(f) & G_{12}(f) & \cdots & G_{1p}(f) \\ G_{21}(f) & G_{22}(f) & \cdots & G_{2p}(f) \\ \vdots & \vdots & & \vdots \\ G_{p1}(f) & G_{p2}(f) & \cdots & G_{pp}(f) \end{bmatrix}, \tag{7.29}$$

where

$$G_{ij}(f) \equiv G_{x_i x_j}(f), \qquad i, j = 1, 2, \ldots, p. \tag{7.30}$$

The matrix $\mathbf{G}_{xx}(f)$ is Hermitian since it equals its conjugate transpose. This implies, for example, that the eigenvalues of $\mathbf{G}_{xx}(f)$ are real numbers, should these parameters be of interest in an application.

The system of linear equations to obtain a least squares solution for the $\mathbf{H}_{\ell y}(f)$ of Eq. (7.23) is the matrix equation

$$\mathbf{G}_{xy}(f) = \mathbf{G}_{xx}(f)\, \mathbf{H}(f). \tag{7.31}$$

This is equivalent to

$$\begin{bmatrix} G_{1y}(f) \\ G_{2y}(f) \\ \vdots \\ G_{py}(f) \end{bmatrix} = \begin{bmatrix} G_{11}(f) & G_{12}(f) & \ldots & G_{1p}(f) \\ G_{21}(f) & G_{22}(f) & \ldots & G_{2p}(f) \\ \vdots & \vdots & & \vdots \\ G_{p1}(f) & G_{p2}(f) & \ldots & G_{pp}(f) \end{bmatrix} \begin{bmatrix} H_{1y}(f) \\ H_{2y}(f) \\ \vdots \\ H_{py}(f) \end{bmatrix}. \tag{7.32}$$

The solution to this system of equations is

$$\mathbf{H}(f) = \mathbf{G}_{xx}^{-1}(f)\, \mathbf{G}_{xy}(f). \tag{7.33}$$

The computational procedures necessary for these operations subdivide into three groups:

1. Power and cross spectral density function computational routines.
2. A procedure for simultaneously handling $p + 1$ variables to efficiently obtain the spectral density functions among all these variables.
3. The complex variable arithmetical and matrix operations to compute the multidimensional linear system parameters.

The spectral density functions necessary can be generated by either fast Fourier transform procedures or correlation PSD procedures. The main requirement is that all possible combinations of cross spectra are computed. Because of the Hermitian symmetry, only the diagonal and the upper right portion of the matrix need be calculated. That is,

$$G_{ij}(f) = G_{ji}^{*}(f) = G_{ij}(-f). \tag{7.34}$$

The computational procedures described here are for computing parameters of a mathematical model, assuming a p-input $[x_1(t), i = 1, 2, \ldots, p]$ and single-output $[y(t)]$ linear system. The system parameters to be computed are

1. Frequency response functions between each of the inputs and the output
2. Ordinary coherence functions between all pairs of variables
3. The multiple coherence function between the output and all of the inputs
4. Partial (conditional) coherence functions between each input and the output while conditioning on the other inputs.

The first operation that must be performed is a sorting procedure. The spectral density functions are normally computed as a function of frequency. A program for multiple-input linear system analysis eventually must operate on the $(p+1) \times (p+1)$ spectral density matrices, one matrix for each frequency value.

The data operated on by the program is a set of spectral density matrices at frequencies indexed by k as follows:

$$\mathbf{G}_{yxk} = \begin{bmatrix} G_{yyk} & G_{y1k} & G_{y2k} & \cdots & G_{yqk} \\ G_{1yk} & G_{11k} & G_{12k} & \cdots & G_{1qk} \\ \vdots & \vdots & \vdots & & \\ G_{qyk} & G_{q1k} & G_{q2k} & \cdots & G_{qqk} \end{bmatrix},$$

$$k = 0, 1, \ldots, m. \tag{7.35}$$

The frequency index k will usually represent special frequency values

$$f_k = \frac{kf_n}{m}, \qquad k = 0, 1, \ldots, m, \tag{7.36}$$

where f_n is the Nyquist cutoff frequency. More generally, k can represent the frequency values

$$f_r = f_1 + k\Delta f, \quad k = 0, 1, \ldots, m, \tag{7.37}$$

where

f_1 = beginning frequency,

Δf = frequency increment.

The m separate $(p+1) \times (p+1)$ spectral density matrices can be visualized in the three-dimensional form illustrated in Fig. 7.2. The initial computation of any spectral density function provides a single-element $(m+1)$ longitudinal column of the $(p+1) \times$

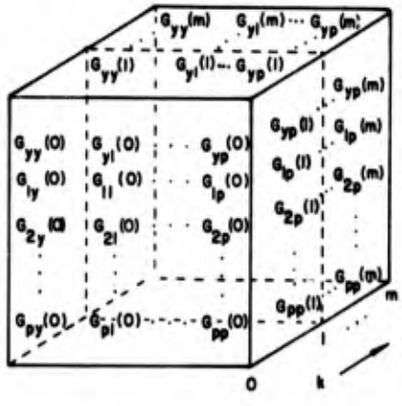


Fig. 7.2. Three-dimensional illustration of spectral matrices.

$(p + 1) \times (m + 1)$ block in Fig. 7.2. For the frequency response computations, a single slice of the block in the horizontal-vertical plane is needed. Logistical problems arise if magnetic tape is used for intermediate storage, since then the data must have been arranged in a serial fashion. Disk file storage is convenient for the necessary rearranging that has to take place. The p input variables and the output variable are assumed to be zero mean, stationary, Gaussian processes whenever any statistical distribution results are discussed. The functions $H_{\ell y}(f)$, $\ell = 1, 2, \ldots, p$, are the frequency response function (transfer function) characteristics of the linear systems through which the variables are passing to make up $y(t)$.

The variables $x_\ell(t)$, $\ell = 1, 2, \ldots, p$, and $y(t)$ are assumed to be discrete (digitized) sequences of N points each. The notation for the N discrete points may be

$$\begin{aligned} x_{1n} &= x_1(n\Delta t), \\ x_{2n} &= x_2(n\Delta t), \\ &\vdots \\ x_{pn} &= x_p(n\Delta t), \\ y_n &= y(n\Delta t), \qquad n = 1, 2, \ldots, N, \end{aligned} \tag{7.40}$$

where Δt is the sampling (digitizing) interval.

The matrix equation to be solved to determine the frequency response function is Eq. (7.32), where the function argument will be omitted for notational simplicity (e.g., G_{11} is written instead of G_{11k}). The matrix and vectors in Eq. (7.32) are complex valued and hence require complex arithmetical operations for correct manipulation. In particular,

$$G_{i\ell} = C_{i\ell} - jQ_{i\ell}, \tag{7.41}$$

where C_k and Q_k are the appropriate cospectral and quadspectral density functions at index value k.

The solution to Eq. (7.32) is

$$\begin{bmatrix} H_{1y} \\ H_{2y} \\ \vdots \\ H_{py} \end{bmatrix} = \begin{bmatrix} G_{11} & G_{12} & \cdots & G_{1q} \\ G_{21} & G_{22} & \cdots & G_{2q} \\ \vdots & \vdots & & \vdots \\ G_{p1} & G_{p2} & \cdots & G_{pp} \end{bmatrix}^{-1} \begin{bmatrix} G_{1y} \\ G_{2y} \\ \vdots \\ G_{py} \end{bmatrix}, \tag{7.42}$$

or in simpler notation,

$$\mathbf{H}_{xy} = \mathbf{G}_{xx}^{-1}\,\mathbf{G}_{xy}. \tag{7.43}$$

An individual frequency response function is given by

$$H_{iy} = \sum_{\ell=1}^{p} G^{i\ell}, \qquad i = 1, 2, \ldots, p. \tag{7.44}$$

In terms of real and imaginary parts,

$$H_{iy} = \mathrm{Re}(H_{iy}) + j\mathrm{Im}(H_{iy}) = \sum_{\ell=1}^{p} [(C^{i\ell}\,C_{\ell y} - Q^{i\ell}\,Q_{\ell y}) - j(C^{i\ell}\,Q_{\ell y} + Q^{i\ell}\,C_{\ell y})], \tag{7.45}$$

where

$$G^{i\ell} = C^{i\ell} - jQ^{i\ell}, \tag{7.46}$$

are elements of $\|G_{i\ell}\|^{-1} = \|G^{i\ell}\|$, as required in Eq. (7.42). Equation (7.45) is the computational form implemented unless a complex arithmetic package is available, in which case Eq. (7.44) is used.

The ordinary coherence functions between the output y and each input x_i are computed by

$$\gamma_{iy}^2 = \frac{|G_{iy}|^2}{G_{ii}\,G_{yy}} = \frac{C_{iy}^2 + Q_{iy}^2}{G_{ii}\,G_{yy}}, \qquad i = 1, 2, \ldots, p. \tag{7.47}$$

The multiple coherence function between the output y and *all* of the inputs $x_1, x_2, \ldots, x_p$ is computed by

$$\gamma_{y\cdot x}^2 = 1 - [G_{yy}\,G^{yy}]^{-1}, \tag{7.48}$$

where G^{yy} denotes the first diagonal element of the inverse matrix $\mathbf{G}_{yxx}^{-1}$ associated with $\mathbf{G}_{yxx}$ of Eq. (7.35).

Ordinary and multiple coherence functions for the set of inputs x_1 alone are defined by considering the $p \times p$ spectral matrix of the inputs $\mathbf{G}_{xx}$. The ordinary coherence function between any pair of inputs x_i and x_ℓ is computed by

$$\gamma_{ij}^2 = \frac{G_{i\ell}^2}{G_{ii}\,G_{\ell\ell}} = \frac{G_{i\ell}^2 + Q_{i\ell}^2}{G_{ii}\,G_{\ell\ell}}. \tag{7.49}$$

The multiple coherence function between x_i and all other inputs $x_1, x_2, \ldots, x_p$ excluding x_i, is computed by

$$\gamma^2_{i \cdot x} = 1 - [G_{ii}\, G^{ii}]^{-1},$$

where G^{ii} denotes the ith diagonal element of the inverse matrix $(\mathbf{G}_{xx})^{-1}$ associated with $\mathbf{G}_{xx}$ of Eq. (7.35). This quantity is convenient for understanding frequency response function confidence limits.

To obtain the partial coherence function between any input, say x_1, and the output conditioned on the remaining $(p-1)$ inputs, one partitions $\mathbf{G}_{yxx}$ as indicated below in Eq. (7.51);

$$\mathbf{G}_{yxx} = \left[\begin{array}{cc|ccc} G_{yy} & G_{y1} & G_{y2} & \cdots & G_{yp} \\ G_{1y} & G_{11} & G_{12} & \cdots & G_{1p} \\ \hline G_{2y} & G_{21} & G_{22} & \cdots & G_{2p} \\ \vdots & \vdots & \vdots & & \vdots \\ G_{py} & G_{p1} & G_{p2} & \cdots & G_{pp} \end{array}\right] = \left[\begin{array}{cc} \Sigma_{yy} & \Sigma_{y1} \\ \Sigma_{1y} & \Sigma_{11} \end{array}\right]. \tag{7.51}$$

Then compute the conditional spectral matrix;

$$\mathbf{G}_{xy|p} = \Sigma_{yy} - \Sigma_{y1}\,(\Sigma_{11})^{-1}\,\Sigma_{1y}. \tag{7.52}$$

This procedure requires the inversion of the $(p-1) \times (p-1)$ complex-valued matrix Σ_{11}. An individual element $G_{1y|p}$ of the 2×2 matrix $\mathbf{G}_{xy|p}$ can be written in terms of real and imaginary parts as

$$G_{1y|p} = C_{1y|p} - jQ_{1y|p}. \tag{7.53}$$

The partial coherence function between the input x_1 and the output y, conditioned on the other $(p-1)$ inputs, is now computed by

$$\gamma^2_{1y|p} = \frac{|G_{1y|p}|^2}{G_{11|p}\, G_{yy|p}} = \frac{C^2_{1y|p} + Q^2_{1y|p}}{G_{11|p}\, G_{yy|p}}. \tag{7.54}$$

Similar results apply for x_2 by interchanging x_2 with x_1, for x_3 by interchanging x_3 with x_2, etc.

In the special case of a single-input single-output linear system, all coherence functions are identical. This can be verified by examining Eq. (7.56) for partial

coherence and Eq. (7.48) for multiple coherence. Upon substituting values when $p = 1$, these equations will both reduce to Eq. (7.47).

The equations in terms of real and imaginary parts are as follows. Let the elements of Σ_{11}^{-1} be

$$G^{hi} = U^{hi} - jV^{hi}. \tag{7.55}$$

The real and imaginary parts of an individual element $G_{\ell k|p}$ of the 2 X 2 matrix $\mathbf{G}_{xy|p}$ are obtained in two additional steps. First define the intermediate quantity

$$C'_{ik} - jQ'_{ik} = \sum_{h=2}^{p} [(U^{ih}C_{hk} - V^{ih}Q_{hk}) - j(U^{ih}Q_{hk} + V^{ih}C_{hk})],$$

$$j = 2, \ldots, p; k = y, 1. \tag{7.56}$$

The final equation is

$$G_{\ell k|p} = G_{\ell k} - \sum_{i=2}^{p} [(C_{\ell i}C_{ik} - Q_{\ell i}Q'_{ik}) - j(C_{\ell i}Q'_{ik} + Q_{\ell i}C'_{ik})],$$

$$\ell \equiv y, 1; k = y, 1. \tag{7.57}$$

These values are then inserted into Eq. (7.54) to obtain the partial coherence function.

An alternative formula for the multiple coherence function is obtained from similar formulas. First the matrix $\mathbf{G}_{yxx}$ is repartitioned.

$$\mathbf{G}_{yxx} = \left[\begin{array}{c|ccc} G_{yy} & G_{y1} & \cdots & G_{yp} \\ \hline G_{1y} & G_{11} & \cdots & G_{1p} \\ \vdots & \vdots & & \vdots \\ G_{py} & G_{p1} & & G_{pp} \end{array}\right] = \begin{bmatrix} \mathbf{G}_{yy} & \Sigma_{yx} \\ \Sigma_{xy} & \mathbf{G}_{xx} \end{bmatrix}. \tag{7.58}$$

The conditional (residual) output spectrum is computed from

$$G_{y|x} = G_{yy} - \Sigma_{yx}\mathbf{G}_{xx}^{-1}\Sigma_{xy}. \tag{7.59}$$

This formula is obtained in terms of real and imaginary parts as follows. From Eq. (7.45), let the real and imaginary parts of H_{iy} be denoted by

$$C''_{iy} = \mathrm{Re}\,[H_{iy}], \tag{7.60a}$$

$$Q''_{iy} = \mathrm{Im}\,[H_{iy}]. \tag{7.60b}$$

Finally,

$$G_{y|x} = G_{yy} - \sum_{i=1}^{p} [(C_{\cdot i}C''_{iy} - Q_{yi}Q''_{iy}) - j(C_{yi}Q''_{iy} + Q_{yi}C''_{iy})]. \tag{7.61}$$

Note that the result must be real. Thus, a convenient computational check formula is for the imaginary part of Eq. (7.62) to be zero.

The multiple coherence in terms of the conditional output spectrum is

$$\gamma^2_{y\cdot x} = 1 - \frac{G_{y|x}}{G_{yy}}. \tag{7.62}$$

This formula graphically illustrates an important interpretation of the multiple coherence function. The conditional or residual output power spectrum is the power remaining after all power that can be accounted for with linear filter relations is subtracted out. Therefore, $G_{y|x}/G_{yy}$ is the fraction of power not accounted for with linear relations and one minus this quantity is the fraction of power accounted for with linear relations. The multiple coherence function is the fraction of power in the output accounted for by simultaneous linear filter relationships with all the inputs.

7.4 Complex Matrix Inversion and Numerical Considerations

Since complex-valued quantities are involved, the requirement for inversion of a complex matrix exists. Two approaches may be followed:

1. Complex arithmetic can be used to invert a matrix of complex elements directly,

2. A special procedure can be employed to reduce the complex inversion to more inversions of real matrices.

There are practical advantages to both procedures. Many library routines can be found to invert real matrices. These routines will normally make checks for the condition and rank of the matrix. Also, procedures might be included in the routine to optimize the inversion as far as numerical accuracy is concerned. There are enough definite advantages to capitalize on the effort expended in developing this kind of a routine.

On the other hand, efficient complex matrix conversion routines are more difficult to come by. Thus, in some instances, it is more efficient to perform the inversion in terms of real and imaginary parts separately. On the other hand, the escalator method of matrix inversion is particularly neat for the spectral analysis of linear systems.

A method is presented in Ref. 35 for the inversion of a complex matrix. Write a given spectral matrix G in terms of its real and imaginary parts;

$$\mathbf{G} = \mathbf{C} - j\mathbf{Q}. \tag{7.63}$$

Then, the inverse of G is given by

$$\mathbf{G}^{-1} = \mathbf{C}_1 - j\mathbf{Q}_1, \tag{7.64}$$

where

$$\mathbf{C}_1 = (\mathbf{C} + \mathbf{Q}\mathbf{C}^{-1}\mathbf{Q}) \tag{7.65}$$

and

$$\mathbf{Q}_1 = -\mathbf{C}_1\mathbf{Q}\mathbf{C}^{-1}. \tag{7.66}$$

The computation steps are

1. Compute the $p \times p$ inverse matrix $\mathbf{C}^{-1}$,
2. Compute the matrix product $\mathbf{Q}\mathbf{C}^{-1}$ and save,
3. Compute the product $\mathbf{Q}\mathbf{C}^{-1}\mathbf{Q}$,
4. Compute the sum $\mathbf{C} + \mathbf{Q}\mathbf{C}^{-1}\mathbf{Q}$,
5. Invert the $p \times p$ matrix from Step 4 to obtain $\mathbf{C}_1 = (\mathbf{C} + \mathbf{Q}\mathbf{C}^{-1}\mathbf{Q})^{-1}$.

Thus, two real matrix inversions, three matrix multiplications, and one matrix summation are required to invert the $p \times p$ complex spectral matrix.

The escalator method is an iterative scheme whereby one proceeds from an inverse of order $(\ell - 1)$ to an inverse of order ℓ. The method has two advantages:

1. It is simple to take advantage of the complex conjugate symmetry of the spectral matrix, and

2. the three orders of inverse matrix, $p - 1$, p, and $p + 1$, are obtained in natural progression.

The method presented below is a special case of the one derived in Ref. 36. Partition a $p \times p$ spectral density matrix in the manner indicated;

$$\mathbf{G} = \begin{bmatrix} \mathbf{G}_{yy} & \mathbf{G}_{xy} \\ \\ \mathbf{G}_{yx} & \mathbf{G}_{xx} \end{bmatrix}, \tag{7.67}$$

where G_{yy} is a scalar, $\mathbf{G}_{xx}$ is $(p-1) \times (p-1)$, and $\mathbf{G}_{xy}$ is $1 \times (p-1)$. Let the inverse be denoted as

$$\mathbf{G}^{-1} = \begin{bmatrix} S & \mathbf{T} \\ \mathbf{T}^* & \mathbf{U} \end{bmatrix}. \tag{7.68}$$

The two matrices may be multiplied and set equal to the identity matrix. The resulting equations may be solved to obtain the following formulas:

$$S = \frac{1}{G_{yy} - \mathbf{G}_{xy}\mathbf{G}_{xx}^{-1}\mathbf{G}_{yx}}, \tag{7.69}$$

$$\mathbf{T}^* = -\mathbf{G}_{xx}^{-1}\mathbf{G}_{yx}S, \tag{7.70}$$

$$\mathbf{U} = \mathbf{G}_{xx}^{-1} - \mathbf{G}_{xx}^{-1}\mathbf{G}_{yx}\mathbf{T}. \tag{7.71}$$

Note that $\mathbf{T}$ need not be explicitly computed. Thus, all the elements of a $p \times p$ inverse matrix have been determined with knowledge of the $(p-1) \times (p-1)$ inverse matrix $\mathbf{G}_{xx}^{-1}$ and additional matrix multiplication operations.

Equations (7.69), (7.70), and (7.71) are applied recursively beginning with the 1×1 inverse $1/G_{xx}$. Note that at the pth stage, the inverse matrix necessary for the solution of Eq. (7.43) is available. The multiple coherence function of the output with all p inputs may be obtained from Eq. (7.69), applied at the $(p+1)$ stage. Namely, since

$$\frac{1}{G^{yy}} = S^{-1} = G_{yy} - \mathbf{G}_{xy}\mathbf{G}_{xx}^{-1}\mathbf{G}_{yx}, \tag{7.72}$$

then

$$\gamma_{y\cdot x}^2 = 1 - \frac{S^{-1}}{G_{yy}}. \tag{7.73}$$

Also, at the $(p+1)$ stage, the frequency response vector is related to $\mathbf{T}^*$ of Eq. (7.70) by the equation

$$\mathbf{H}_{xy} = -\mathbf{T}^*\mathbf{S}^{-1}. \tag{7.74}$$

Thus, in computing the various quantities desired in the solution of the matrix frequency response problem, the $(p+1) \times (p+1)$ spectral density matrix is effectively inverted.

The partial coherence function between the output and each of the inputs may be obtained without explicitly employing the $(p-1) \times (p-1)$ inverse in Eq. (7.52). Once the $p \times p$ inverse is obtained, all of the subinverses are obtained. An alternate formula for an individual frequency response function (an individual element of the vector $\mathbf{H}_{xy}$) is

$$H_{xiy} \equiv H_{iy} = \frac{G_{xiy|p}}{G_{ii|p}}. \tag{7.75}$$

The partial coherence can therefore be written

$$\gamma^2_{iy|p} = |H_{iy}|^2 \frac{G_{ii|p}}{G_{yy|p}}. \tag{7.76}$$

The two additional quantities are available when the $p \times p$ inverse $\mathbf{G}^{-1}_{xx}$ is obtained. First, from Eq. (7.72),

$$G_{yy|p} = 1/G^{yy} = S^{-1}. \tag{7.77}$$

Also from Eq. (7.50), $G_{ii|p}$ is the reciprocal of the ith diagonal element in the inverse matrix $\mathbf{G}^{-1}_{xx}$;

$$G_{ii|p} = 1/G^{ii}. \tag{7.78}$$

The partial coherence functions can then be obtained from the formula

$$\gamma^2_{iy|p} = |H_{iy}|^2 \, S^{-1}/G^{ii}. \tag{7.79}$$

Thus, the escalator approach to the necessary matrix inverse unifies many of the formulas used for computing the various desired quantities.

7.5 Confidence Limit Computations

In addition to the basic parameter estimates, the confidence limits for the different coherence functions and for the frequency response functions can be computed. Confidence limits can easily be determined for gain and phase or real and imaginary parts. Computing the coherence function limits requires a single type of formula for all three types of functions. Only the degree-of-freedom parameter need be adjusted. A Gaussian approximation is used for the distribution of sample coherence, which is valid roughly in the range $0.3 \leqslant \gamma^2 \leqslant 0.98$ and for the d.f. parameter $n \geqslant 20$. The formula for the true distribution of sample coherence is not at all conveniently evaluated. See Ref. 37 for tables and the exact formula. Tables in this reference give the (cumulative) probability

distribution function for ordinary and multiple coherence functions as a function of the number of complex degrees of freedom, n_c, and the number of variables q. For a p-input, single-output system,

$$q = p + 1. \tag{7.80}$$

In terms of real degrees of freedom, $n = 2B_eT$, where B_e is the effective spectral resolution bandwidth and T is the effective record length,

$$n_c = \frac{n}{2} = B_eT. \tag{7.81}$$

For the special case of the single-input, single-output system, $q = 2$, and the tables apply to ordinary coherence functions.

A transformation that leads to an accurate normal (Gaussian) approximation for the distribution of sample coherence functions is shown in Ref. 38 to be given by

$$z = \tanh^{-1} \hat{\gamma} = \frac{1}{2} \ell n \left[\frac{1 + \hat{\gamma}}{1 - \hat{\gamma}}\right], \tag{7.82}$$

where $\hat{\gamma}$ is the positive square root of the sample coherence estimate $\hat{\gamma}^2$. This transformation is valid when $n > 20$ and when the true coherence is in the range $0.3 \leqslant \gamma^2 \leqslant 0.98$. The mean value and variance associated with z are approximated by

$$\mu_z = \tanh^{-1} \gamma + \left(\frac{p}{n - 2p}\right), \tag{7.83}$$

$$\sigma_z^2 = \frac{1}{n - 2p}, \tag{7.84}$$

where $n = 2n_c = 2B_eT$.

From the previous equations, for measured values of γ^2 and n, one can determine $(1 - a)$ confidence limits for the true value γ^2 by the following relation:

$$[\tanh (z - b - \sigma_z z_{a/2}) \leqslant \gamma \leqslant \tanh (z - b + \sigma_z z_{a/2})], \tag{7.85}$$

where z_a is the $100a$ percentage point of the normal distribution, and

$$b = \frac{p}{n - 2p}. \tag{7.86}$$

The above confidence limit formula applies either to ordinary coherence functions where $p = 1$ or to the multiple coherence functions where $p > 1$.

A simple adjustment can be made to obtain partial coherence function confidence limits. In general, one must reduce the number of degrees of freedom in the analysis by the number of conditional variables whose effects have been subtracted out. For example, in the case where the effects of $(p-1)$ inputs are subtracted out, one uses n' real degrees of freedom given by

$$n' = n - (p-1), \tag{7.87}$$

where $n = 2B_eT$.

The confidence bands for the frequency response functions $H_{iy}(f)$, $i = 1, 2, \ldots, p$, representing the model of Fig. 7.1, depend upon the sample coherence function between the output and the inputs, the sample multiple coherence function between the inputs, the sample input and power spectral density functions, and the sample frequency response functions. The basis for these results is discussed in Refs. 5 and 39.

Assume negligible bias error in the various spectral estimates involved. Let the true gain factor be $|H_{iy}|$ and the true phase factor be ϕ_{iy} so that

$$H_{iy} = |H_{iy}|\, e^{-j\phi_{iy}}. \tag{7.88}$$

Then the $(1-\alpha)$ confidence intervals for H_{iy} and ϕ_{iy} are given simultaneously at every i and at any specified frequency f (or digitally by the frequency index k)

$$\left.\begin{aligned} \hat{H}_{iy} - \hat{r}_i &\leq H_{iy} \leq \hat{H}_{iy} + r_i \\ \hat{\phi}_{iy} - \Delta\hat{\phi}_i &\leq \phi_{iy} \leq \hat{\phi}_{iy} + \Delta\hat{\phi}_i \end{aligned}\right\}, \quad i = 1, 2, \ldots, p, \tag{7.89}$$

where $\hat{H}_{iy}$ and $\hat{\phi}_{iy}$ are sample estimates. The square of the radial error, $\hat{r}_i^2 \equiv \hat{r}_i^2(f)$, and the phase error, $\Delta\hat{\phi}_i \equiv \Delta\hat{\phi}_i(f)$, are computed for each i by

$$\hat{r}_i^2 = \frac{2q}{n-2q}\left(F_{n_1,n_2;\alpha}\right)\frac{\left[1-\hat{\gamma}_{y\cdot x}^2\right]}{\left[1-\hat{\gamma}_{i\cdot x}^2\right]}\frac{\hat{G}_y}{\hat{G}_i}, \tag{7.90}$$

$$\Delta\hat{\phi}_i = \sin^{-1}\left(\frac{\hat{r}_i}{|\hat{H}_{iy}|}\right). \tag{7.91}$$

The various quantities in Eqs. (7.90) and (7.91) are

p = number of inputs (excluding output),

$n = 2B_eT$ = number of degrees of freedom in each spectral estimate,

$F_{n_1,n_2;\alpha}$ = 100α percentage point of an F distribution with $n_1 = 2p$ and $n_2 = 2n - 2p$ degrees of freedom,

$\hat{\gamma}^2_{y\cdot x}$ = sample estimate of the multiple coherence function between the output y and all the measured inputs, as defined by Eq. (7.63),

$\hat{\gamma}^2_{i\cdot x}$ = sample estimate of the multiple coherence function between the input x_i and the other measured inputs excluding x_i, as defined by Eq. (7.50),

$\hat{G}_y$ = power spectrum estimate for the output y,

$\hat{G}_i$ = power spectrum estimate for the input x_i.

A polar diagram for the confidence region represented by Eq. (7.89) is shown in Fig. 7.3 at the frequency f_0. Different confidence regions apply to each specified frequency f_0 and to each of the possible $i = 1, 2, \ldots, p$.

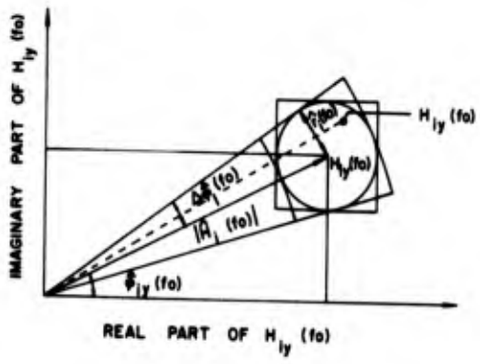


Fig. 7.3. Confidence diagram for multiple frequency response functions.

Equation (7.93) gives the radial error for the most general case of multiple coherent inputs. For special situations when the p inputs are coherent or there is only a single input $p = 1$, the square of the radial error takes the special form

$$\hat{r}_i^2 = \frac{2}{n-2}\,(F_{n_1,n_2;a})\,[1 - \hat{\gamma}^2_{yi}]\,\frac{\hat{G}_y}{\hat{G}_i}, \tag{7.92}$$

where n, G_y, and G_i are the same as before, while

$F_{n_1,n_2;a}$ = $100a$ percentage point of an F distribution with $n_1 = 2$ and $n_2 = n - 2$ degrees of freedom,

$\hat{\gamma}^2_{yi}$ = sample estimate of the ordinary coherence function between y and x_i, as defined by Eq. (7.16).

The phase error $\Delta\hat{\phi}_i$ is calculated as before by Eq. (7.91); however, the $\hat{r}_i$ obtained from Eq. (7.95) is used instead of Eq. (7.92).

The necessary F distribution values may be determined to a moderate degree of accuracy with an approximation from Ref. 40 (Section 26.6.1b, p. 947). If $p = (1 - a)$ is the confidence band desired, then

$$F_{n_1, n_2; a} = e^{2w}, \tag{7.93}$$

where

$$w = \frac{p(h + \lambda)^{1/2}}{h} - \left(\frac{1}{2b - 1} - \frac{1}{2a - 1}\right)\left(\lambda + \frac{5}{6} - \frac{2}{3h}\right), \tag{7.94}$$

$$h = 2\left(\frac{1}{2a - 1} + \frac{1}{2b - 1}\right)^{-1}, \tag{7.95}$$

$$\lambda = \frac{p^2 - 3}{6}, \tag{7.96}$$

$$n_1 = 2b, \; n_2 = 2a. \tag{7.97}$$

The quantity z_p is the pth percentile of the normal distribution function and can be evaluated by the series approximation in Appendix B. This approximation is inaccurate for small values of n_1. An empirically determined correction is

$$F'_{n_1, n_2; a} = F_{n_1, n_2; a} - \epsilon(n_1), \tag{7.98}$$

where

$$\epsilon(n_1) = 15.76003 - 16.260806 n_1 + 6.675521 n_1^2 - 1.3535354 n_1^3 + 0.1354177 n_1^4 - 0.0053333 n_1^5. \tag{7.99}$$

7.6 Flow Charts

A set of flow charts for the matrix frequency response function computations is given as Fig. 7.4. There are four stages.

1. Compute power and cross spectral density functions,
2. Rearrange into spectral density matrices,
3. Perform computations,
4. Rearrange into frequency functions for plotting.

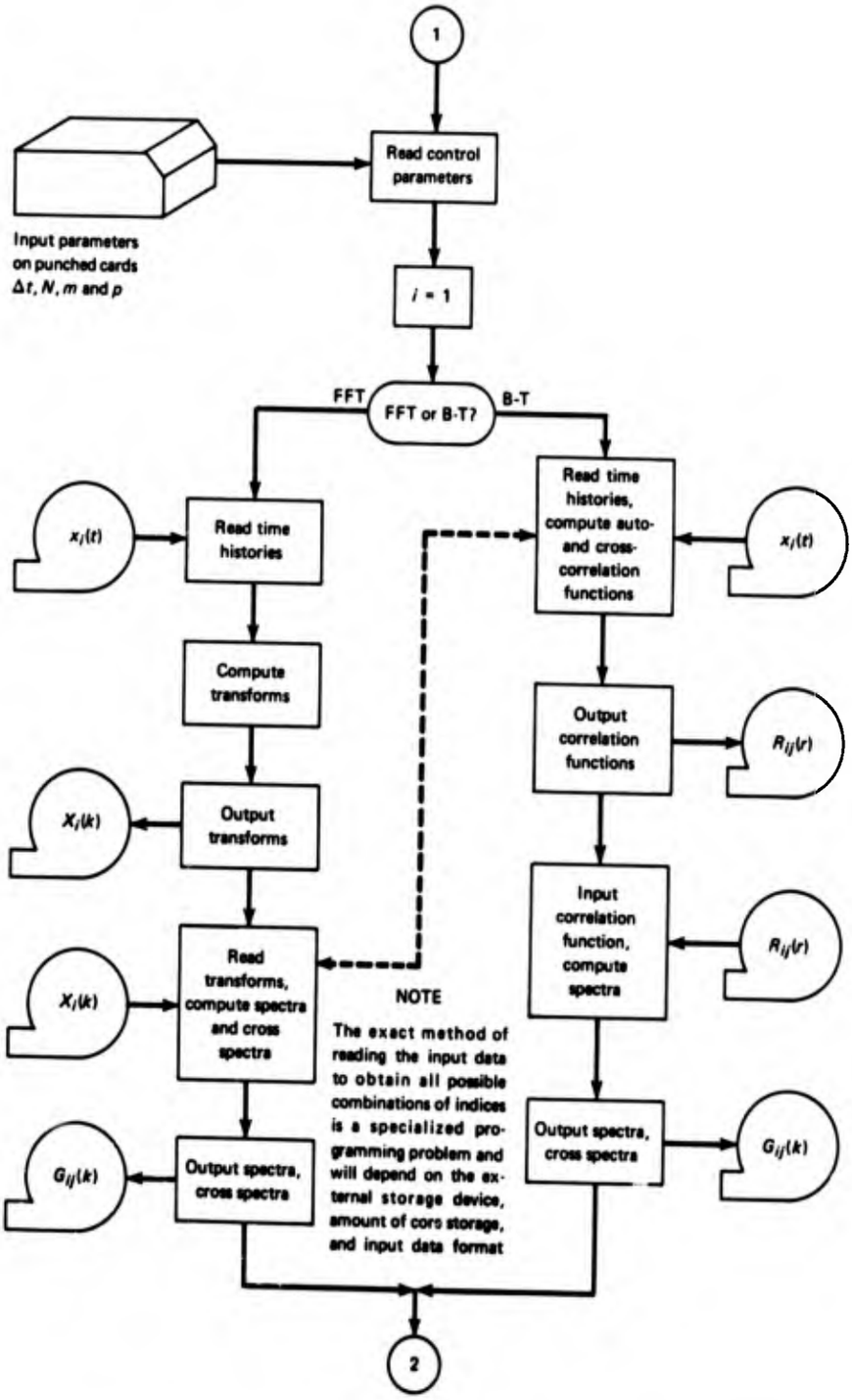


Fig. 7.4. Flow chart of computations for the matrix frequency response function.

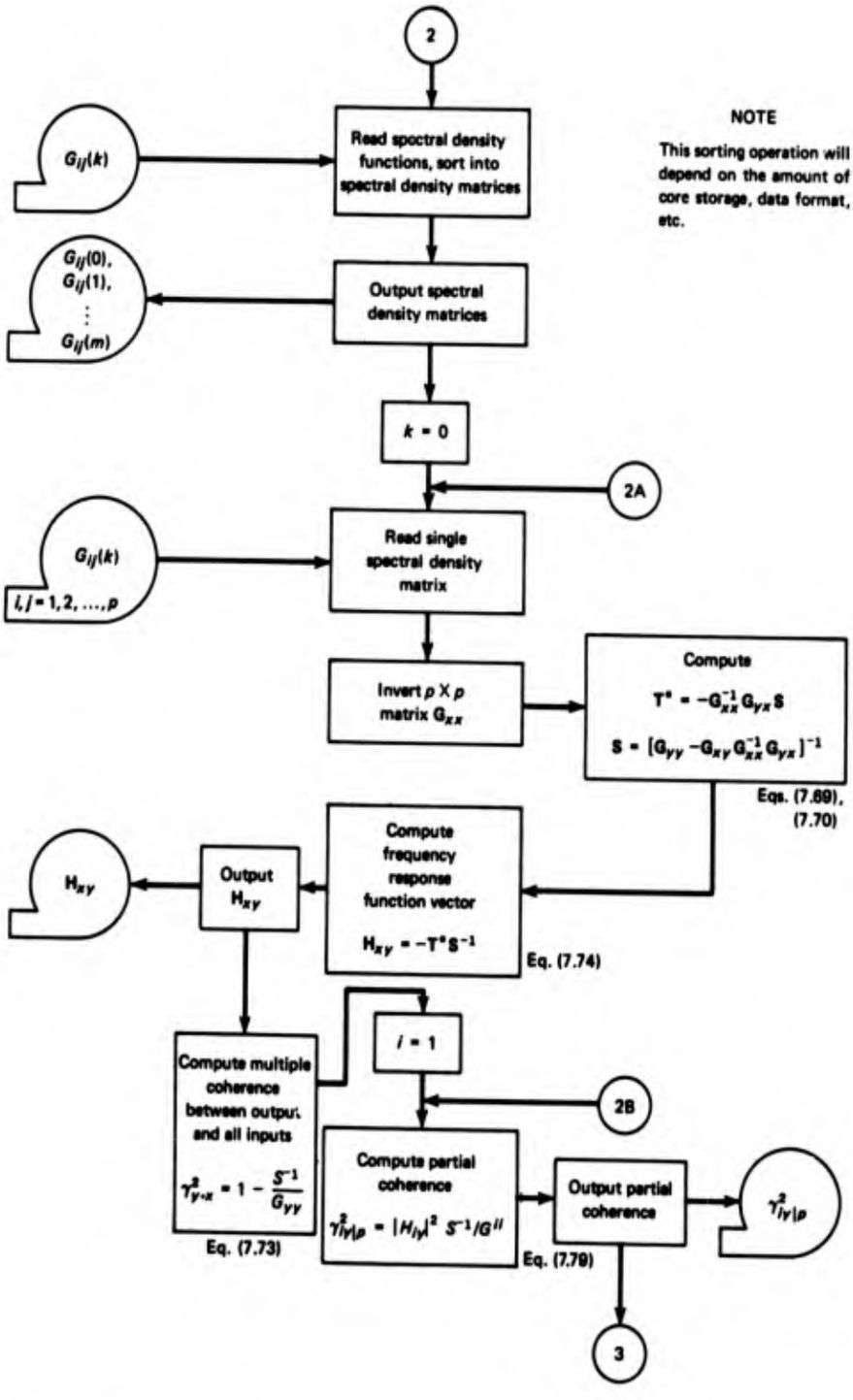


Fig. 7.4. Flow chart of computations for the matrix frequency response function.

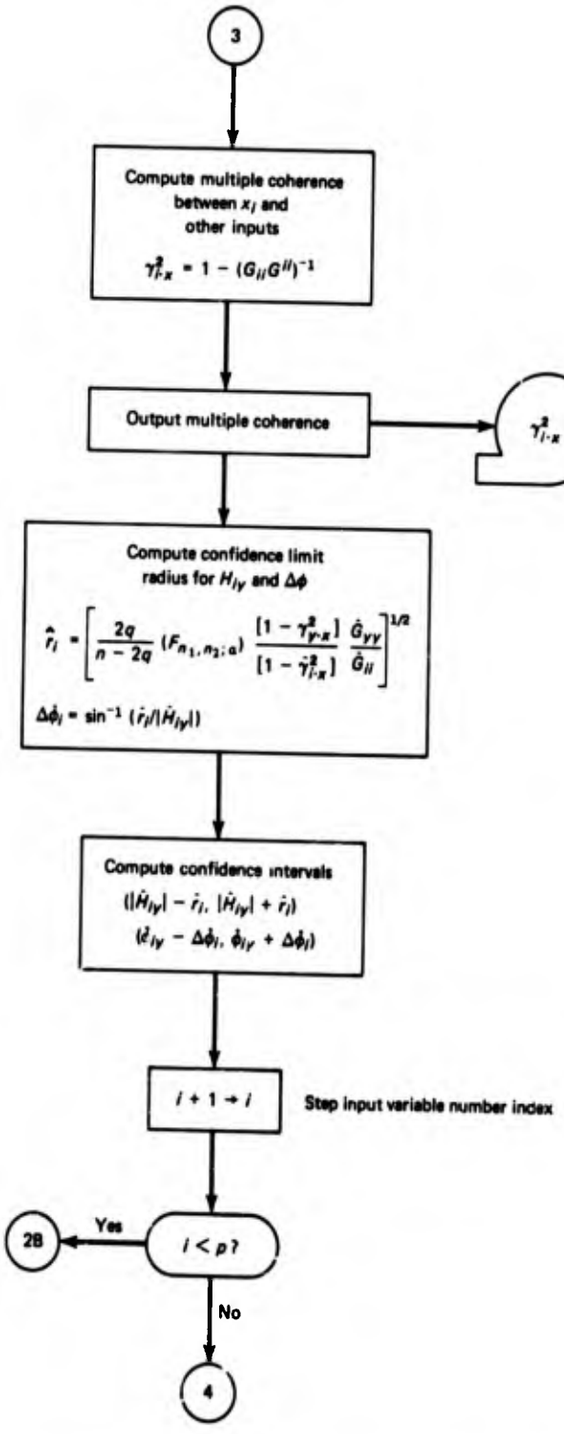


Fig. 7.4. Flow chart of computations for the matrix frequency response function.

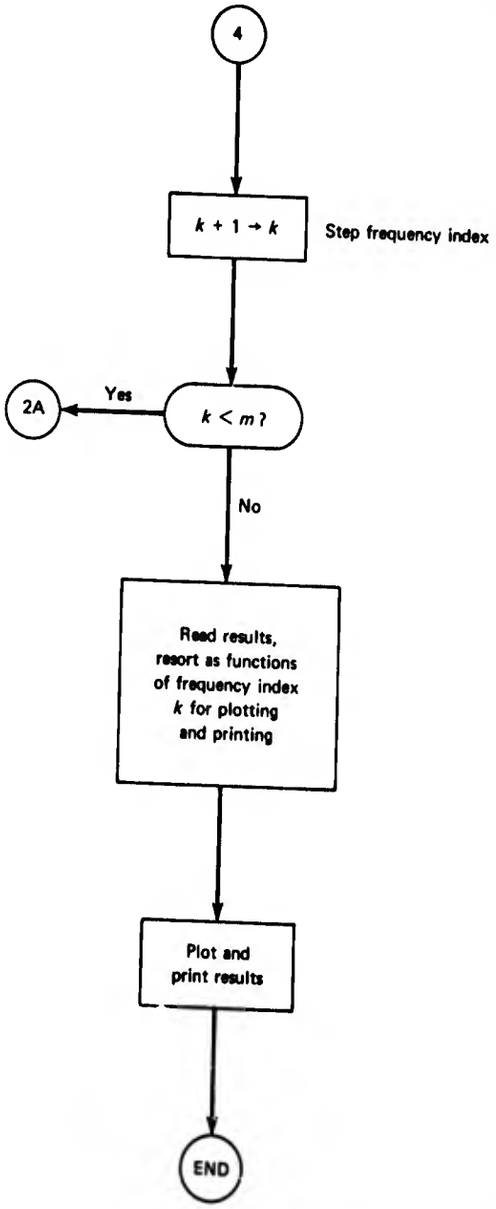


Fig. 7.4. Flow chart of computations for the matrix frequency response function.

CHAPTER 8
PROBABILITY DENSITY FUNCTION COMPUTATIONS

8.1 Review of Basic Statistical Terminology–Sample Density Function

Definitions of the mean and variance were given in Sections 1.2 and 1.3. This section will discuss these terms more carefully, but not in the detail that will be found in a work on statistics. The precise definitions of the above parameters are more involved than those given earlier, and the definition of a random process would require an elaborate mathematical scaffolding because of the variables which may be examined. Fortunately, in the applications considered here, the pathological functions considered by mathematicians do not occur. Hence, several simplifying assumptions can be made without impairing the validity of the results. These are

1. That the functions being considered are bounded and, before digitization, were continuous, and
2. That the random process underlying the function is ergodic and stationary.

The technical definition of a random variable basically requires that the variable be measurable. Assumption 1 guarantees this, as bounded, continuous functions form a subset of functions that are measurable. The remainder of this section will be spent in an intuitive discussion of Assumption 2.

Suppose that a large number of identical noise generators has been turned on at some remote time in the past and left to run. Associated with the output of all the generators is a function $f(x,t)$, the probability density function, with the following characteristics: For a certain time, say t_0, the probability that the output of the ith signal generator $x_i(t_0)$ lies between values a and b is given by the integral

$$\mathbf{P}\,[a \leqslant x_i(t_0) < b] = \int_a^b f(x,t_0)dx. \tag{8.1}$$

Note that the integration is performed with respect to the range of the random variable. The expected value of any function involving x, denoted by $\mathbf{E}\,[g(x)]$ where $g(x)$ is the expression whose expectation is sought, is defined to be

$$\mathbf{E}\{g\,[x(t_0)]\} = \int_{-\infty}^{\infty} g(x(t_0))f(x,t_0)dx. \tag{8.2}$$

In particular, the mean and variance are given by

$$\mu(t_0) = \int_{-\infty}^{\infty} x(t_0)\, f(x,t_0)dx; \tag{8.3}$$

$$\sigma^2(t_0) = \int_{-\infty}^{\infty} [x(t_0) - \mu(t_0)]^2\, f(x,t_0)dx. \tag{8.4}$$

If the random process is stationary, the parameters $\mu(t_0)$ and $\sigma^2(t_0)$ are independent of time. That is,

$$\mu(t_0) = \mu(t_1) \equiv \mu, \tag{8.5}$$

$$\sigma^2(t_0) = \sigma(t_1) \equiv \sigma^2, \tag{8.6}$$

where t_0 and t_1 are arbitrary. As stationarity is assumed, the mean and variance hereafter will be written without the qualifying t_0.

The assumption of ergodicity permits ensemble averages to be replaced with time averages. In the noise generator example, the generators were exactly alike, so that even though an individual generator were producing a different, unique random function, the output of any one of them would be sufficient to define the statistics for all. Thus, the expression for the mean in Eq. (8.3) may be replaced with

$$\mu = \lim_{T\to\infty} \frac{1}{2T}\int_{-T}^{T} x(t)dt, \tag{8.7}$$

which is the time average based on a single record of the process. A similar expression for the variance is

$$\sigma^2 = \lim_{T\to\infty} \frac{1}{2T}\int_{-T}^{T} [x(t) - \mu]^2\, dt. \tag{8.8}$$

Because only a sample of the random variable is taken, rather than a record length which is defined for infinite time, the sample mean and sample variance are denoted by symbols different from those used for the theoretical parameters. In particular, the sample mean for digital data is calculated from

$$m = \frac{1}{N}\sum_{i=1}^{N} x_i. \tag{8.9}$$

The unbiased* sample variance s^2 is obtained from

$$s^2 = \frac{1}{N-1} \sum_{i=1}^{N} (x_i - m)^2. \tag{8.10}$$

Criteria exist for determining the confidence intervals for the sample mean and sample variance. Detailed formulas and tables are to be found in Ref. 5. For the record lengths encountered in vibration analysis, the statistical variation of these two parameters is usually low and hence not a problem.

Sample probability density functions or histograms may also be obtained from the data. The sample density functions are not unique for a given data group as are m and s^2, but depend upon the values of certain parameters used to determine them. The histogram is computed in the following manner: An interval of the range of x, say $a \leqslant x \leqslant b$, is subdivided into k subintervals of equal length so that the entire range of x is broken up into $(k + 2)$ intervals. All of the data are examined, and the number of occurrences in each interval is tabulated. The histogram consists of a plot showing the number of occurrences for each of the intervals.

More formally, let $\{N_j\}$ be the set of integers obtained by counting the occurrences of $\{x_i\}$ in the jth interval. Let $c = (b - a)/k$, and $d_j = a + jc$. Then $\{N_j\}$ is defined by

j	N_j
0	[Number of x such that $x < a$]
⋮	⋮
j	[Number of x such that $d_{j-1} \leqslant x < d_j$]
⋮	⋮
k	[Number of x such that $d_{k-1} \leqslant x < b$]
$(k + 1)$	[Number of x such that $x \geqslant b$].

Figure 8.1 illustrates these quantities. The $\{N_j\}$ terms are frequently called pockets. One method of doing this sorting on a digital computer is to examine each x_i, $i = 1, \ldots, N$ in turn, making the following checks:

1. If $x_i < a$, add one to N_0,
2. If $x_i \geqslant b$, add one to N_{k+1},

*If the division of the expression for s^2 were $1/N$ rather than $1/(N - 1)$, then the expected value of s^2 would be $[(N - 1)/N]\ \sigma^2$, so that the result would be biased. For large N the error, however, would be small.

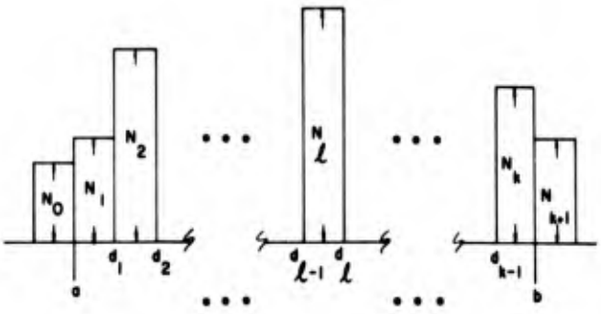


Fig. 8.1. Histogram construction.

3. If neither of the two preceding requirements are met, then $a \leqslant x_i \leqslant b$; therefore, compute

$$L = \frac{x_i - a}{c}, \qquad c = \frac{(b-a)}{k}.$$

Pick the largest integer ℓ such that $\ell \leqslant L$, then add one to N_ℓ. This technique turns out to be easy to implement on most digital computers.

Three forms of sequences based on the above are used. The first is the histogram, which is simply the sequence $\{N_j\}$ without change. The second sequence is $\{\mathrm{P}_j\}$ where

$$\mathrm{P}_j = \text{sample probability that } [d_{j-1} \leqslant x < d_j]$$

$$= N_j/N. \qquad j = 0, 1, \ldots, (k+1). \tag{8.12}$$

The third is the sample probability density function (PDF) which takes the form of the sequence $\{p_j\}, j = 0, \ldots, (k+1)$, where

$$p_j = \frac{N_j k}{N(b-a)}. \tag{8.13}$$

This can be interpreted as the derivative of the distribution function at the midpoint of each interval.

Before the above procedure can be effected, values for a, b, and k must be chosen. The question naturally arises, what is a reasonable criterion for the choice of these three parameters? There is no good single answer to this problem. Much of the choice must rest on assumptions about the underlying distribution being examined and the manner in which the data were collected. Data obtained using a system like the hypothetical DAS of Chapter 2 have two limitations imposed on them; namely, that the data are restricted in range, and

within that range there are only a finite number of possible levels. If the digitizer has only 128 levels, then it is clearly senseless to choose a $k > 128$, as some of the levels must be empty. Also, it is easy to visualize a situation in which the apportionment of ADC counts (or their converted equivalents) to the subintervals of the sample PDF would cause a biasing of answers.

Example 8.1 Suppose a digitizer with 16 levels is used and that the output is analyzed without any conversion to engineering units. Suppose further that each of the levels 0, ..., 15 is equally likely, so that the priority probability of the ith count occurring is 1/16. If k is taken to be 12 and a and b are 0 and 15 respectively, then the following distribution of counts would take place.

Pocket	Range	Levels Contained
1	0 – 1.25	0, 1
2	1.25 – 2.50	2
3	2.50 – 3.75	3
4	3.75 – 5.00	4
5	5.00 – 6.25	5, 6
6	6.25 – 7.50	7
7	7.50 – 8.75	8
8	8.75 – 10.00	9
9	10.00 – 11.25	10, 11
10	11.25 – 12.50	12
11	12.50 – 13.75	13
12	13.75 – 15.00	14

(8.14)

The expected contents of pockets 1, 5, and 9 will be twice as large as those of the other pockets, so that the sample PDF obtained would tend to be highly biased at those values, and thus, imply an incorrect result.

The above example shows that reasonable care should be exercised when setting up the calculations for the PDF and that the criteria employed should be scrutinized in order to avoid the pitfalls of biasing. One criterion for establishing the parameters arises from an attempt to determine the answer to a different but related problem; namely, that of deciding whether or not the data are Gaussian or normal in distribution.

The PDF for the normal distribution, denoted by ϕ, is given by the expression

$$\phi(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-(x-\mu)^2/(2\sigma^2)\right]. \tag{8.15}$$

The probability distribution function is the integral of the density function:

$$\text{Prob}\,[x \leq X] = \frac{1}{2\pi\sigma^2} \int_{-\infty}^{X} \exp\left[-(s-\mu)^2/(2\sigma^2)\right]\, ds$$

$$\text{Prob}\,[x \leqslant X] = \frac{1}{2\pi}\int_{-\infty}^{\frac{X-\mu}{\sigma}} \exp(-s^2/2)\,ds$$

$$= \Phi\left(\frac{X-\mu}{\sigma}\right). \tag{8.16}$$

Methods for computing this function are discussed in Appendix B. The probability that the variable lies between α and β is given by

$$P\,[\alpha \leqslant x < \beta] = \Phi\left(\frac{\beta-\mu}{\sigma}\right) - \Phi\left(\frac{\alpha-\mu}{\sigma}\right). \tag{8.17}$$

The normal distribution is assumed as a hypothesis in many analyses; and it arises naturally out of many theoretical calculations. It may therefore be desirable to see if indeed the collected data appear to be Gaussian. One procedure for making a check of the hypothesis is known as the chi-square goodness-of-fit test. The general procedure involves the use of the chi-square statistic as a measure of the discrepancy between an observed PDF and the theoretical density function. A hypothesis of equivalence is then tested by studying the sampling distribution of chi-squares. The number of occurrences that would be expected to fall within the ith class interval, if the data are Gaussian, is called the expected frequency in the class interval and will be denoted by F_j. The discrepancy between the observed frequency and expected frequency is $(N_j - F_j)$. To measure the total discrepancy, each interval must be used since

$$\sum_{j=0}^{(k+1)} N_j = \sum_{j=0}^{(k+1)} F_j = N. \tag{8.18}$$

The sum of the discrepancies must be zero. Note that F_j, in general, will not be an integer. The F_j are computed as follows:

$$F_0 = N\Phi\left(\frac{a-m}{s}\right)$$

$$\vdots$$

$$F_j = N\left\{\Phi\left(\frac{a+jc-m}{s}\right) - \Phi\left(\frac{a+c(j-1)-m}{s}\right)\right\}$$

$$F_{k+1} = N\left[1 - \Phi\left(\frac{b-m}{s}\right)\right].$$

The sample chi-square is obtained as follows:

$$X^2 = \sum_{j=0}^{(k+1)} \frac{(N_j - F_j)^2}{F_j}. \tag{8.20}$$

Under suitable assumptions, this sample chi-square may be compared with the theoretical chi-square distribution denoted by $\chi^2_{n;\alpha}$.

The distribution for χ^2, which was introduced in Section 6.5, is discussed in many references, such as Ref. 5. It depends upon the number of independent squared variables in χ^2 (the number of degrees of freedom, n). The value of n is equal to $(k + 2)$ if all pockets including the end ones are used, minus the number of different independent linear restrictions imposed on the observations. There is one such restriction because once the frequencies of the first $(k + 1)$ class intervals are known, the frequency in the last class interval is known as their sum in N. There are two additional restrictions caused by fitting the theoretical normal density function to the frequency histograms for the observed data. These arise from the fact that the sample mean and sample variance, rather than the true mean and variance, are used to calculate the $\{F_j\}$. The effect of this is to subtract another two d.f. from the data. Thus, if all $\{N_j\}$ are used, then

$$n = (k+2) - 3 = k - 1. \tag{8.21}$$

The value for n actually used may be smaller than this, as pockets for which $N < 2$ should be combined with other pockets. The details of this are described below.

Having established the proper d.f., n, for χ^2, a hypothesis test may be performed as follows. Let it be hypothesized that the variable x is normally distributed. After grouping the sampled observations into the $(k + 2)$ class intervals and computing F_j for each interval based on the sample mean and variance, compute X^2 as indicated in Eq. (8.20). Any deviation of the sample PDF from the normal distribution will cause X^2 to increase. The hypothesis that data are normally distributed is accepted if

$$X^2 \leq \chi^2_{n;\alpha}. \tag{8.22}$$

In this case, that acceptance is at the $(1 - \alpha)$ confidence level. If X^2 is greater than $\chi^2_{n;\alpha}$, the hypothesis is rejected at the α level of significance. Significance levels of 5, 10, and 20 percent (corresponding to confidence levels of 95, 90, and 80 percent) are commonly employed. The particular level selected is largely a matter of personal choice. The authors tend to favor α equal to 5 percent, given no additional information.

Based on the assumption that a chi-square goodness-of-fit test for normality is to be made, an expression for the number of class intervals for a given N has been derived [41]. This expression assumes that the data are uncorrelated and that $\alpha = 0.05$:

$$\text{Number of class intervals} = 1.87\,(N-1)^{2/5}. \tag{8.23}$$

This function is tabulated in Table 8.1. As stated earlier, as soon as the number of class intervals becomes comparable with the number of digitizer count levels, large biases may result.

A standard rule of thumb used by statisticians when applying the chi-square test is that every interval should have at least two occupancies. This requirement enables one to determine reasonable values for a and b. The end pockets have the smallest expected occupancy. Thus, the parameter a should satisfy the following equation:

$$2 = N\left\{\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\frac{a-m}{s}} e^{-t^2/2}\,dt\right\}. \tag{8.24}$$

This can be solved implicitly for a. After having found a value for a, the parameter b is simply

$$b = 2m - a. \tag{8.25}$$

The parameter k is given by

$$k = [\text{number of class intervals}] - 2. \tag{8.26}$$

After having establishing these three parameters, it is then possible to calculate the sample PDF and the expected normal occupancy. Before computing X^2, however, certain problems must be taken care of. It may turn out that there are

Table 8.1. Minimum Optimum Number (k) of Class Intervals for Sample Size N when $\alpha = 0.05$

N	k	N	k
200	16	20,000	94
400	20	40,000	129
600	24	70,000	162
800	27	100,000	187
1,000	30	200,000	247
1,500	35	400,000	326
2,000	39	700,000	407
4,000	57	1,000,000	470
7,000	65	1,140,000	500
10,000	74		

less than two occupancies in some pockets. In such a case, the contents of both the N_j and F_j pockets must be combined with adjacent ones. While easy to do visually, it is a little harder to implement on the computer. One scheme, which has been found to be satisfactory in most cases, is the following: Find the integer p such that $F_p > F_j$ for all j. Then define the sequences $\{Q_j\}$ and $\{R_j\}$ by the following procedure where all Q_j and R_j are zero to begin with.

$$\text{If } F_j \geq 2, \text{ then} \begin{cases} N_j + Q_j \to Q_j \\ \\ F_j + R_j \to R_j \end{cases}$$

$$\text{If } F_j < 2, \text{ then} \begin{cases} 0 \to Q_j, R_j \\ N_j + Q_{j+1} \to Q_{j+1} \\ F_j + R_{j+1} \to R_{j+1} \end{cases} \qquad (j \leq p)$$

$$\text{or} \qquad \begin{cases} 0 \to Q_j, R_j \\ N_j + Q_{j-1} \to Q_{j-1} \\ F_j + R_{j-1} \to R_{j-1} \end{cases}$$

$$(j > p). \qquad (8.27)$$

The sequences generated by this procedure are similar to $\{N_j\}$ and $\{F_j\}$ except that the tails have been shifted toward the center. Next, define the sequence $\{H_j\}$ by

$$H_j = \begin{cases} 1 & \text{if } Q_j \neq 0 \\ \\ 0 & \text{if } Q_j = 0. \end{cases} \qquad (8.28)$$

The quantities X^2 and n, the number of d.f., are thus given by

$$X^2 = \sum_{j=0}^{k+1} \frac{(Q_j - R_j)^2}{R_j} H_j;$$

$$n = \left(\sum_{j=0}^{k+1} H_j \right) - 3. \qquad (8.29)$$

The next step is to make the comparison to see if the data are to be accepted or rejected insofar as normality is concerned. As mentioned earlier, the usual procedure for comparing X^2 with $\chi^2_{n;a}$ is that of computing a', where a' is defined by

$$a' = \text{Prob}\,[X^2 > \chi^2_{n;a'}]. \tag{8.30}$$

The a' parameter is a function of χ^2 and n only. Having computed it, compare it with the preselected value a and conduct the test for normality on the following basis:

$$\begin{aligned} a' &< a \qquad \text{Reject} \\ a' &> a \qquad \text{Accept.} \end{aligned} \tag{8.31}$$

One method for computing a' is the following:

$$a' = 2\Phi(X) - 1 - 2\,e^{-X^2/2}\left[\sum_{r=1}^{(n-1)/2} \frac{X^{2r-1}}{1\cdot 3\cdot 5\ \ldots\ (2r-1)}\right] \tag{8.32a}$$

for n odd,

or

$$a' = 1 - e^{-X^2/2}\left[1 + \sum_{r=1}^{(n-2)/2} \frac{X^{2r}}{2\cdot 4\cdot 6\ \ldots\ (2r)}\right] \tag{8.32b}$$

for n even.

The computer program flow charts, Fig. 8.2, show one standard scheme for setting up a program. As a convenience they include the steps required to process some additional parameters not mentioned in the above discussion, such as the minimum and maximum values of x. The arithmetical expressions on the charts, e.g., $x_i^2 + x^2 \rightarrow x^2$, refer to both data values and storage locations. The above example would read, "The current value of x_i is squared and added to the contents of the storage location of the running total of x^2, and the results stored in that location."

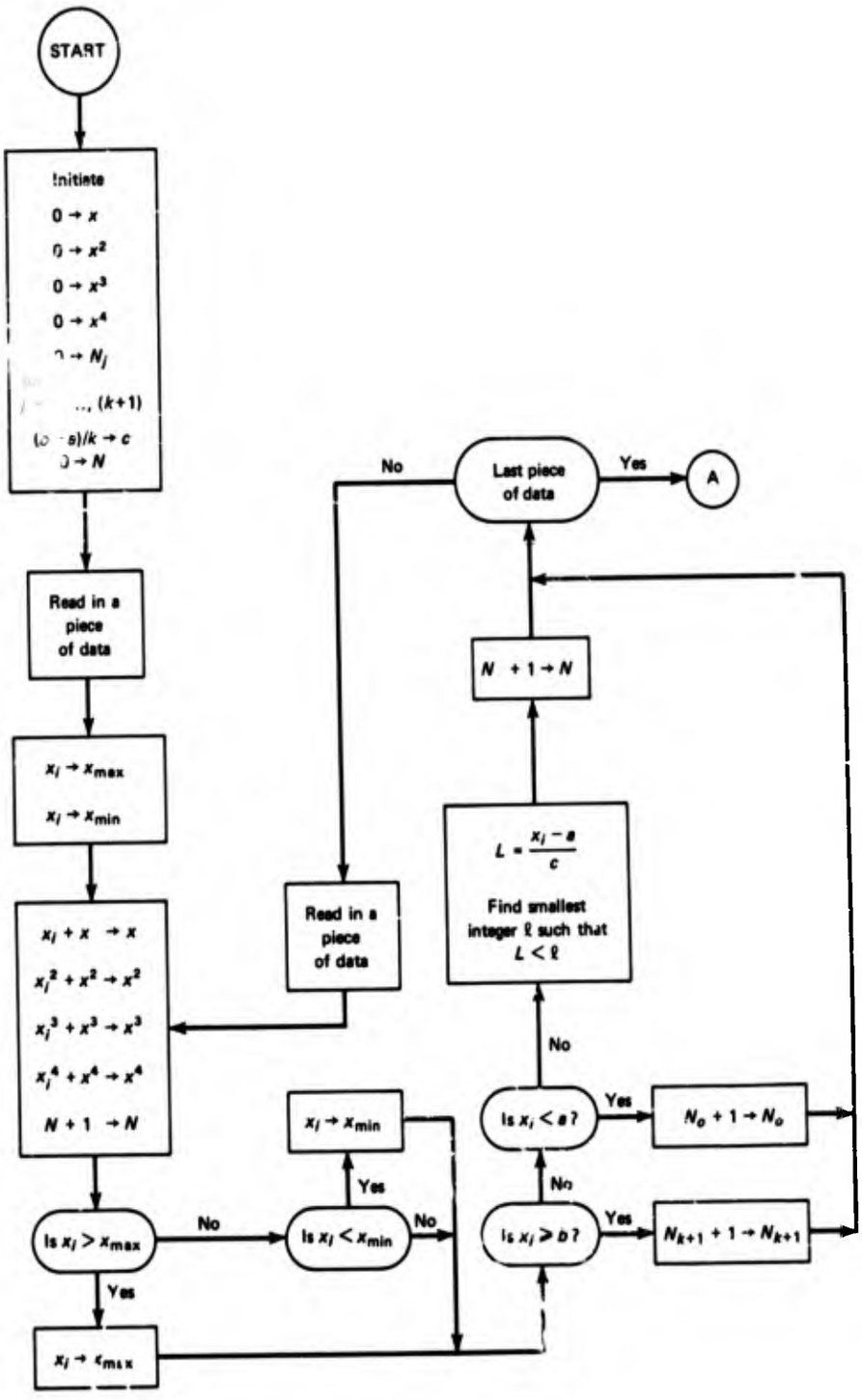


Fig. 8.2a. Processing, initial phase.

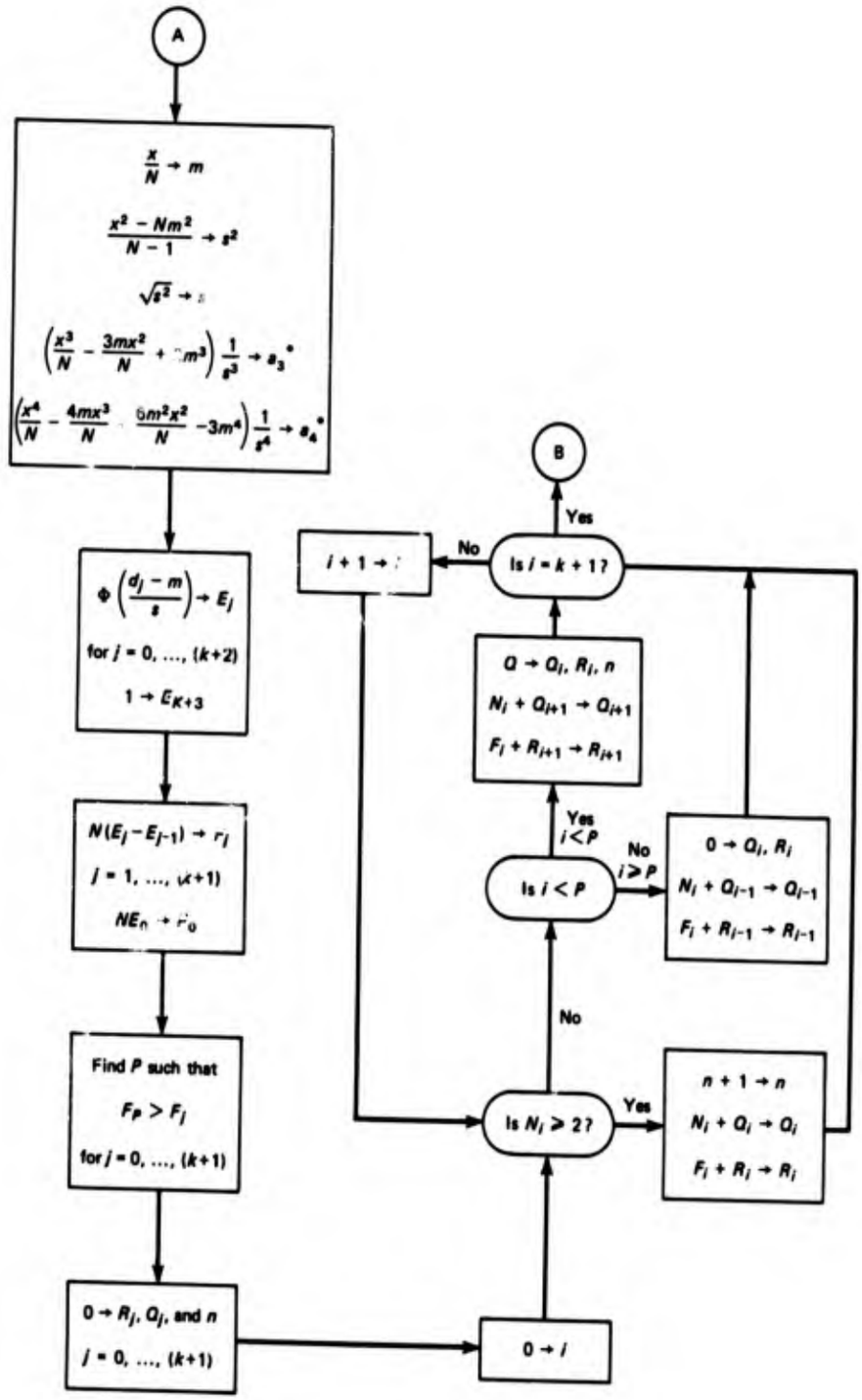


Fig. 8.2b. Processing 1.

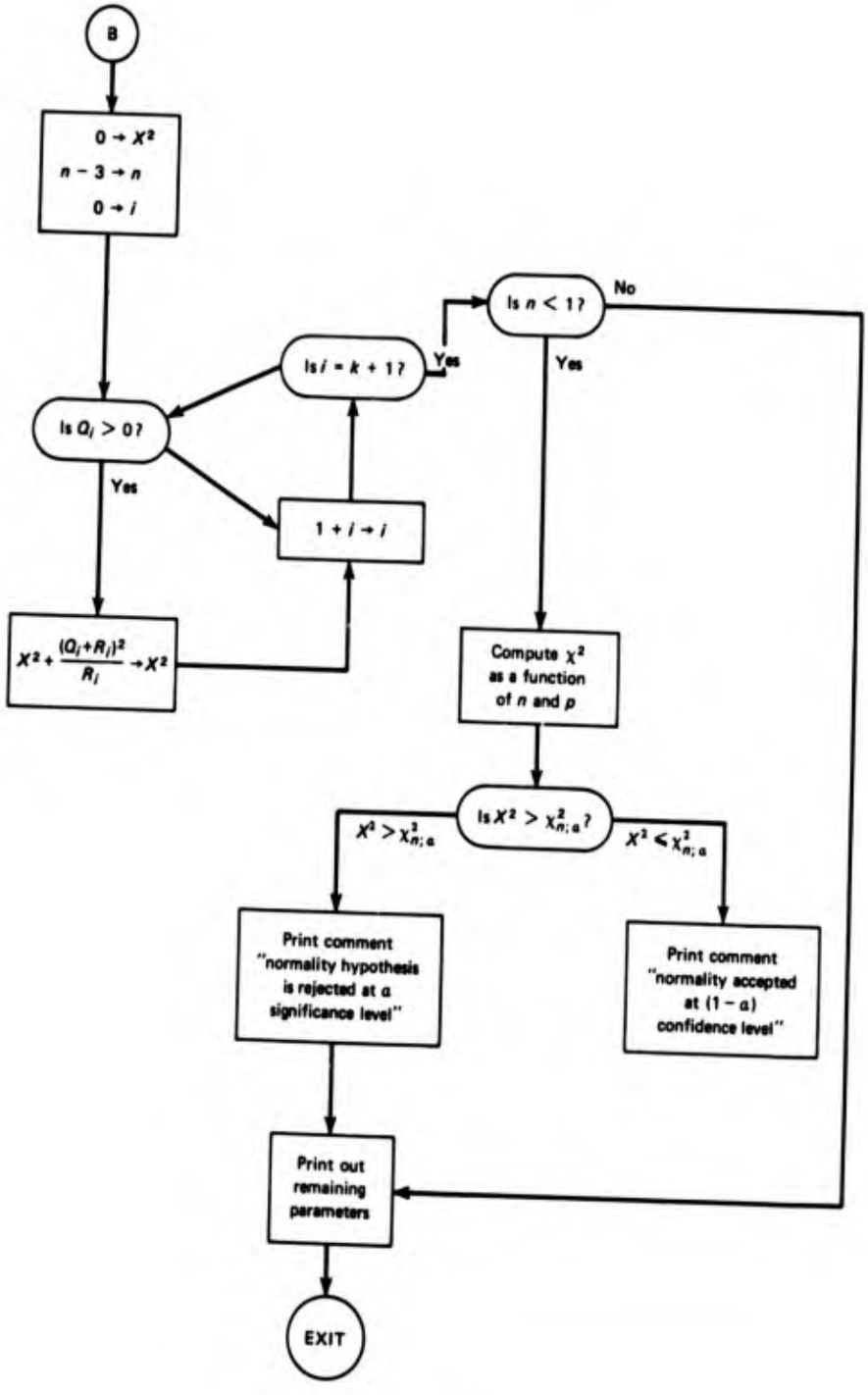


Fig. 8.2c. Processing 2.

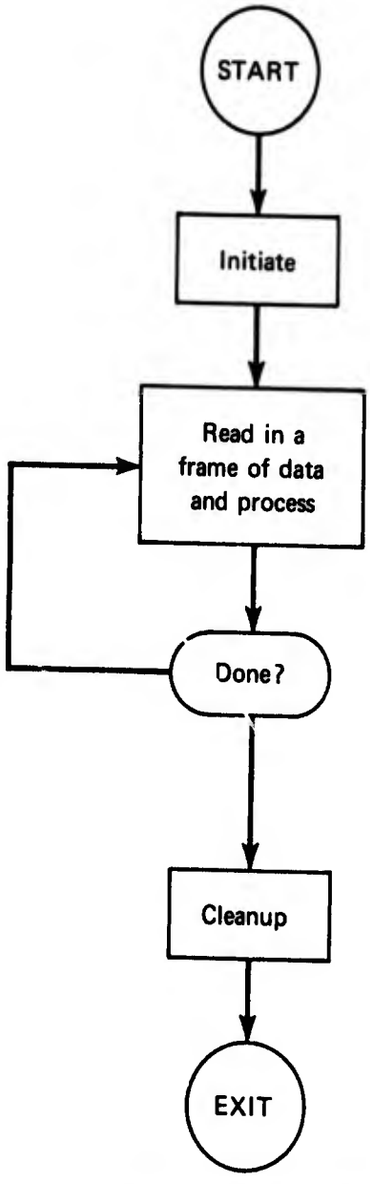


Fig. 8.3. Overall flow chart.

The extension of the procedure to multifunction parallel processing is fairly obvious, so it is omitted from the charts to clarify them. The overall flow of data is shown in Fig. 8.3. The form of program arrangement shown in this figure allows the user to recover some of the time frequently lost because of the slowness of input devices. Double buffering routines can be employed, with the processing operating independently of the routines performing the input operations, thus allowing the input routine coding to be a sort of universal building

block, usable with other applications or even separate parts of the same basic data processing program.

As a final comment on the flow charts, let us note that the boxes with printout or data input functions will probably require the largest part of the programming effort, as these items can mushroom into large tasks if speed of processing and clarity of output are requirements.

8.2 Peak Probability Density Functions

There is a serious nomenclature problem regarding the definition of peaks, maxima, and minima. There are at least three distinct problems that are of interest when discussing the occurrence of extreme values in a given record. The three are

1. Distribution of the largest (or smallest) value in a record of length T,
2. Distribution of the largest value occurring between two zero crossings,
3. Distribution of the peak values.

As an example of the first type, suppose that N values of the function $\{x_i\}$ are recorded and that they are independent with density function $f(x)$ and distribution function $\mathbf{F}(x)$. Then, the density function of the largest value of x is

$$f(x_{\max}, N) = Nf(x)\ [F(x)]^{N-1}. \tag{8.33}$$

The distribution function of $x_{\max}$ is

$$F(x_{\max}, N) = [F(x)]^N. \tag{8.34}$$

The expected value of $x_{\max}$, $\mathbf{E}[x_{\max}]$, is

$$\mathbf{E}[x_{\max}] = \int_{-\infty}^{\infty} xNf(x)\, F^{N-1}(x)\, dx. \tag{8.35}$$

Example 8.2 Suppose x_i consists of N independent, uniformly distributed random variables with the range of x being $[-1/2, 1/2]$. Then

$$f(x_i) = \begin{cases} 1 & -1/2 \leqslant x_i \leqslant 1/2 \\ 0 & \text{elsewhere} \end{cases}$$

$$F(x_i) = \begin{cases} 0 & x < -1/2 \\ x + 1/2 & -1/2 \leqslant x \leqslant 1/2 \\ 1 & x > 1/2 \end{cases}$$

$$f(x_{\max},N)=\begin{cases}N(x+1/2)^{N-1} & -1/2 \leq x_{\max} \leq 1/2\\ 0 & \text{elsewhere}\end{cases}$$

$$F(x_{\max},N)=\begin{cases}0 & x_{\max} < -1/2\\ (x+1/2)^N & -1/2 \leq x_{\max} \leq 1/2\\ 1 & x_{\max} > 1/2\end{cases}$$

$$\mathrm{E}[x_{\max}] = \int_{-1/2}^{1/2} xN(x+1/2)^{N-1}\,dx$$

$$= \frac{N-1}{2(N+1)}. \tag{8.36}$$

For $N = 1$, $\mathrm{E}[x_{\max}] = 0$, as would be expected. For large N, $\mathrm{E}[x_{\max}] \to 0.5$. If $N = 100$, then $\mathrm{E}[x_{\max}] = 0.490099^{+}$.

While $\mathrm{E}[x_{\max}]$ as given in Eq. (8.35) may be difficult to evaluate in theory, the finding of $x_{\max}$ in a sample set of data is quite easy. Indeed, even if there is no specific interest in $x_{\max}$ or $x_{\min}$, it is a good idea to have them found by the computer program anyway, as they frequently turn out to be useful when checking a set of data which is suspect.

The second type of peak analysis encountered is the largest value between two zero crossings. These are simple to find. The following algorithm is typical of one that might be employed. As usual, it is assumed that $\{x_i\}$ has N points:

1. Compute a table of $\{I_k\}, k = 1, \ldots, K$, where I_k is in the table if

$$x_{I_k} \leq 0 \quad \text{and} \quad 0 < x_{(I_k+1)}$$

or

$$0 < x_{I_k} \quad \text{and} \quad x_{(I_k+1)} \leq 0.$$

2. If $0 < x_{I_k}$, find the minima of $x_i, i = (I_k + 1), \ldots, I_{(k+1)}$; otherwise, find the maxima value.

3. Continuing, if $x_{I_k} < 0$, find the maxima of $x_i, i = (I_k + 1), \ldots, I_{(k+1)}$. Otherwise, find the minima of this same set.

With the minima and maxima thus found, two things could be done. First, use techniques from the preceding section and find the separate sample PDF's for both the set of minima and maxima. Second, set the minima positive and combine them with the maxima. Statistical properties of zero crossings are well known for many types of data, so that the maximum (or minimum) between zero crossings is relatively amenable to analysis. A somewhat more difficult

problem is presented in the third case being considered in this section, that of peak values.

In this sense, a peak value is the largest value between any two relative minima. For example, as shown in Fig. 8.4, there is only one Type 2 peak, but

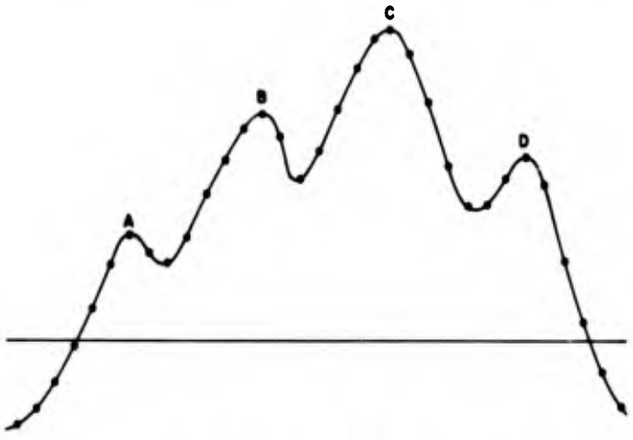


Fig. 8.4. Various types of peak values.

there are four Type 3 peaks. The procedure for locating these values is as follows:

1. For the maxima, find all x_p such that

$$x_p - x_{p-1} > 0$$

$$x_p - x_{p+1} > 0.$$

2. Similarly, for the minima, find those values x_q for which

$$x_q - x_{q-1} < 0$$

$$x_q - x_{q+1} < 0.$$

As before, these sets of maxima and minima would be processed using the type of techniques outlined in the preceding sections.

8.3 Multidimensional Density Functions

If $\{x_i\}$ and $\{y_i\}$ are any two data functions, it is possible to compute their joint sample PDF or joint histogram. This is accomplished by dividing the range of each into k_1 and k_2 intervals, as shown in Fig. 8.5. In general, more computer

storage will be required for the two-dimensional than for the one-dimensional case. For example, $(k_1 + 2)$ cells would be required for $\{x_i\}$ by itself and $(k_2 + 2)$ cells for $\{y_i\}$. Thus, $(k_1 + 2)(k_2 + 2)$ cells might be a reasonable number for their joint distribution. This might very well be excessive, especially if there are more than two functions to be processed. If there are r separate functions, then there are $(r)(r - 1)/2$ ways of comparing them. Suppose that $k_1 = k_2 = \ldots = k_r = k$, then a total of

$$\frac{(r)(r-1)}{2}\,(k+2)^2$$

storage cells would be necessary. If $r = 50$ and $k = 100$, this is 12,744,900, whereas the one-dimensional sample PDF's with the same parameter requirements would need only 5,100 cells of computer memory. (See Fig. 8.5.)

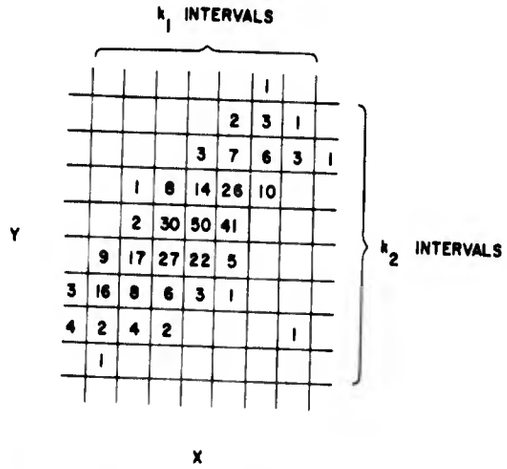


Fig. 8.5. Sample joint probability density–histogram.

A meaningful display of the results may be difficult to achieve. Computer listings of the sample joint PDF's tend to be difficult to read because of the overwhelming number of digits presented to the eye. Graphic representation is also difficult. Contour plots have been used with partial success, but their generation is not trivial. New developments in three-dimensional display devices should increase the usefulness of joint PDF's. Presently, however, the lack of completely satisfactory methods to display the results severely complicates the generation of sample PDF's and impedes their interpretation.

CHAPTER 9
NONSTATIONARY PROCESSES

9.1 Introduction

Before discussing nonstationarity, it is necessary to review the concept of stationarity itself. There is a hierarchy of definitions which refer to that term, as shown in Fig. 9.1.

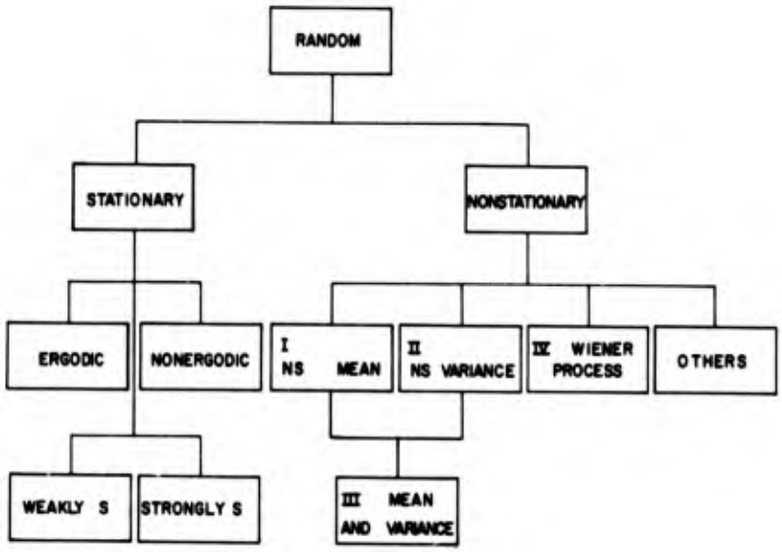


Fig. 9.1. Classification of various types of random processes.

Up to this point, it has been assumed that the data samples being examined were stationary and ergodic (no modifiers), implying that the function has expectations (mean, variance, correlation, etc.), which are invariant with translations in time. Ergodicity permits time averages to be used in place of ensemble averages. When only one or a few time histories are available, it is difficult to talk about stationarity in general, which is an ensemble property. Hence definitions in terms of time averages on a single record must be interpreted with some care. If all higher order moments and joint moments within a record are invariant with translation, the record is strongly stationary. This is a difficult proposition to prove statistically from samples of data. On the other hand, weak stationarity requires that

$$m_x(t_0) = \frac{1}{T}\int_{t_0}^{t_0+T} x(t)\,dt$$

and

$$R_x(t_0, t_0+\tau) = \frac{1}{T}\int_{t_0}^{t_0+T} x(t)\,x(t+\tau)\,dt \tag{9.1}$$

be invariant with translations of t_0. Clearly, T must be sufficiently large so that there are no problems with having too little data for the sample mean and correlation to be meaningful. Weak stationarity is usually assumed. Tests of the hypothesis that the record is weakly stationary are discussed in the next section.

What does it mean when the record is nonstationary? There are a number of conditions for stationarity that could be violated, and a variety of causes for the violation. Of the many possibilities, three have been given attention in the literature because of their frequent occurrence in actual situations.

The first such nonstationary process was analyzed extensively by Norbert Wiener, and generally goes by his name. The Wiener process can be visualized as the result of putting an ordinary stationary process into an integrator which is turned on at time zero. If x is the input and y the output, then

$$y(t) = \int_0^t x(\tau)\,d\tau. \tag{9.2}$$

If x has a mean μ_x, then the mean value of y varies with time:

$$\mu_y(t) = \begin{cases} t\mu_x & t \geqslant 0 \\ 0 & t < 0. \end{cases} \tag{9.3}$$

Similarly, it can be shown that if $R_x(\tau)$ is the correlation of x, then the variance of y is

$$\sigma_y^2(t) = \begin{cases} \displaystyle\int_0^t \int_0^t R_x(t_1 - t_2)\,dt_1\,dt_2 & t \geqslant 0 \\ 0 & t < 0. \end{cases} \tag{9.4}$$

For example, if

$$R_x(\tau) = \sigma_x^2 \delta(\tau) \tag{9.5}$$

then

$$\sigma_y^2(t) = t\sigma_x^2. \tag{9.6}$$

This is only one form of the Wiener process, which more usually is thought of as involving Brownian motion. It is referred to here because such processes do occur in some data gathering circumstances. One instance is the integration of accelerometer data to obtain velocity. If the acceleration term has been contaminated by noise, the integration noise will give rise to terms which take the form of rambling trends.

The second nonstationary process is one in which it is assumed that $R_x(\tau)$ is stationary but $\mu_x(t)$ is not. This is most easily illustrated by again supposing that an accelerometer is used as a transducer and that its output is integrated to obtain velocity. Suppose further that the integration is digital. If the conversion is off by a constant amount, then when the constant is integrated, the trend becomes linear.

The third nonstationary process to be considered will be represented in the form

$$x(t) = a(t)\, z(t). \tag{9.7}$$

By assumption, x is a zero mean process. The z function is assumed to be stationary and such that

$$\mathrm{E}[z(t)] = 0$$

$$\mathrm{E}[z^2(t)] = 1$$

$$\mathrm{E}[z(t)\, z(t+\tau)] = R_z(\tau). \tag{9.8}$$

The $a(t)$ function, however, also varies with time. The only assumptions made about it are that it changes fairly slowly and that

$$a(t) \geqslant 0, \qquad \text{for all } t. \tag{9.9}$$

This definition leads to a separable correlation function of the locally stationary form, see Ref. 42. This type of process has been discussed in Ref. 5 and others. It turns out to be a good model for several important physical systems. The one chiefly of interest is vibration in a launch vehicle. Experience has shown that

$\sigma(t)$ will reach peaks at times corresponding to lift-off, max-Q flight, and transonic flight, while at other times the level is considerably lower.*

9.2 Nonstationarity Trend Test

A useful test for trends in either mean or variance can be adapted from Ref. 41. First, one obtains a sequence of roughly uncorrelated mean values or mean square values from a time series. This can be done by subdividing the entire record of data into N time slices. From each time slice, a short-time-averaged mean value and variance can be obtained. One simple method of doing this is with the RC filtering procedure described in Section 9.4. Instead of utilizing every output point of the filter, use only those points separated sufficiently in time so as to be considered roughly uncorrelated.

The statistic to be considered is termed a *reverse arrangement*. Suppose the short-time-averaged mean or mean square values are denoted by

$$x_1, x_2, x_3, \ldots, x_N.$$

Define a reverse arrangement as occurring every time

$$x_j > x_i, j > i, i = 1, 2, \ldots, N-1. \tag{9.10}$$

For a given value of the index i, denote the number of reverse arrangements by A_i. Then consider the total number of reverse arrangements

$$A = \sum_{i=1}^{N-1} A_i. \tag{9.11}$$

It can be shown that the average value of A in a random sequence of integers is

$$\mathrm{E}[A] = \frac{N(N-1)}{4}. \tag{9.12}$$

This is easily seen by considering the fact that x_1 is equally likely to be larger or less than the succeeding $(N-1)$ members of a random sequence. Therefore, the average number of reverse arrangements, when considering x_1 against the remaining members, is

$$\mathrm{E}[A_1] = (N-1)/2. \tag{9.13}$$

*It should be noted at this point that there is also a tendency for the peaks in the PSD of this type of data to move higher in frequency as time increases, mainly because of the depletion of fuel.

Now x_2 is compared against the $(N-2)$ remaining members of the sequence, hence

$$\mathbf{E}[A_2] = (N-2)/2. \tag{9.14}$$

Continuing the reasoning leads to

$$\mathbf{E}[A_{N-2}] = 2/2$$

and

$$\mathbf{E}[A_{N-1}] = 1/2. \tag{9.15}$$

The average value of A is obtained by adding up all these individual averages:

$$\mathbf{E}[A] = \sum_{i=1}^{N-1} \mathbf{E}(A_i) = \frac{1}{2} \sum_{i=1}^{N-1} i = \frac{N(N-1)}{4}, \tag{9.16}$$

which is one-half the sum of the first $(N-1)$ integers.

Also, the variance of A is found to be

$$\text{Var}[A] = \frac{2N^3 + 3N^2 - 5N}{72}. \tag{9.17}$$

Fortunately, for $N \geq 10$ the distribution of A may be closely approximated by the normal distribution

$$\mathbf{P}[c] = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{c} e^{-x^2/2}\, dx, \tag{9.18}$$

where

$$c = \frac{\left[A + \frac{1}{2} - \frac{N(N-1)}{4}\right]}{\sqrt{\frac{2N^3 + 3N^2 - 5N}{72}}}. \tag{9.19}$$

Note that c in Eq. (9.19) above is of the form $(x-\mu)/\sigma$, that is, a standardized observation where $x = A + 1/2$, $\mu = N(N-1)/4$, and $\sigma = \sqrt{\text{Var}[A]}$. The additive factor of 1/2 which appears in the numerator accounts for the fact that A is an integer and that, in a continuous approximation, A is considered to be in the interval $[A - 1/2, A + 1/2]$. An alternative method of defining A is as the sum

$$A = \sum_{i<j} \sum a_{ij},$$

where

$$a_{ij} = \begin{cases} 1 & \text{if } x_i > x_j \\ 0 & \text{otherwise.} \end{cases} \tag{9.20}$$

A test of a hypothesis of the existence of a trend would proceed as follows: Note that if, for example, the mean square value of a process was rising linearly as indicated in Fig. 9.2, then one would expect A to be very large. That is, one would expect to find x_1 less than all subsequent observations except for a few cases caused by statistical sampling variations. On the other hand, an extremely small value for A would tend to indicate a decreasing trend in the mean square value.

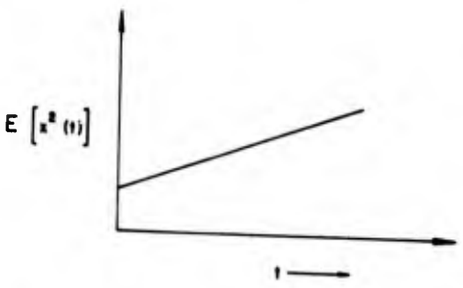


Fig. 9.2. Process with linearly increasing variance.

The test is performed at a specified level of significance by use of Eq. (9.18) if $A \geq 10$ and by Table 9.1 for $A < 10$ to obtain the necessary probabilities of A being a given size. For example, suppose $N = 8$ and $A = 14$. According to Table 9.1, the probability of A being less than or equal to 13 is 0.452. The right half of the probabilities for $N = 8, 9, 10$ for Table 9.1 are not given explicitly but are obtained directly since the distribution is symmetric. That is, the entry in the table for $N = 8$ and $A = 14$ is $1 - P(13) = 0.548$. Similarly, the entry for $A = 15$ is $1 - P(12) = 0.640$, etc.

For the stated example, a hypothesis of no trend would be accepted at, say, the $\alpha = 5$-percent level of significance. A two-tailed test would be used, which means that either too large or too small a value for A would tend to indicate a trend and require rejecting the hypothesis of no trend. For this level of significance, if the probability of obtaining the experimentally determined A is either less than 0.025 = 2.5 percent or larger than 0.975 = 97.5 percent, then the hypothesis of no trend would be rejected.

For values of $N \geq 10$, c is computed as given by Eq. (9.6), and a value for $P[c]$ is found either by inspecting a table of the normal distribution or (in a computer) by employing the approximation in Appendix B. For the given example,

Table 9.1. Percentage Points of Reserve Arrangement Distribution

Values of $A_{N;a}$ such that $[A_N > A_{N;a}] = a$.
where N = total number of measurements

N	a					
	0.99	0.975	0.95	0.05	0.025	0.01
10	9	11	13	31	33	35
12	16	18	21	44	47	49
14	24	27	30	60	63	66
16	34	38	41	78	81	85
18	45	50	54	98	102	107
20	59	64	69	120	125	130
30	152	162	171	263	272	28[illegible]
40	290	305	319	460	474	489
50	473	495	514	710	729	751
60	702	731	756	1013	1038	1067
70	977	1014	1045	1369	1400	1437
80	1299	1344	1382	1777	1815	1860
90	1668	1721	1766	2238	2283	2336
100	2083	2145	2198	2751	2804	2866

$$c = -\frac{13.5 - 8(7)/4}{\sqrt{\frac{2(8)^3 + 3(8)^2 - 5(8)}{72}}} = \frac{13.5 - 14}{4.8} = -0.10. \tag{9.21}$$

Inspection of the normal distribution table yields $P[c] = 0.460$. This value is in error by a little more than 2 percent from the correct value 0.452 obtained from Table 9.1, which indicates that even for N as small as 8, the Gaussian approximation is useful for some purposes.

The Type 2 error has not been established for various specific trends, although a rough comparison has been performed against the data given in Ref. 43, Section 16.4.3. The comparison indicates that this nonstationarity trend test is approximately as powerful as stationarity test B for the type of nonstationarity employed in those experiments. This nonstationarity trend test will be effective against monotonic trends, but one can intuitively see that the test will have many limitations as far as certain types of nonstationary data are concerned. For example, the experiments performed in Ref. 43 used nonstationary data with a mean square value as indicated in Fig. 9.3a. However, if the mean square value jumped as indicated above, but then returned to its original level as shown in Fig. 9.3b, and if the test were being applied over this entire set of data, the test would

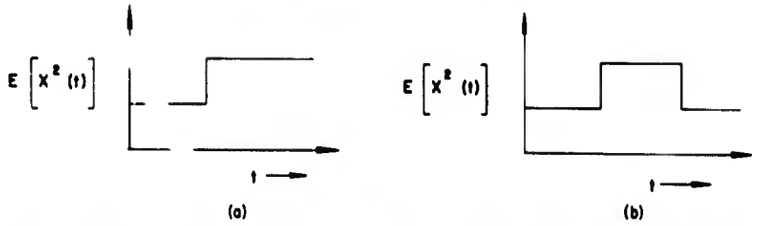


Fig. 9.3. (a) Another nonstationary variance, (b) An example the test did not detect.

probably fail. This is because, although probably too many reverse arrangements would occur over the first portion of the data, too few would most likely occur in the latter parts to obtain an overall value which was not unusual. Similarly, the test is probably effective in detecting the nonstationarity of the mean square value of the type indicated in the top part of the Fig. 9.4 and ineffective against the type of nonstationarities indicated in the lower part of the figure.

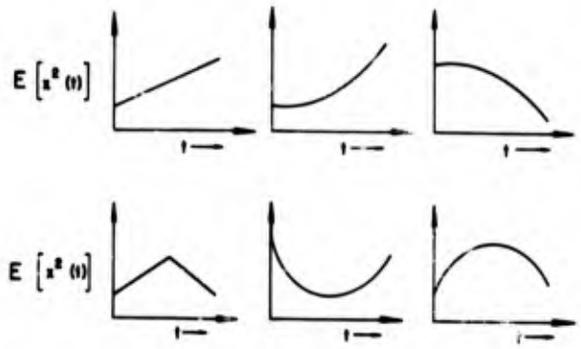


Fig. 9.4. Varieties of nonstationary variances.

9.3 Nonstationary Frequency Response Test

An approach to determining stationarity of estimated filter (frequency response) relations has been applied to seismic array noise (Refs. 44 through 47). The idea here is as follows: Suppose an input-output relation $\hat{H}_1(f)$ is determined from measured time histories over time interval T_1. The question now arises, is this input-output relation the same now as $\hat{H}_2(f)$ measured later over time interval T_2? The first tendency is to perform a statistical test by direct comparison of $\hat{H}_1(f)$ and $\hat{H}_2(f)$ which is possible by the confidence interval results of Chapter 7. This approach is unreasonable, as can be seen by considering a case of three highly coherent variables. Suppose

$$Y(f) = H_1(f)X_1(f) + H_2(f)X_2(f). \tag{9.22}$$

If $X_1(f)$ and $X_2(f)$ are perfectly coherent, then $X_1(f) = X_2(f)$ and

$$Y(f) = [H_1(f) + H_2(f)]X_1(f). \tag{9.23}$$

Then, individual estimates of $H_1(f)$ and $H_2(f)$ could be highly variable even though $[H_1(f) + H_2(f)]$ is stable.

A better question to ask is, How well do the filter relations $\hat{H}_1(f)$ and $\hat{H}_2(f)$ determined at time T_1 allow $X_1(f)$ and $X_2(f)$ to account for $Y(f)$ at a later time T_2? The multiple coherence function can be applied to this question. It will not depend on the individual estimates $\hat{H}_1(f)$ and $\hat{H}_2(f)$, but rather on how well the combination of the two works to predict $X(f)$. The quantity employed to make a plausible test of this type of stationarity is a pseudo multiple coherence function defined as follows: Let $\hat{\gamma}^2_{ij}(f)$ denote the multiple coherence function using the frequency response estimates from the ith time slice applied to the jth time slice. Let $\hat{\mathbf{H}}_{ii}(f)$ be the frequency response function vector of p components estimated from the ith time slice. Then $\hat{\gamma}^2_{ij}$ (dropping the frequency argument for convenience) is defined by

$$\hat{\gamma}^2_{ij} = \hat{\gamma}^2_{jj} - \frac{(\hat{\mathbf{H}}_{ii} - \hat{\mathbf{H}}_{jj})^* \hat{\mathbf{G}}_{xxj} (\hat{\mathbf{H}}_{ii} - \hat{\mathbf{H}}_{jj})}{\hat{G}_{yyj}}. \tag{9.24}$$

Thus the multiple coherence $\hat{\gamma}^2_{ij}$ is the multiple coherence $\hat{\gamma}^2_{jj}$ at the jth time slice degraded by the amount that comes from using $\hat{\mathbf{H}}_{ii}$ instead of $\hat{\mathbf{H}}_{jj}$. Note that if $\hat{\mathbf{H}}_{j}$ is substituted for $\hat{\mathbf{H}}_{ii}$ in Eq. (9.24), then $\hat{\gamma}^2_{ij}$ becomes equal to $\hat{\gamma}^2_{jj}$ as it should since the amount of degradation goes to zero.

The statistical aspects of comparing the multiple coherence are not worked out unless time slices T_i and T_j are separated sufficiently to make the multiple coherences independent. Then the confidence limit formula of Chapter 7 is employed. Otherwise the values of the various γ^2_{ij} can only be compared in an approximate manner. However, it does provide a reasonable approach to evaluating stationarity of a multichannel process.

The computational sequence is

1. Perform all spectral matrix, frequency response, and coherence function computations for a time slice T_i,
2. The parameter estimates $\hat{\gamma}^2_{ii}$, $\hat{\mathbf{H}}_{ii}$, $\hat{\mathbf{S}}_{xx_i}$, and $\hat{\mathbf{S}}_{xy_i}$ all must be saved (on magnetic tape for example),
3. Steps 1 and 2 are repeated for all desired time slices,
4. The pseudo coherence $\hat{\gamma}^2_{ij}$ is computed for all values of i and j. This produces a $q \times q$ array, if there are q time slices, for each frequency point f.

9.4 Time-Varying Mean and Variance

By far the most common test for stationarity is that of simple visual observation, usually of the smoothed data, or the input of an rms meter, using some

form of analog hardware. For the mean value, the analog device may be thought of as an RC filter such as that described in Section 3.1. The time history is the input to it, and the output is plotted on an oscillograph. The equivalent sort of operation may be performed using digital filters. As will be recalled from Section 3.3, the digital RC filter is

$$y_i = ay_{i-1} + (1-a)x_i,$$

$$a = e^{-(\Delta t/RC)}, \tag{9.25}$$

where x is the input to and y is the output from the filter. The parameter RC (and hence a) can only be chosen from previous knowledge of the data or as the result of reruns on the same data. Experience in performing the operation will help the user in subsequent tests.

The rms procedure is somewhat different when using a computer from that followed in using analog devices, where rectification is frequently performed in place of squaring. The digital procedure is shown in Fig. 9.5. Squaring is easily

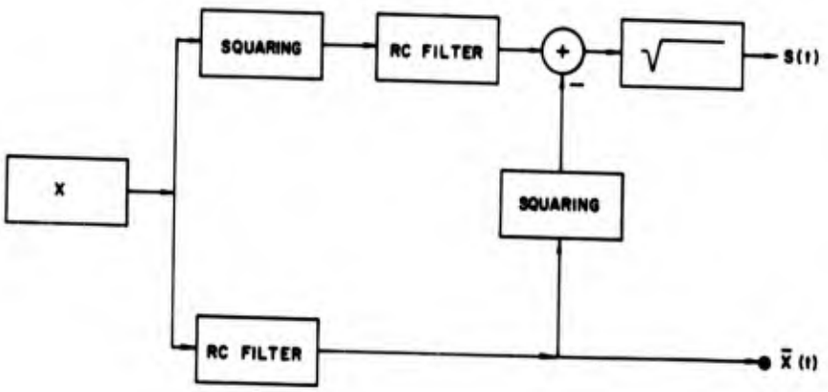


Fig. 9.5. Digital rms procedure.

done on a computer so that this type of operation is relatively simple. The equations for $\bar{x}_i$ and s_i, the running mean and standard deviation, are

$$\bar{x}_i = ax_i + (1-a)\bar{x}_{i-1}$$

and

$$s_i = \sqrt{z_i - \bar{x}_i^2},$$

where

$$z_i = ax_i^2 + (1-a)\,z_i. \tag{9.26}$$

Alternatively, low-pass filters of the type described in Section 3.6 may be used in place of the RC filter employed above. There is no obvious gain from so doing, other than the difference in frequency response functions.

If much of this type of processing has to be done, it may be possible to save large amounts of computer time by decimating the data, as described in Section 3. Decimation in this case would be performed before the data are put through the rms procedure. There is a problem in interpreting the information obtained from this process. If both $s(t)$ and $x(t)$ are relatively constant, the hypothesis of stationarity has some plausibility. On the other hand, if either of them is varying noticeably, it would seem reasonable to assume that x is nonstationary.

The faults of the above are fairly obvious:

1. There are no quantitative parameters in the test on which a decision is based. Acceptance or rejection is purely qualitative.

2. The procedure, if it tests any hypothesis at all, tests for homogeneity of mean and variance rather than stationarity. For example, suppose the first half of the record consists of a sinusoid of frequency f_0 and the second half consists of a sinusoid of frequency f_1, where $f_1 \neq f_0$, but their amplitudes are both A units. $\bar{x}(t)$ will be zero and $s(t)$ will be $\sqrt{2}/2A$. On the basis of these observations, the observer would accept the data as being stationary when indeed it is not.

3. If too little smoothing is done, $\bar{x}_i$ and s_i may fluctuate considerably, perhaps leading the observer into believing that the data are nonstationary when in fact they are stationary.

Nevertheless, this procedure is probably one of the most effective in performing a preliminary analysis of a record that possibly is nonstationary.

9.5 Time-Varying Power Spectra

The concept of a time-varying PSD is a generalization of the time-varying variance. As with the standard PSD, there are three ways of performing the calculations. These are

1. Computing the correlation function and the Fourier transform,
2. Computing the Fourier transform and then taking its modulus squared,
3. Using numerical filters.

The last two appear to be the most promising, so they will be discussed in detail.

When the Fourier transform procedure is chosen, the FFT should be employed. Some power of two, say 512, is selected. The data record is broken into segments, each containing 256 data points. The PSD is computed by taking the FFT of each pair of adjoining segments. Label each segment as S_ℓ. Then

$$\begin{aligned} S_\ell &= \{x_{\ell i}\} \\ &= (x_{\ell 1}, x_{\ell 2}, \ldots, x_{\ell 256}) \\ x_{\ell i} &= x_{256\ell + i}. \end{aligned} \tag{9.27}$$

At time T_ℓ, where

$$T_\ell \equiv [512\ell - 256]\ \Delta t, \qquad \ell = 1, \ldots, \tag{9.28}$$

the Fourier transform of the two segments is computed:

$$F_{\ell k} = \Delta t \left\{ \sum_{i=0}^{255} e^{-\frac{j\pi ik}{512}} x_{\ell i} + \sum_{i=0}^{255} e^{\frac{-j\pi k(i+256)}{512}} x_{(\ell+1)i} \right\},$$

$$k = 0, 1, \ldots, 511. \tag{9.29}$$

The PSD $\{G_{\ell k}\}$ is therefore

$$G_{\ell k} = \frac{1}{256\Delta t} |F_{\ell k}|^2, \qquad k = 0, 1, \ldots, 255. \tag{9.30}$$

Equation (9.29) looks formidable. In actual practice, the process is less difficult than it would appear. The computer program would take the following steps:

1. Initiate, including reading segment S_1 into a buffer area.
2. Transfer contents of the buffer into the upper half of the transform area. Read the next segment into the buffer area and then transfer it into the lower half of the transform area.
3. Compute the FFT of the contents of the transform area.
4. Compute the PSD from the Fourier transform and output the results.
5. Return to Step 2 until all of the data have been processed.

The reason for the slight complication in Step 2 is simple. Many of the computer subroutines to compute the FFT are destructive, i.e., they put the computed results back on top of the original data. This requires that the next segment be saved elsewhere in the computer. Also, the data probably will have to be moved in any event. While it is possible to compute the transform with the data scattered in various locations within the computer memory, this adds another level of complexity to the routines used to compute the FFT. On the other hand, with a fixed size of 512 points at a time, there are a number of economies which can be made in the computation with little effort. A little reflection will show that only 128 numbers are required to express all of the sines and cosines required. Thus, these can be precomputed and a short routine written to deliver the appropriate values. Secondly, the 512 reverse addresses required by the FFT may also be precomputed.

Another peculiarity of the FFT procedure may be employed. As discussed in Chapter 4, the FFT is obtained from one complex or two real sequences at the same time. Therefore, as the data processes under consideration are assumed to be real, either two processes may be transformed at one time or a single one may be operated upon at double speed. The latter choice has the following steps:

1. Initiate. This includes reading segment S_1 into the buffer.

2. The contents of the buffer are moved to the upper half of the real area. The next two segments are read in. The segments are stored as follows:

$$\text{Buffer} \rightarrow \text{upper half of real area}$$

$$s_i \rightarrow \begin{cases} \text{lower half of the real area} \\ \text{upper half of the complex area} \end{cases}$$

$$s_{i+1} \rightarrow \begin{cases} \text{lower half of the complex area} \\ \text{buffer} \end{cases}$$

3. Compute the FFT and sort out the results corresponding to the real and imaginary parts.

4. Compute the two PSD's and output the results.

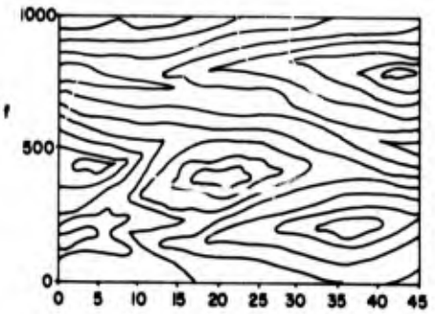
5. Return to Step 2 until all of the data have been processed.

One point glossed over in the above description of the procedure is the manner and form of the output. There are a number of choices. Among the most prominent of these are plots of the following types:

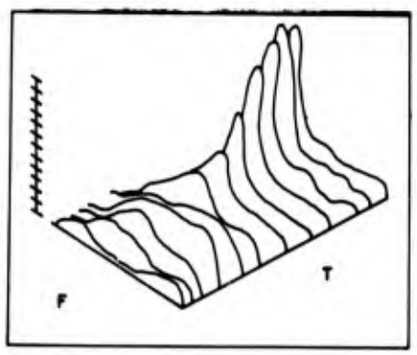
1. One plot for each time at which the computations were made

2. One plot for each frequency; each such plot would be a function of time versus power.

3. A contour plot; this would read like a topological map. One edge of the plot (usually the bottom) would be time. The side axis would be frequency. The power would be represented by a series of contours; along each contour the power would be constant. Five to ten levels would be selected, and the resulting curves would tend to be concentric and closed. This is shown in Fig. 9.6a.



(a)



(b)

Fig. 9.6. (a) PSD contour plot; (b) PSD profile plot.

4. A profile plot can be made as shown in Fig. 9.6b. These are somewhat easier to visualize than contour plots, but it is more difficult to make accurate readings from them.

Contour plots are probably the most expensive to generate. All of the data (or at least significant portions of it) must be readily available in storage. There is a lot of checking and interpolation to be performed, which adds to the running time, and many people find contour plots difficult to read.

Profile plots pose a dilemma: If all the information is plotted, the graphs look confusing because of overlapping lines. On the other hand, if portions of data that fall behind more forward parts of the graph are deleted in order to enhance the plot, important results may be lost from view. Profile plots are considerably cheaper to generate, however, and may be produced without requiring amounts of core storage. A reasonable compromise might be to produce the second type of plot discussed above, i.e., one plot of power versus time for each frequency, followed by a profile plot to give an overall impression of the process.

Bandwidth is another important consideration in setting up such an analysis. It is likely that, rather than employing the 256 equally spaced frequencies, the user would prefer a broader bandwidth analysis. Power could be combined in several ways, as discussed in Chapter 6, depending upon the application. Perhaps a more natural way of computing the time-varying PSD's is through the use of filtering. A PSD produced in this manner is a simple extension of two procedures discussed earlier, namely the direct filtering method described in Section 6.2 and the time-varying variance developed in Section 9.4. Each frequency would take the form shown in Fig. 9.7. The first part of the procedure consists of a simple routine to remove the mean. The bandpass filter that follows could be of any size, but a six-pole filter is a good compromise between the sharpness of the filter and the cost of computation. The squaring and filtering that follow are exactly the same as for the variance computations described in Eq. (9.26).

The problems of display are identical to those of the FFT discussed above with one exception: The RC filter used to smooth the squared data introduces a delay, which must be accounted for when labeling the time axis. This delay is approximately

$$\text{Delay} = RC\ \ell n\ 2, \tag{9.31}$$

where RC is the filtering parameter.

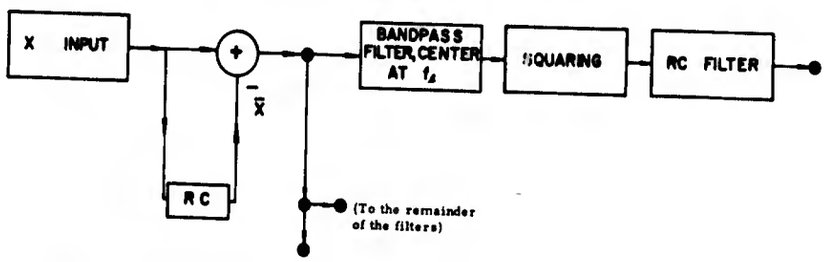


Fig. 9.7. One segment of a program designed to compute a time-varying PSD.

9.6 Clipped-Correlation Nonstationary Procedure

The relation described in Chapter 5 between the correlation function for a strongly clipped signal and the correlation function for the original signal is applied here. The relations described are not proven with any theoretical rigor. However, arguments are given for the plausibility of the methods, and they should be very useful in practice.

The special form of nonstationary processes to be considered are of the third type discussed in Section 9.1. That is, where

$$x(t) = a(t)\, z(t). \tag{9.32}$$

In Eq. (9.32), $z(t)$ is assumed to be a zero-mean, Gaussian, stationary process. The quantity $a(t)$ is again assumed to be a slowly varying modulating function acting like a time-varying scale factor. It has been empirically demonstrated in Ref. 42 that Eq. (9.32) can serve in some instances as a satisfactory model for nonstationary data. For max Q, the period of maximum aerodynamic boundary layer noise during a missile launch, the nonstationarity was shown to be satisfactorily modeled by Eq. (9.32). For the transonic portion of the flight (transition through Mach 1), the model was unsatisfactory.

Consider the clipped process

$$y(t) = \begin{cases} 1 & x(t) \geqslant 0 \\ -1 & x(t) < 0. \end{cases} \tag{9.33}$$

The correlation function of $x(t)$ relates to the correlation function of $y(t)$ in the following manner:

$$\rho_x(\tau) = \sin\left[\frac{\pi}{2} R_y(\tau)\right], \tag{9.34}$$

where $\rho(\tau)$ is the normalized correlation coefficient function. It can be very simply argued now that the autocorrelation function of $x(t)$ computed by this method is exactly the same as that of $z(t)$ computed by a clipping method. This is true because, when the process is clipped, only the zero crossing information remains. The time-varying scale factor $a(t)$ will have no effect on the zero crossings (assuming $a(t)$ is nonnegative). Hence, since the correlation function based on clipping the $x(t)$ time history is computed from the identical information as the correlation function based on the $z(t)$ time history, ρ_x computed from Eq. (9.34) can be used as an estimator of the stationary autocorrelation function ρ_z. Finally, the time-varying characteristics of ρ_x can be obtained from a separate estimate of Var $[x(t)]$. The simplest approach on a digital computer is to use the squared output of a low-pass RC recursive numerical filter as discussed in Section 9.4. Under the assumption that $a(t)$ is not varying too

rapidly (as compared to $z(t)$), then reasonably short-time-averaged estimates of Var $[x(t)]$ can be obtained. The selection of the RC time constant (or equivalently, the effective averaging span) of the filter will depend on the particular type of data.

The estimation procedure for a nonstationary spectral analysis then is

1. Perform extreme clipping of $x_i = a_i z_i$,

$$y_i = \begin{cases} 1 & x_i \geq 0 \\ -1 & x_i < 0. \end{cases} \tag{9.35}$$

2. Compute $\hat{R}_{yr}$, $r = 0, 1, \ldots, m$. Special high-speed computational procedures based on the clipped process can be employed here.

3. Perform the sin (x) bias correction to obtain $\hat{\rho}_{xr}$;

$$\hat{\rho}_{xr} = \sin\left[\frac{\pi}{2}\hat{R}_{yr}\right], \qquad r = 0, 1, \ldots, m. \tag{9.36}$$

4. Compute the Fourier transform of $\hat{\rho}_{xr}$ tapered by a lag window to normalized spectrum to obtain $\hat{g}_{xk}$;

$$\hat{g}_{xk} = \mathcal{F}[u_{mr}\hat{\rho}_{xr}], \qquad k = 0, 1, \ldots, m. \tag{9.37}$$

5. Compute estimates of Var $[x_i]$ by numerical RC filtering as illustrated in Section 9.3;

$$s_j^2 = \text{Var}\,[x_i] = \overline{x_i^2} - (\overline{x}_i)^2, \tag{9.38}$$

where x^2 and $(x_i)^2$ are the outputs from an RC numerical filter. The final plotted spectrum should be

$$\hat{G}_{xk} = s_j^2\,\hat{g}_{xk}. \tag{9.39}$$

For each value of the time index j, a plot of G_{xk}, $k = 0, 1, \ldots, m$ can be made.

Many of the statistical aspects of spectral density estimation using clipped time series are covered in Ref. 25. An example for an ideal narrow bandwidth spectrum is presented there. It is shown that the standard deviation of the spectrum estimated by the clipping procedure is $0.72\pi/2 = 1.13$ larger than with regular methods. Thus, for this case, a 13-percent increase in standard deviation is experienced. For variances, the factor is $(0.72\pi/2)^2 = 1.28$, which is a 28-percent increase. A corresponding increase in record length is required to maintain equivalent statistical accuracy.

CHAPTER 10
TEST CASE AND EXAMPLES

In this chapter, several plots obtained by using a package of computer programs, called the MAC/RAN* system, are presented. These plots illustrate typical results, which can be obtained from digital computer processing of data with programs that implement the methods described in the preceding chapters. Many plots of frequency response functions in various forms, power spectra, cross spectra, autocorrelation, crosscorrelation, and the various types of coherence functions are given as figures in the following pages.

To produce these results, several pseudo random time histories were generated by pseudo random number generators. The basic procedure followed to obtain the time histories used for the test case is as follows:

1. Use a subroutine for uniformly distributed random numbers to generate a sequence of N independent pseudo random numbers uniformly distributed on the interval $-1/2$ to $+1/2$.

2. Form the sum of each set of 12 contiguous numbers to form a new set of random numbers x_i, $i = 1, \ldots, N/12$. The original set of uniformly distributed random numbers had a mean value $\mu = 0$ and a variance $\sigma^2 = 1/12$. The variance of the sum of independent random variables is the sum of the variances and likewise for the mean values. Thus, the random variables x_i have a mean value of zero and a variance of unity. By invoking the central limit term, the variable x_i will have an approximately Gaussian probability density function and lie within the range -6σ to $+6\sigma$ and thus have a range of ± 6 standard deviations. These sequences now represent approximately Gaussian, white noise processes. Successive data values are very close to being independent, so the power spectral density function will be approximately flat and, hence, these will be white noise variables.

3. Variables generated in this manner can now be combined in various ways to construct correlated pseudo random white noise time histories. For the example at hand, three independent processes x_2, x_4, and x_{11} were combined in the following manner:

$$x_3 = x_2 + x_4$$

$$x_{12} = \frac{1}{2} x_{11} + x_4.$$

In this manner, the time histories x_2, x_4, and x_{11} become related to x_3 and x_{12} while x_2, x_4, and x_{11} remain independent of one another.

*MAC/RAN is a registered trademark of Measurement Analysis Corporations, Marina del Rey, California.

4. These time histories are operated on by numerical filters, and time histories having specified correlation function properties or spectral density properties are generated. The numerical filtering procedure amounts to taking linear combinations of contiguous sets of the original independent random variables. Hence, the output will be sequences of data points that will be correlated and thus have a correlation function different from a spike and in turn have a power spectral density function which has been given some shape. For the sample case being illustrated, new time histories were generated as follows:

$$x_5 = \text{RC filtered } x_2 \quad \text{(weights 0.8 and 0.2)}$$

$$x_6 = \text{RC filtered } x_{12} \quad \text{(weights 0.6 and 0.4)}$$

$$x_7 = \text{RC filtered } x_3 \quad \text{(weights 0.6 and 0.4)}$$

5. Finally, the sum of these independent random variables is computed, which may now be thought of as the output of a three-dimensional linear system. For the case at hand, the output variable is

$$x_8 = x_5 + x_6 + x_7 + \frac{\text{white noise}}{10}.$$

In the results which follow, the test case is the set of four time histories $x_8(t)$, $x_5(t)$, $x_6(t)$ and $x_7(t)$, some of which are correlated with one another. The three inputs were put through RC numerical filters, and the output then was the sum of these filtered time histories. In addition, one more independent time history was added to the output to represent extraneous noise introduced into the system.

In Fig. 10.1, the probability histograms of the four time histories are shown, on which are superimposed the Gaussian probability density curves, having the sample mean and variance determined from the sample time histories. As one can see from inspection, these probability density functions are very close to normal probability density functions, as the central limit term predicts. Note that the plots are labeled Channel 8, Channel 2, Channel 3, and Channel 12. These are merely labels to identify the original time histories $x_8(t)$, $x_2(t)$, $x_3(t)$, and $x_{12}(t)$. In some later plots the identifications Channel 1, Channel 2, and Channel 3 are used to designate the three transmission channels of the linear system.

Figure 10.2a is the autocorrelation function of $x_8(t)$, the output, for M = 64 lags. Figure 10.2b is the corresponding Blackman-Tukey PSD with n = 64 d.f. Figure 10.2c is the PSD of the same time history with n = 64 d.f. but computed via the FFT procedure described in Chapter 6.

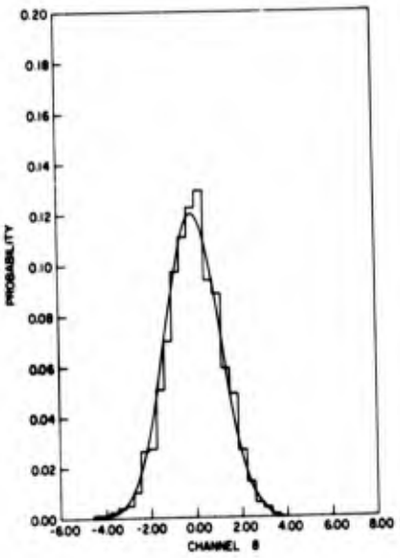


Fig. 10.1a. PDF for Channel 8.

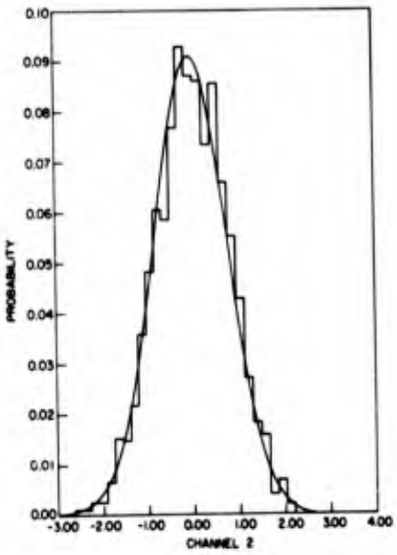


Fig. 10.1b. PDF for Channel 2.

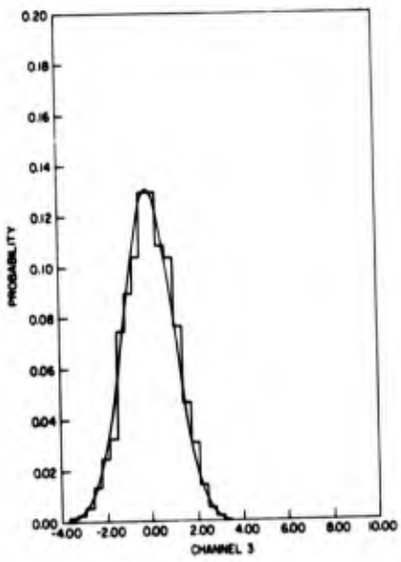


Fig. 10.1c. PDF for Channel 3.

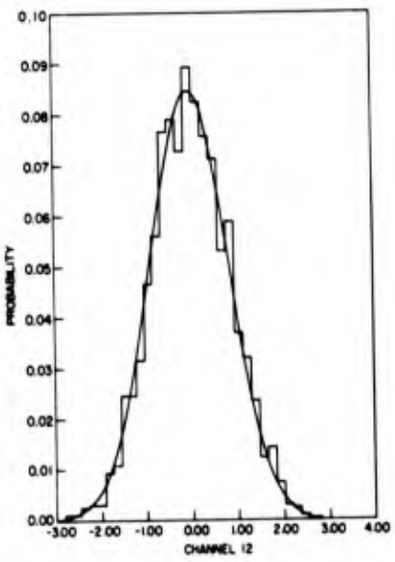


Fig. 10.1d. PDF for Channel 12.

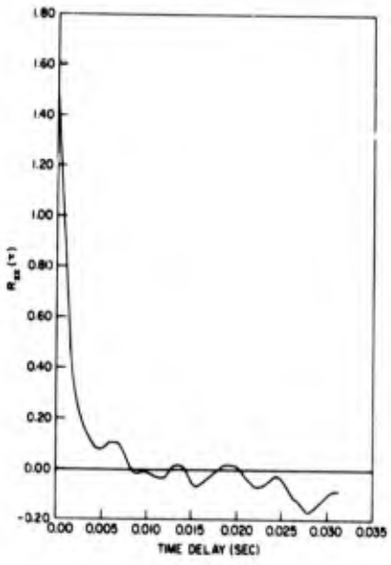


Fig. 10.2a. Autocorrelation of Channel 8 (64 lag).

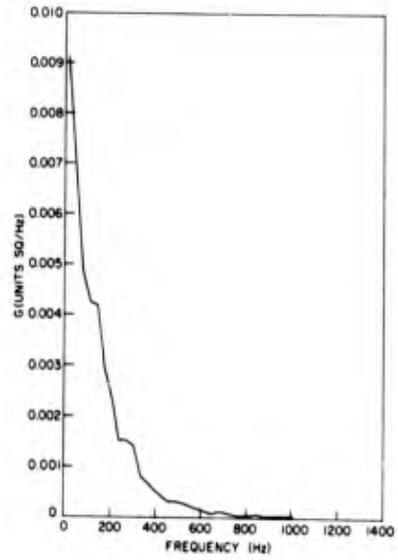


Fig. 10.2b. B-T PSD of Channel 8 (64 d.f.).

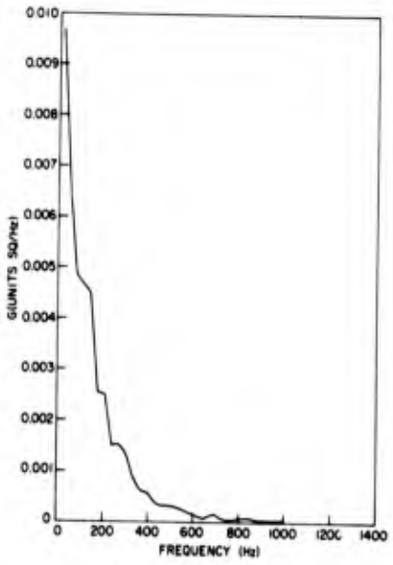


Fig. 10.2c. FFT PSD Function of Channel 8 (64 d.f.).

Figures 10.3a, b, and c are the 64-lag autocorrelation function, 64 d.f. B-T PSD, and 64 d.f. FFT PSD, respectively for $x_3(t)$. On the facing page are Figs. 10.4a, b, and c, which are the 128-lag correlation function, 32 d.f. B-T PSD, and 32 d.f. FFT PSD, respectively. Thus, the resolution bandwidth is halved in going from Fig. 10.3 to Fig. 10.4, as is the number of d.f. The FFT PSD plots illustrate the increase in statistical variability most dramatically. The automatic scaling procedure selected a scale for the B-T PSD in Fig. 10.4b, which is half that of Fig. 10.3b. Also, the number of points plotted for the second B-T PSD was not doubled as for the second FFT PSD. Hence, at first glance the B-T PSD of Fig. 10.4b does not appear to have experienced the increase in variability that the FFT PSD of Fig. 10.4c has. On more careful consideration, however, one can see that the increase is comparable.

Figure 10.5a presents the crosscorrelation function of the output variable $x_8(t)$ with one of the input channels, $x_3(t)$. As can be seen by the cross-correlation function, a small time delay has been generated in the numerical filtering of the data since the maximum point in the corsscorrelation function does not occur at $\tau = 0$. The corresponding plots of the B-T cospectrum and quadspectrum appear in Fig. 10.5b and c.

The modulus and phase of the cross spectrum are presented in Fig. 10.6. The B-T version is in Fig. 10.6a and b, and the FFT version is in Fig. 10.6c and d.

The corresponding frequency response function for the same input-output pair computed three different ways is given in Figs. 10.7 and 10.8. The gains are in Fig. 10.7, which are the B-T version, the FFT version, and FFT version computed directly from ratios of smoothed transforms. Parts a and b are dB plots, and c is in linear units. The corresponding phase angles are in Fig. 10.8 on the facing page. The title, "Channel 2," refers to the second transmission channel, which has $x_3(t)$ as the input.

The various coherence functions generated via the two different methods make up Figs. 10.9 and 10.10. Parts a, b, and c of Fig. 10.9 are the B-T version ordinary, multiple, and partial coherence functions for the Channel 2 $[x_3(t)]$ input. The corresponding parts of Fig. 10.10 are the FFT counterparts.

The remaining plots are presented to illustrate the improved statistical reliability obtained in frequency response estimates when the coherence is essentially unity. Figures 10.11a, b, and c and 10.12a, b, and c are the counterparts of the frequency response functions plotted in Figs. 10.7 and 10.8. The d.f. of the estimates are the same, but a single-input, single-output system is simulated with no contaminating extraneous noise. That is, $x_3(t)$ is still the input time history, but $x_7(t)$ rather than $x_8(t)$ is now the output. As can be seen, the plots are now nearly ideal, as would be expected from a perfect, discrete, linear system. Figures 10.11a, b, and c are the gains via the B-T, FFT, and FFT transform ratio methods respectively. Figures 10.12a, b, and c are the corresponding phase angles.

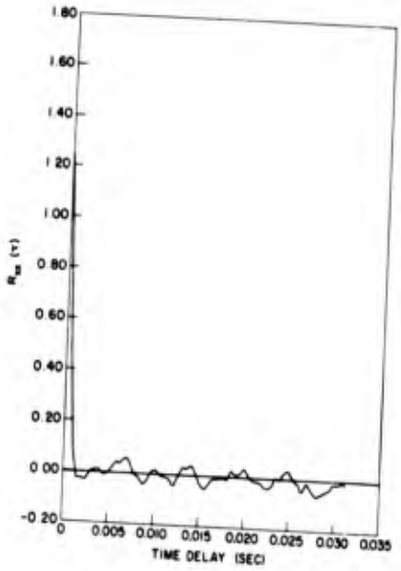


Fig. 10.3a. Autocorrelation of Channel 3 (64 lag).

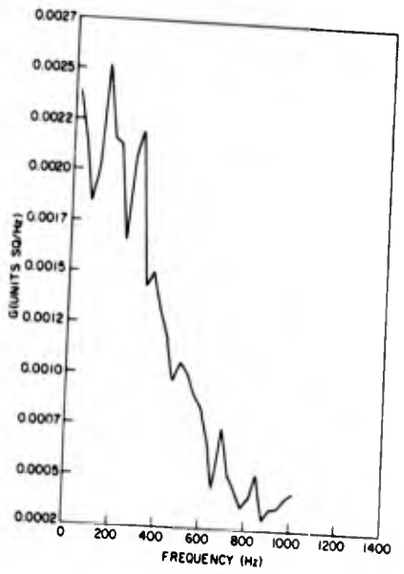


Fig. 10.3b. B-T PSD of Channel 3 (64 d.f.).

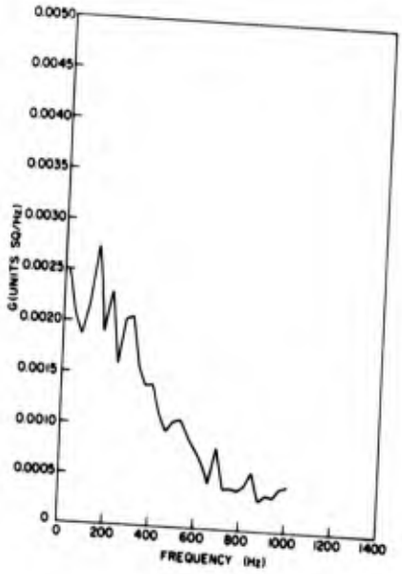


Fig. 10.3c. FFT PSD of Channel 3 (64 d.f.).

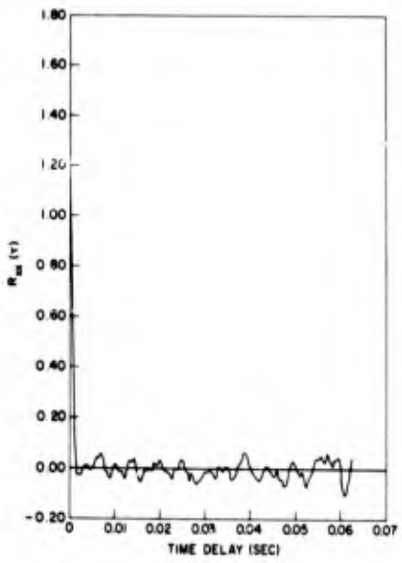


Fig. 10.4a. Autocorrelation of Channel 3 (128 lag).

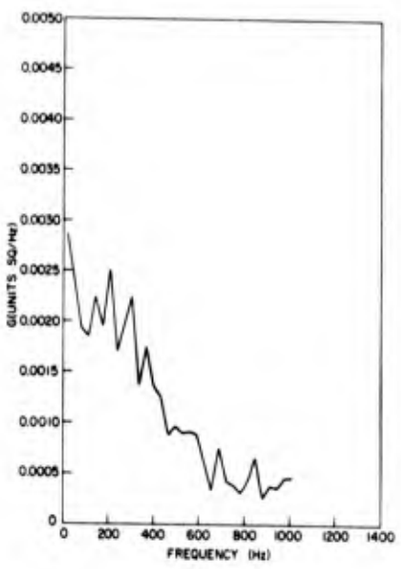


Fig. 10.4b. B-T PSD of Channel 3 (32 d.f.).

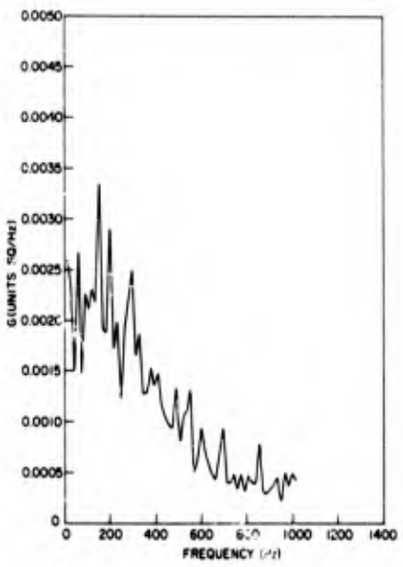


Fig. 10.4c. FFT PSD of Channel 3 (32 d.f.).

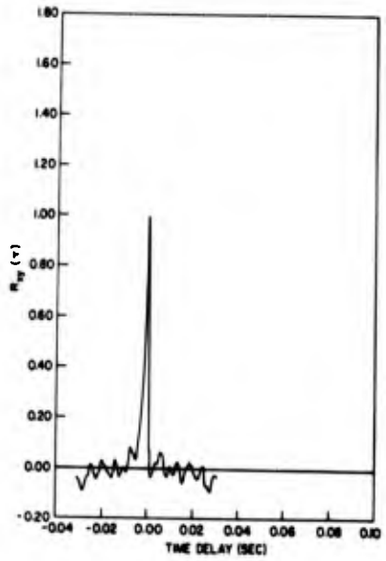


Fig. 10.5a. Crosscorrelation of Channels 8 and 3.

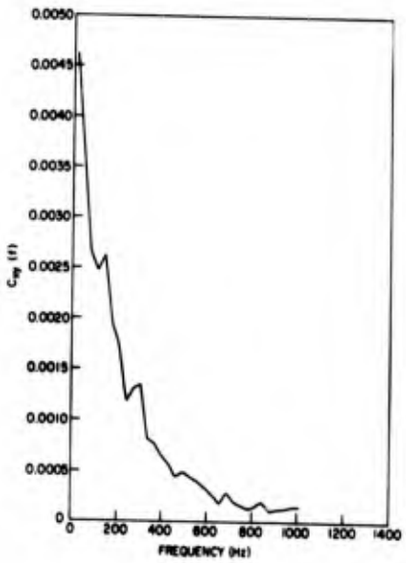


Fig. 10.5b. B-T CSD of Channels 8 and 3.

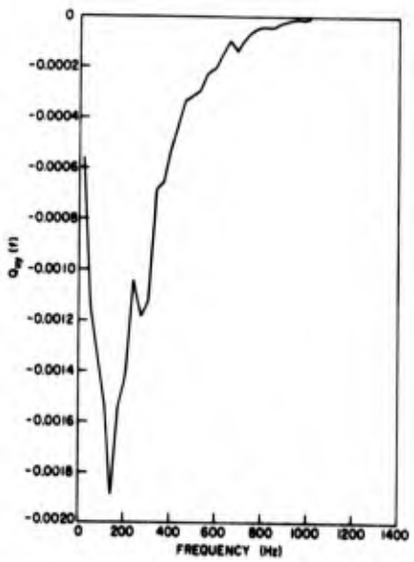


Fig. 10.5c. B-T QSD of Channels 8 and 3.

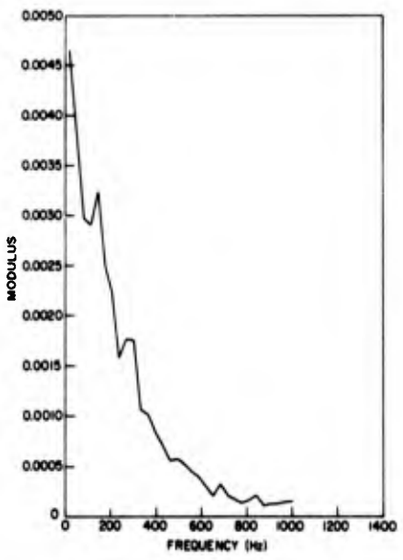


Fig. 10.6a. Modulus of B-T CSD of Channels 8 and 3.

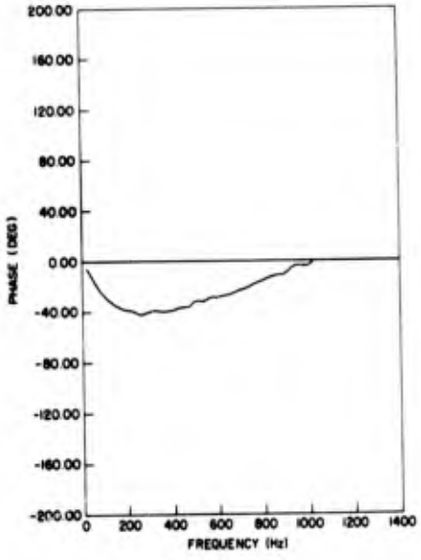


Fig. 10.6b. Phase of B-T QSD of Channels 8 and 3.

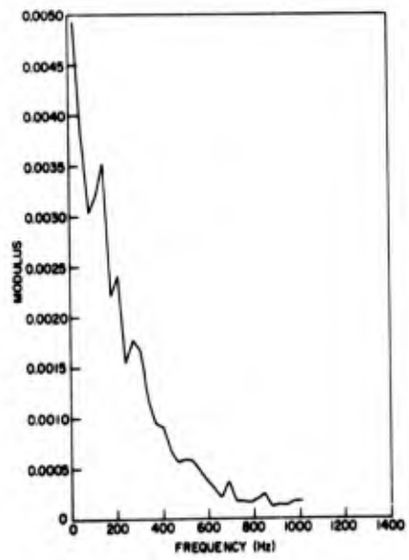


Fig. 10.6c. Modulus of FFT CSD of Channels 8 and 3.

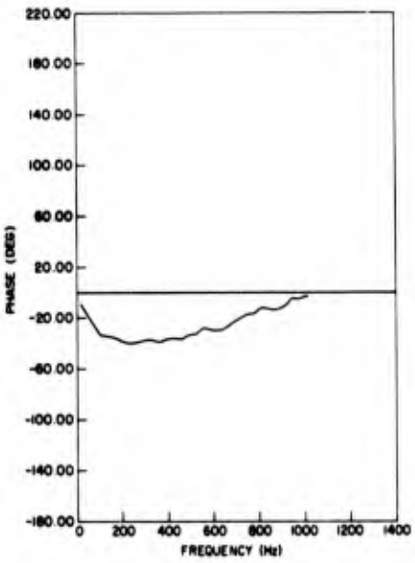


Fig. 10.6d. Phase of FFT QSD of Channels 8 and 3.

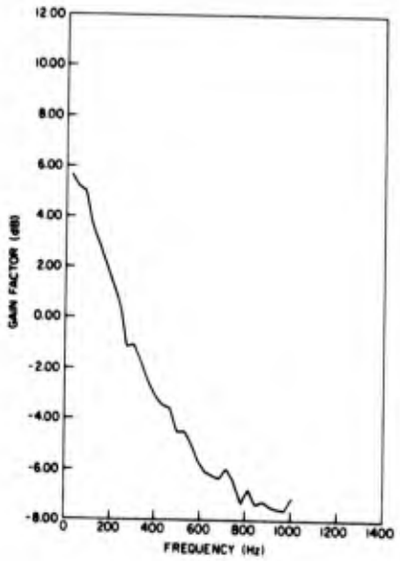


Fig. 10.7a. B-T Version of gain factor (Ch. 2).

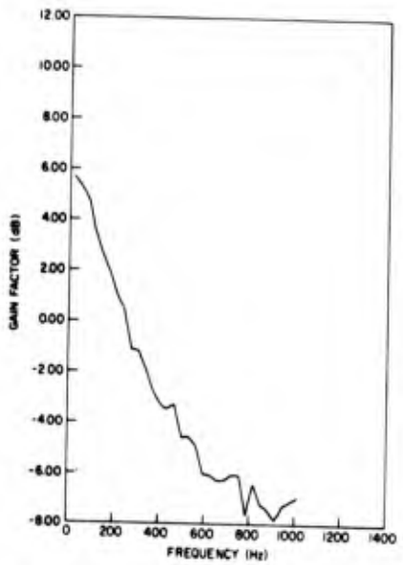


Fig. 10.7b. FFT Version of gain factor (Ch. 3).

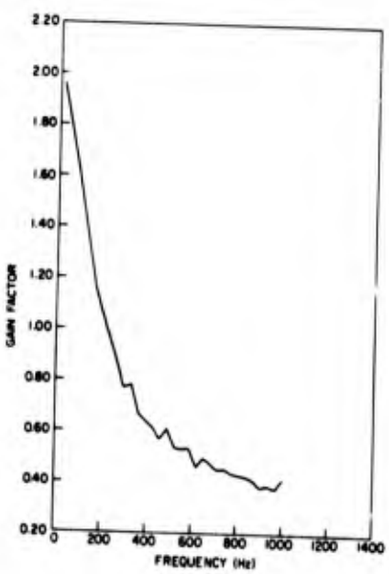


Fig. 10.7c. FFT Version of gain factor (Chs. 8 and 3).

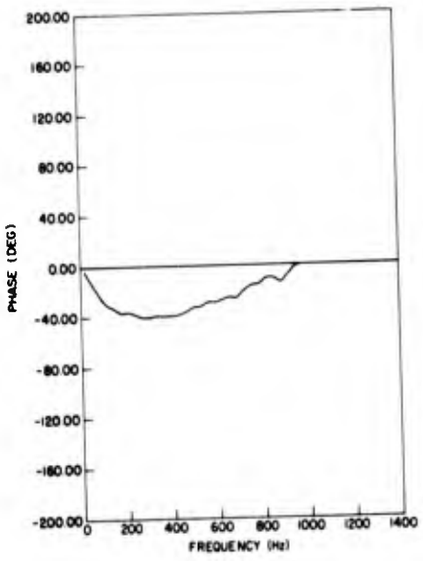


Fig. 10.8a. B-T version of phase factor (Ch. 2).

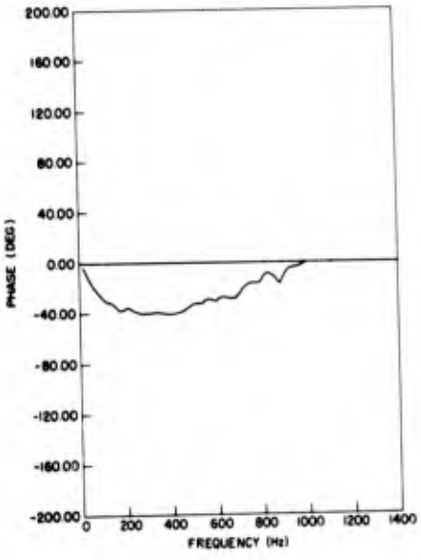


Fig. 10.8b. FFT version of phase factor (Ch. 3).

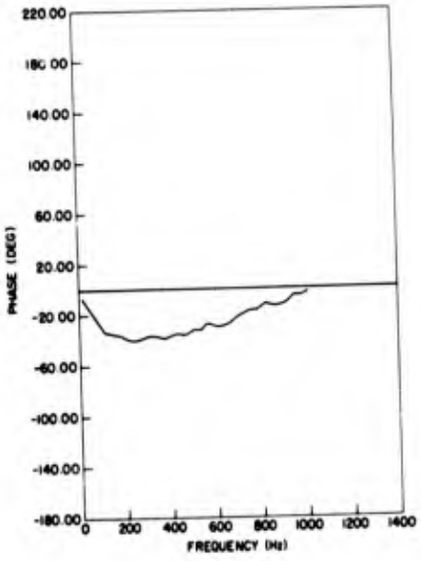


Fig. 10.8c. FFT version of phase factor (Chs. 8 and 3).

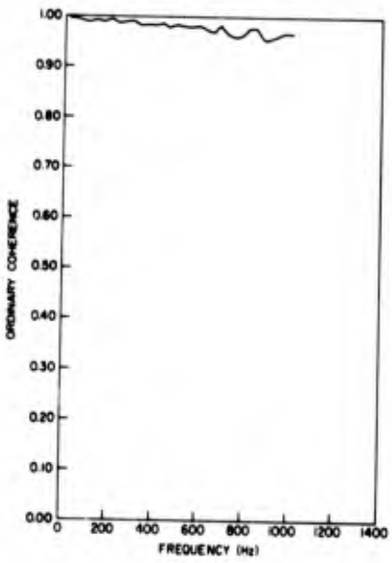


Fig. 10.9a. B-T version, ordinary coherence function (Ch. 2).

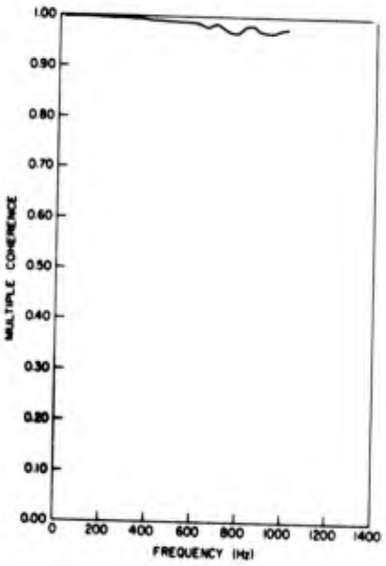


Fig. 10.9b. B-T version, multiple coherence function (Ch. 2).

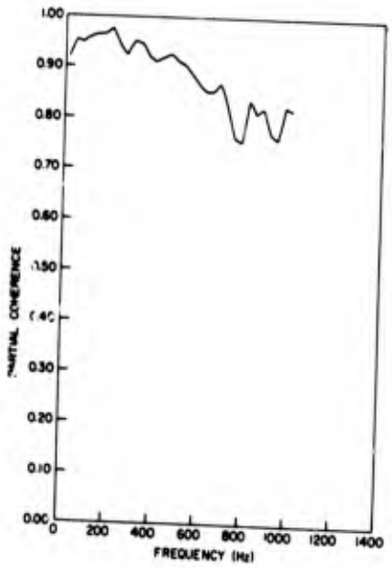


Fig. 10.9c. B-T version, partial coherence function (Ch. 2).

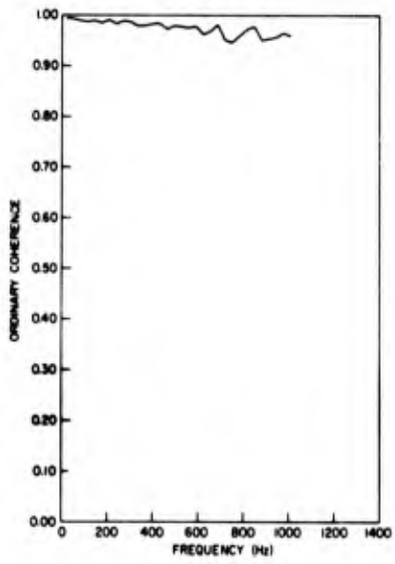


Fig. 10.10a. FFT version, ordinary coherence function (Ch. 2).

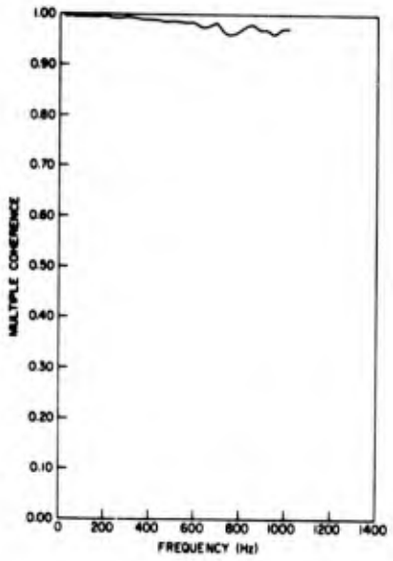


Fig. 10.10b. FFT version, multiple coherenc function (Ch. 2).

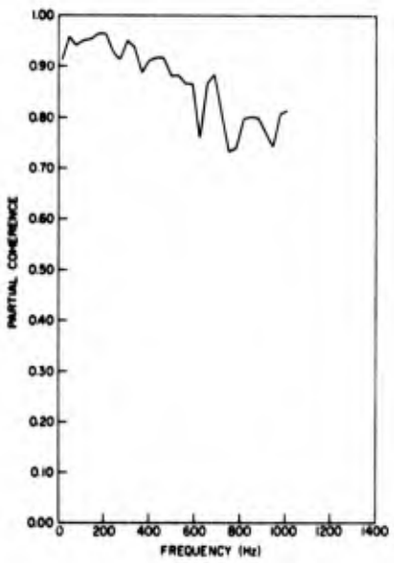


Fig. 10.10c. FFT version, partial coherence function (Ch. 2).

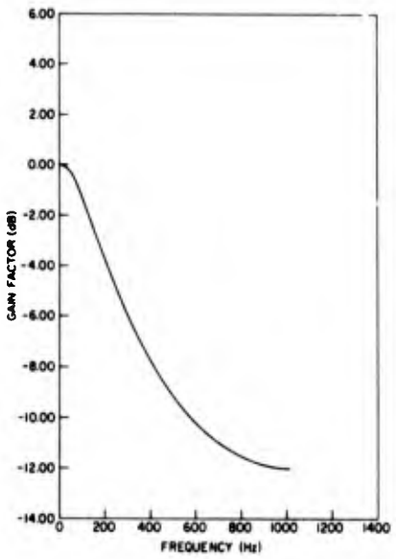


Fig. 10.11a. Fig. 10.7a when coherence is unity.

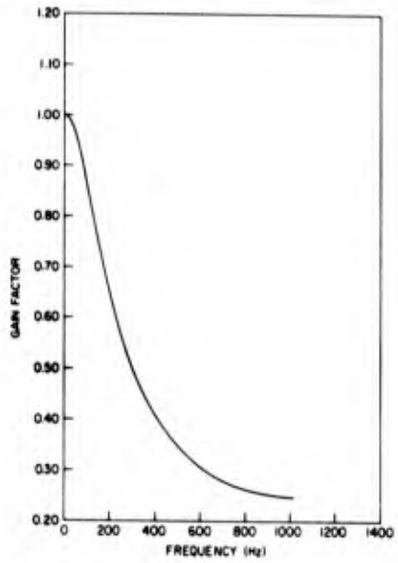


Fig. 10.11b. Fig. 10.7b when coherence is unity.

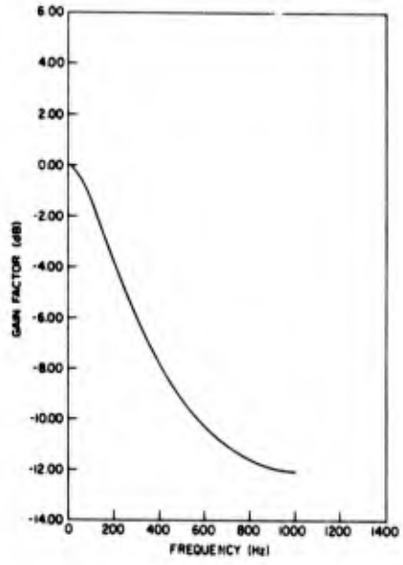


Fig. 10.11c. Fig. 10.7c when coherence is unity.

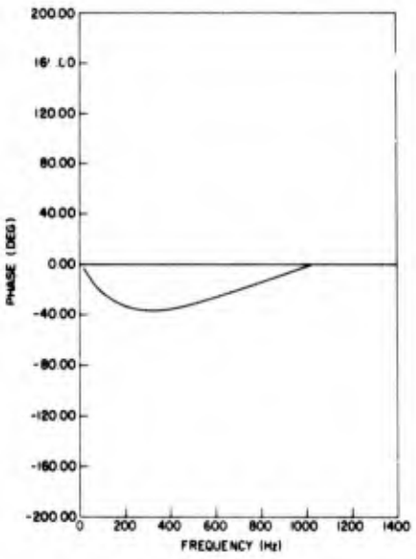


Fig. 10.12a. Fig. 10.8a when coherence is unity.

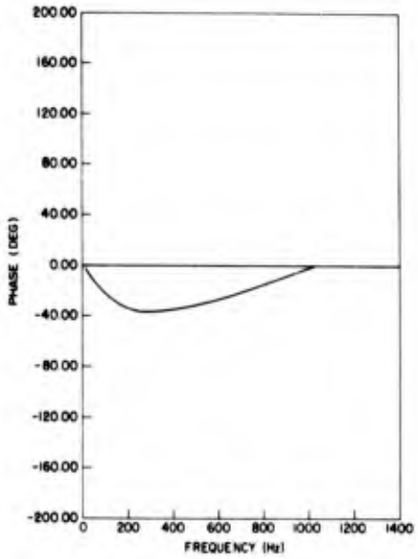


Fig. 10.12b. Fig. 10.8b when coherence is unity.

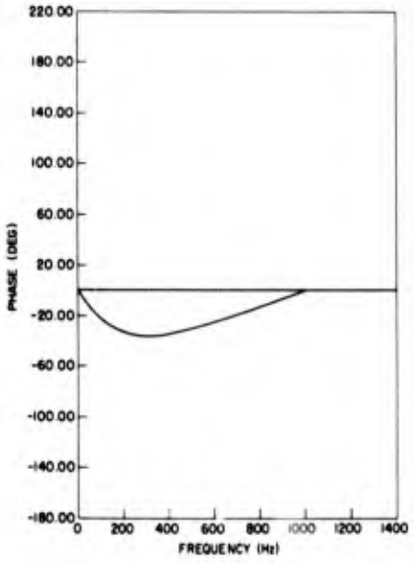


Fig. 10.12c. Fig. 10.8c when coherence is unity.

APPENDIX A
GLOSSARY OF ABBREVIATIONS AND SYMBOLS

ADC	Analog-to-digital converter
B	Bandwidth, folding frequency
B_e	Bandwidth of analysis, effective bandwidth
B–T	Blackman-Tukey
BCD	Binary coded decimal
c	Arbitrary constant, digital filter weight
c_k	Complex coefficients of Fourier series
C	Electrical capacitance, maximum output of a limiter
$C_{xy}(f)$	One-sided cospectral density function
CSD	Cross spectral density
C–T	Cooley-Tukey; authors of an early version of the FFT
DAS	Data acquisition system
d.f.	Degrees of freedom
e	Voltage input
E[ ]	Expected value of
exp[ ]	$e^{[\ ]}$, $e = 2.718\ldots$
f	Frequency
$f(x)$	Probability density function
FFT	Fast Fourier transform
$F(x)$	Probability distribution function
$\mathcal{F}[\ ]$	Fourier transform of [ ]
$\mathcal{F}^{-1}[\ ]$	Inverse Fourier transform of [ ]
$F_{n_1,n_2;a}$	a percentage point of the F distribution with n_1 and n_2 d.f.
FM	Frequency modulation
$G_x(f)$	One-sided power spectral density
$G_{xy}(f)$	One-sided cross spectral density
G_{xk}	One-sided power spectral density at frequency $f = k\Delta f$
G_{xyk}	One-sided cross spectral density at frequency $f = k\Delta f$
$h(\tau)$	Weighting function (unit impulse response function)
h_k	Digital filter weight
$H(f)$	Frequency response function [the Fourier transform of $h(\tau)$]
$\lvert H(f)\rvert$	Gain
Hz	Hertz, the units of frequency
Im[ ]	Imaginary part of [ ]
i	An index
j	$\sqrt{-1}$, an index
k	An index
ℓ	An index
ℓn[ ]	Natural logarithm of [ ]

m	Number of lags in discrete correlation function or discrete convolution
n	Number of degrees of freedom
$n(t)$	Random noise function
N	Sample size or correlation of white noise correlation function
p	Number of inputs in multidimensional linear system
P	Probability
PCM	Pulse code modulation
PDF	Probability density function
PSD	Power spectral density
$Q_{xy}(f)$	One-sided quadrature spectral density
QSD	Quad (quadrature) spectral density
R	Electrical resistance
$R_x(\tau)$	Autocorrelation function
R_{xr}	Discrete autocorrelation function at lag value $\tau = r\Delta\tau$
$R_{xy}(\tau)$	Crosscorrelation function
R_{xyr}	Discrete crosscorrelation function at lag value $\tau = r\Delta\tau$
Re	Real part of [ ]
s	Sample standard deviation
s^2	Sample variance
$S_x(f)$	Two-sided power spectral density
sgn(x)	The sign of x
sps	Samples per second
t	Time variable
$t_{n;a}$	a percentage point of statistical "t" distribution with n d.f.
T	Observation time, record length
T_m	Span of autocorrelation used (one side)
$u(t)$	Boxcar function (time)
$U(f)$	Fourier transform of the boxcar function
$x(t), y(t)$	Time-dependent variables
x_i, y_i	Sampled function at time $t = i\Delta t$
$\hat{x}$	Estimate of x
$\bar{x}$	Sample mean value of $x(t)$
$\tilde{x}$	Raw estimate of x
$X(f)$	Fourier transform of $x(t)$ or x_i
X_k	Fourier transform of x_i
X^2	Sample χ^2 variable
W	Usually exp $[-j2\pi/N]$
W	Fourier transform matrix representation
|[ ]|	Absolute value of [ ]
a	A small probability, significance level, number of counts
$\gamma^2(f)$	Coherence function
γ_{ij}^2	Ordinary coherence function

γ_{ix}	Multiple coherence function
$\gamma^2_{iy/p}$	Partial coherence function
$\Gamma(\ )$	Gamma function evaluated for ( )
$\delta(t)$	Delta function
Δt	Interval between samples
ϵ	Normalized standard error
ϵ^2	Normalized variance
ζ	Damping ratio
ζ_m	mth damping ratio
θ	Phase angle
$\theta_{xy}(f)$	Argument of $G_{xy}(f)$, (phase)
λ	Normalized frequency ($\lambda = 2\pi f \Delta t$)
λ_m	mth natural frequency
Λ_m	Complex frequency (digital)
μ	Mean value
ρ	Correlation coefficient
$\rho_x(\tau)$	Normalized correlation function; correlation coefficient function
σ	Standard deviation
σ^2	Variance
Σ	Indicates summation
τ	Time displacement
$\phi(f)$	Phase factor
$\Phi(x)$	Normal (Gaussian) probability distribution function
$\chi^2_{n\,;\,a}$	A percentage point of statistical chi-square variable with n d.f.
ω	Frequency in radians/sec ($\omega = 2\pi f$)

APPENDIX B
MISCELLANEOUS NUMERICAL EXPRESSIONS

1. Numerical subroutine for $\phi(x)$ (normal distribution):

By $\phi(x)$ is meant

$$\phi(x) = \frac{1}{2\pi} \int_{-\infty}^{x} e^{-t^2/2}.$$

Approximations are readily available for erf (y), the *error* function, whose definition is

$$\text{erf}\,(y) = \frac{2}{\sqrt{\pi}} \int_0^y e^{-t^2}\, dt$$

$$\approx 1 - \frac{1}{\left[1 + \sum_{i=1}^{6} a_i y^i\right]^{16}}$$

where

$$a_i = 0.0705,\ 2307,\ 84$$

$$a_2 = 0.0422,\ 8201,\ 23$$

$$a_3 = 0.0092,\ 7052,\ 72$$

$$a_4 = 0.0001,\ 5201,\ 43$$

$$a_5 = 0.0002,\ 7656,\ 72$$

$$a_6 = 0.0000,\ 4306,\ 38.$$

$\phi(x)$ may be calculated from

$$\phi(x) = \begin{cases} \frac{1}{2} + \frac{1}{2}\,\text{erf}\left(\frac{x}{\sqrt{2}}\right) & x \geqslant 0 \\ \frac{1}{2} - \frac{1}{2}\,\text{erf}\left(\frac{-x}{\sqrt{2}}\right) & x < 0. \end{cases}$$

As usual,

$$\frac{1}{\sqrt{2\pi\sigma^2}}\int_a^b e^{-\frac{(x-\mu)^2}{2\sigma^2}}\,dx = \phi\left(\frac{b-\mu}{\sigma}\right) - \phi\left(\frac{a-\mu}{\sigma}\right).$$

The approximation for erf (x) is due to Hastings, Ref. 48.

2. Useful formulas

$$e^{j\theta} = \cos\theta + j\sin\theta$$

$$e^{-j\theta} = \cos\theta - j\sin\theta$$

$$\cos\theta = \frac{e^{j\theta} + e^{-j\theta}}{2}$$

$$\sin\theta = \frac{e^{j\theta} - e^{-j\theta}}{2j}$$

$$1 + a + a^2 + \ldots + a^r = \frac{1 - a^{r+1}}{1 - a}, \qquad a \neq 1$$

$$e^{j\theta} + e^{2j\theta} + \ldots + e^{Nj\theta} = \frac{1 - e^{jN\theta}}{1 - e^{j\theta}}\, e^{j\theta}$$

$$\sum_{n=1}^{N} \sin n\theta = \frac{\sin\frac{N\theta}{2}\ \sin\frac{(N+1)\theta}{2}}{\sin\frac{\theta}{2}}$$

$$\sum_{n=1}^{N} \cos n\theta = \frac{\cos\frac{N\theta}{2}\ \sin\frac{(N+1)\theta}{2}}{\sin\frac{\theta}{2}}$$

$$\sum_{n=-N}^{N} e^{jn\theta} = \sum_{n=-N}^{N} \cos n\theta = \frac{\sin\left[\left(\frac{2N+1}{2}\right)\theta\right]}{\sin\frac{\theta}{2}}$$

$$\sin\alpha \pm \sin\beta = 2\sin\left[\frac{1}{2}(\alpha \pm \beta)\right]\cos\left[\frac{1}{2}(\alpha \mp \beta)\right]$$

$$\cos \alpha + \cos \beta = 2 \cos \left[\frac{1}{2}(\alpha+\beta)\right] \cos \left[\frac{1}{2}(\alpha-\beta)\right]$$

$$\cos \alpha - \cos \beta = -2 \sin \left[\frac{1}{2}(\alpha+\beta)\right] \sin \left[\frac{1}{2}(\alpha-\beta)\right]$$

$$\cos \alpha \cos \beta = \frac{1}{2}[\cos(\alpha+\beta) + \cos(\alpha-\beta)]$$

$$\sin \alpha \sin \beta = \frac{1}{2}[\cos(\alpha-\beta) - \cos(\alpha+\beta)]$$

$$\sin \alpha \cos \beta = \frac{1}{2}[\sin(\alpha+\beta) + \sin(\alpha-\beta)]$$

$$1 - \cos \theta = 2 \sin^2 \frac{\theta}{2}$$

$$1 + \cos \theta = 2 \cos^2 \frac{\theta}{2}$$

$$\sin(\alpha \pm \beta) = \sin \alpha \cos \beta \pm \cos \alpha \sin \beta$$

$$\cos(\alpha \pm \beta) = \cos \alpha \cos \beta \mp \sin \alpha \sin \beta$$

REFERENCES

1. E. J. Hannan, *Time Series Analysis,* John Wiley & Sons, Inc., New York, 1960.
2. N. Wiener, "Generalized Harmonic Analysis," Acta Mathematica, 55, 117, 1930.
3. M. J. Lighthill, *Introduction to Fourier Analysis and Generalised Functions,* Cambridge University Press, Cambridge, England, 1960.
4. G. A. Korn and T. M. Korn, *Mathematical Handbook for Scientists and Engineers,* McGraw-Hill Book Co., Inc., New York, 1961.
5. J. S. Bendat and A. G. Piersol, *Measurement and Analysis of Random Data,* John Wiley & Sons, Inc., New York, 1966.
6. N. Wiener, *Extrapolation, Interpolation, and Smoothing of Stationary Time Series,* MIT Press, Cambridge, 1949.
7. C. E. Shannon, "Communication in the Presence of Noise," *Proc. IRE,* 37, 10 (Jan. 1949).
8. F. A. Jenkins and H. E. White, *Fundamentals of Optics,* 2d ed., McGraw-Hill Book Co., Inc., New York, 1950.
9. R. B. Blackman and J. W. Tukey, *The Measurement of Power Spectra from the Point of View of Communications Engineering,* Dover Publications, New York, 1958.
10. J. G. Truxal, *Automatic Feedback Control System Synthesis,* McGraw-Hill Book Co., Inc., New York, 1955.
11. J. M. Salzer, "Frequency Analysis of Digital Computers Operating in Real Time," *Proc. IRE,* 42 (2), 457-466 (Feb. 1954).
12. J. F. A. Ormsby, "Design of Numerical Filters with Applications to Missile Data Processing," *J. Ass. Computing Mach.* (July 1961).
13. Daniel Brown, private communication.
14. S. Butterworth, "On the Theory of Filter Amplifiers," *Exp. Wireless,* 7, 536-541 (Oct. 1930).
15. H. Holtz and C. T. Leondes, "The Synthesis of Recursive Digital Filters," *J. Ass. Computing Mach.,* 13, 262 (Apr. 1966).
16. R. K. Otnes, "Notes on Digital Filtering in the Frequency Domain," Measurement Analysis Corporation Technical Report 700-4, 1967.
17. D. R. Brillinger, "A Property of Low-Pass Filters," *SIAM Review,* 7, 65 (Jan. 1965).
18. J. W. Cooley and J. W. Tukey, "An Algorithm for the Machine Calculation of Complex Fourier Series," *Math. Comput.,* 19, 297 (Apr. 1965).
19. W. M. Gentleman and G. Sande, "Fast Fourier Transforms–for Fun and Profit," *AFIPS Conference Proc.,* 29, 563-578 (1966), American Federation of Information Processing Societies, New York; Spartan Books, Washington, D.C.
20. G. D. Bergland, "The Fast Fourier Transform Recursive Equations for Arbitrary Length Records," *Math. Comput.,* 21, 236 (Apr. 1967).
21. D. W. McCowan, "Finite Fourier Transform Theory and Its Application to the Computation of Convolutions, Correlations, and Spectra," Earth Sciences Division of Teledyne Industries, Inc., Research Department Technical Memorandum No. 8-66 (Dec. 1966).
22. G. Sande, "On an Alternative Method for Calculating Covariance Functions," *Princeton Comput. Center Mem.,* Princeton, N.J. (1965).
23. S. M. Simpson, Jr., "Time Series Techniques Applied to Underground Nuclear Detection and Further Digitize Seismic Data," MIT Scientific Report No. 2, AFCRL 62-262, Contract AF 19(604)-7378, ARPA Order No. (80-61) (Dec. 1961).

24. S. Weinreb, "A Digital Spectral Analysis Technique and Its Application to Radio Astronomy," MIT Research Laboratory of Electronics, Technical Report 412 (Aug. 1963).
25. M. Hinich, "Estimation of Spectra After Hard Clipping of Gaussian Processes," Technometrics, 9, 391 (Aug. 1967).
26. G. W. Bordner, C. J. Greaves, and W. W. Wierwille, "Research Studies of Random Process Theory and Physical Applications," NASA CR-61081 (Aug. 1964).
27. L. P. Schmid, "Efficient Autocorrelation," *Commun. ACM*, 8, 115 (Feb. 1965).
28. R. D. Kelly, L. D. Enochson, and L. A. Rondinelli, "Techniques and Errors in Measuring Cross-Correlation and Cross-Spectral Density Functions," NASA CR-74505 (Feb. 1966).
29. A. Y. Khinchin, "Korrelationstheorie der stationaren stochastischen Prozesse," *Mathematische Annalen*, 109, 604 (1934).
30. T. A. Magness, "Estimating the Power Spectrum of a Stochastic Process," Ramo-Wooldridge Corporation Report No. GM-TN-17, May 18, 1956.
31. E. Parzen, "Mathematical Considerations in the Estimation of Spectra," *Technometrics*, 3, 167-190 (May 1961).
32. C. Bingham, M. D. Godfrey, and J. W. Tukey, "Modern Techniques of Power Spectrum Estimation," *IEEE Trans. Audio and Electroacoustics*, AU-15, 56-66 (June 1967).
33. R. K. Otnes, "B-66B Low Level Gust Study, Vol. 1, Technical Analysis," Appendix VII (only), WADD TR 60-305, Wright Air Development Division, ARDC, Wright-Patterson AFB, Ohio (March 1961).
34. W. B. Davenport, Jr., and W. L. Root, *An Introduction to the Theory of Random Signals and Noise*, McGraw-Hill Book Co., Inc., New York, 1958.
35. C. Lanczos, *Applied Analysis*, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1956.
36. A. Ralston, *A First Course in Numerical Analysis*, McGraw-Hill Book Co., Inc., New York, 1965.
37. M. J. Alexander and C. A. Vok, "Tables of the Cumulative Distribution of Sample Multiple Coherence," Rocketdyne Division, North American Aviation, Inc., Research Memorandum 972-351 (Oct. 1963).
38. L. D. Enochson and N. R. Goodman, "Gaussian Approximations to the Distribution of Sample Coherence," AFFDL TR-65-57, Research and Technology Division, AFSC, Wright-Patterson AFB, Ohio (Feb. 1965).
39. N. R. Goodman, "Measurement of Matrix Frequency Response Functions and Multiple Coherence Functions," AFFDL TR-65-56, Research and Technology Division, AFSC, Wright-Patterson AFB, Ohio (Feb. 1965).
40. M. Abramowitz and I. A. Stegun, "Handbook of Mathematical Functions," U.S. Department of Commerce, NBS Applied Mathematics Series 55, U.S. Government Printing Office, Washington, D.C., 1964.
41. M. G. Kendall and A. Stuart, *The Advanced Theory of Statistics*, Hafner Publishing Company, New York, 1961.
42. A. G. Piersol, "Power Spectra Measurements for Spacecraft Vibration Data," *J. Spacecraft and Rockets*, 4, 1613 (Dec. 1967).
43. J. S. Bendat, L. D. Enochson, G. H. Klein, and A. G. Piersol, "Advanced Concepts of Stochastic Processes and Statistics for Flight Vehicle Vibration Estimation and Measurement," Air Force ASD-TDR-62-973 (Dec. 1962).
44. E. F. Chiburis and W. C. Dean, "Multiple Coherence of Long Period Seismic Noise at LASA," Teledyne, Inc., Seismic Data Laboratory Report No. 189, June 23, 1967.
45. ———, "Multiple Coherence of Short Period Noise at LASA," Teledyne, Inc., Seismic Data Laboratory Report No. 190, June 26, 1967.
46. ———, "Multiple Coherence of Noise at Three Vertical Arrays, UBSO, GV-TX, AP-OK," Teledyne, Inc., Seismic Data Laboratory Report No. 191, June 28, 1967.

47. E. F. Chiburis and W. C. Dean, "Multiple Coherence of Short Period Noise at UBSO and TFSO," Teledyne, Inc., Seismic Data Laboratory Report No. 192, June 30, 1967.
48. C. Hastings, *Approximations for Digital Computers*, Princeton University Press, Princeton, N.J., 1955.

SUBJECT AND AUTHOR INDEX

Please note that the author entries appear in italics. The first number (in brackets) following the entry is the reference number. The second number is the page on which the reference was first cited.

☆ U. S. GOVERNMENT PRINTING OFFICE : 1969 O - 354-249