NASA TECHNICAL MEMORANDUM

NASA TM X-2988

NASA TM X-2988

PROGRAM FOR THE ANALYSIS OF TIME SERIES

by

Thomas J. Brown

Langley Directorate
U.S. Army Air Mobility R&D Laboratory

Christine G. Brown and Jay C. Hardin

Langley Research Center
Hampton, Va. 23665

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • SEPTEMBER 1974

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA TM X-2988		

4. Title and Subtitle	5. Report Date
PROGRAM FOR THE ANALYSIS OF TIME SERIES	September 1974
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Thomas J. Brown (Langley Directorate, U.S. Army Air Mobility R&D Laboratory), Christine G. Brown, and Jay C. Hardin	L-9278
	10. Work Unit No. 501-04-01-01
9. Performing Organization Name and Address NASA Langley Research Center Hampton, Va. 23665	11. Contract or Grant No.
	13. Type of Report and Period Covered Technical Memorandum
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D.C. 20546	14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

A digital computer program for the Fourier analysis of discrete time data is described. The program is designed to handle multiple channels of digitized data on general purpose computer systems. It is written, primarily, in a version of FORTRAN II currently in use on Control Data Corporation (CDC) 6000 series computers. Some small portions are written in CDC COMPASS, an assembler level code. However, functional descriptions of these portions are provided so that the program may be adapted for use on any facility possessing a FORTRAN compiler and random-access capability.

Properly formatted digital data are windowed and analyzed by means of a fast Fourier transform algorithm to generate the following functions: (1) auto and/or cross power spectra, (2) autocorrelations and/or cross correlations, (3) Fourier coefficients, (4) coherence functions, (5) transfer functions, and (6) histograms.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Fast Fourier transform Time series analysis Fourier analysis Digital data analysis	Unclassified – Unlimited STAR Category 19

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	116	$4.50

*For sale by the National Technical Information Service, Springfield, Virginia 22151

CONTENTS

PROGRAM FOR THE ANALYSIS OF TIME SERIES

By Thomas J. Brown
Langley Directorate, U.S. Army Air Mobility R&D Laboratory

Christine G. Brown and Jay C. Hardin
Langley Research Center

SUMMARY

A digital computer program for the Fourier analysis of discrete time data is described. The program is designed to handle multiple channels of digitized data on general purpose computer systems. It is written, primarily, in a version of FORTRAN II currently in use on Control Data Corporation (CDC) 6000 series computers. Some small portions are written in CDC COMPASS, an assembler level code. However, functional descriptions of these portions are provided so that the program may be adapted for use on any facility possessing a FORTRAN compiler and random-access capability.

Properly formatted digital data are windowed and analyzed by means of a fast Fourier transform algorithm to generate the following functions: (1) auto and/or cross power spectra, (2) autocorrelations and/or cross correlations, (3) Fourier coefficients, (4) coherence functions, (5) transfer functions, and (6) histograms.

One of four standard data windows may be selected for application to the input data, and a filter, as described by the user, may be applied to the spectral data prior to output or generation of correlation functions. The output, as selected by the user, is written on a binary file for further processing or for user-designed graphics. Output is also printed in tabular form and/or fanfold-plot form as desired.

The basic theory employed in the design of the program is described in sufficient detail to permit the user to make appropriate choices from the options available.

INTRODUCTION

Although there have been many computer programs written for the purpose of time-series analysis, each program depends upon the type of data and the objectives of the research. Standard Fourier series routines are useful in describing deterministic periodic functions. Through a slight generalization, a similar routine can be employed in the analysis of transient deterministic functions. However, when the signal is considered to be random, another type of analysis, based upon the concept of power spectra, must be

utilized. Most power-spectral techniques compute the average lagged product or correlation function of the input signal, which is quite expensive in terms of time and storage. Further, many of these programs are essentially research tools and are inefficient for the analysis of large quantities of data.

The recent advent of the fast Fourier transform algorithms has revolutionized the field of time-series analysis. By the proper use of these algorithms, a single program can be developed to handle the three types of functions discussed in the previous paragraph. Further, in the case of random processes, the average lagged product is no longer required. Thus, such a program is much more efficient than those employing the older technique. For this reason, digital analysis of large quantities of data becomes a practicality.

This report presents a computer program for the digital analysis of random and deterministic time series. The program (PATS) is written in a version of FORTRAN II currently in use on Control Data Corporation (CDC) 6000 series machines. It employs the fast Fourier transform and the concept of block averaging to improve statistical variability. It is intended for use by the practicing engineer who desires a minimum of involvement with the mechanics of time-series analysis. It does, however, require that the user possess a working knowledge of the theory of time-series analysis to obtain meaningful results in an optimal fashion; therefore, aspects of the theory required for operation of the program are discussed in this report. Only the actual equations used in the program are presented in the body of the report, as the basic theory is well documented. Additional information and background may be found in references 1 to 5. However, in the cases where the authors were unable to find satisfactory developments of the fundamental equations used, the necessary derivations were included as appendixes.

Properly formatted discrete time data are analyzed by PATS through the use of a fast Fourier transform, from which the following functions are derived:

(1) Auto and/or cross power spectra

(2) Autocorrelations and/or cross correlations

(3) Fourier coefficients

(4) Coherence functions

(5) Transfer functions

(6) Histograms

Power spectra may be filtered in the frequency domain prior to output or further processing with a filter of the user's description. Data may be output on a line printer in tabular form and/or plotted on the fanfold form, as specified by the user. In addition, all output is saved on binary files, which may be processed further or displayed by means of user-designed graphics.

The program was specifically designed to run economically and efficiently in a batch-processing environment from remote terminal equipment. This requirement places a constraint on the size of the program, which in turn limits the maximum resolution obtainable in the frequency domain. In some cases it was prudent to use existing subroutines which are written in CDC COMPASS, an assembler level language. Functional descriptions of these subroutines are provided in an appendix, together with other information necessary to allow a user to adapt PATS to a non-CDC system with random-access capability.

SYMBOLS

A_k, B_k Fourier coefficients

B_{2j} Bernoulli number

e base value for natural logarithms, 2.7182818284

E( ) expectation operator

f(t) continuous function of time

f_j frequency of data points in amplitude bin j

$F(\omega)$ continuous function of ω, Fourier transform of f(t)

$F_T(\omega)$ continuous function of ω, Fourier transform of f(t) defined on the interval T

$G(\omega)$ continuous function of ω, Fourier transform of a linear system response

h(t) continuous function of time

$H(\omega)$ continuous function of ω, Fourier transform of a transfer function subject to input x(t) and output y(t)

$i = \sqrt{-1}$

Im( ) imaginary part of a complex number

j,k,ℓ,m,n	indices
k_{eq}	equivalent number of degrees of freedom
L	number of blocks of time data
M	amplitude of a square wave
N	number of samples of time data per block
N_b	number of amplitude bins
N_t	total number of samples of time data
p	period
$p(\chi^2)$	probability density function, a function of the variable χ^2
$P_f(\omega)$	continuous function of ω, power spectrum of the time function f(t)
P_m	power in the mth 1/3-octave band
$R_f(\tau)$	continuous function of time lag τ, autocorrelation function of f(t)
$R_x(\tau)$	autocorrelation function (eq. (B4))
$R_{xy}(\tau)$	continuous function of time lag τ, cross correlation function of x(t) and y(t)
Re( )	real part of a complex number
$S_f(\omega)$	continuous function of ω, power spectral density of f(t)
S_m	power spectral density in the mth 1/3-octave band
$S_x(\omega)$	continuous function of ω, power spectral density of x(t)
$S_y(\omega)$	continuous function of ω, power spectral density of y(t)
$S_{xy}(\omega)$	continuous function of ω, cross power spectral density of x(t) and y(t)

t time, sec

T time interval of length T seconds

$u_d(t), U_d(\omega)$ data-window transform pair

$u_m(t), U_m(\omega)$ Hamming data-window transform pair

$u_n(t), U_n(\omega)$ Hann data-window transform pair

$u_p(t), U_p(\omega)$ Parzen data-window transform pair

$u_{T/2}(t), U_{T/2}(\omega)$ "boxcar" data-window transform pair

var( ) variance operator

$W = e^{-i2\pi/N}$

W_R data-window correction factor for correlation estimator

W_u data-window correction factor for spectral estimator

$x(t)$ continuous function of time t

$X(\omega)$ continuous function of ω, Fourier transform of $x(t)$

$X_T(\omega)$ continuous function of ω, Fourier transform of $x(t)$ defined on the interval T

$y(t)$ continuous function of time

$Y(\omega)$ continuous function of ω, Fourier transform of $y(t)$

z_j sequence of complex numbers generated from discretized time histories

z_k sequence of complex numbers related to z_j by $z_k = \sum_{j=0}^{N-1} z_j W^{jk}$

α	significance level of a χ^2 distribution
β	factor dependent on window chosen
$\gamma^2_{xy}(\omega)$	continuous function of ω, coherence function of x(t) and y(t)
$\Gamma(\ )$	incomplete gamma function
$\delta(\ )$	Dirac delta function
Δ	change
μ	mean value
ν	positive integer
σ^2	variance of the random process x
ϕ_k	discrete phase angle, deg
χ^2	chi-square random variable
χ^2_c	critical value of χ^2
χ^2_e	expected value of χ^2
ω	frequency, rad/sec
ω_k,ω_n	discrete frequencies, rad/sec
ω_ν,f_ν	Nyquist frequency, rad/sec and Hz, respectively

Mathematical notation:

~,^	estimated quantities
'	new variable
*	complex conjugate operator

( )! factorial operator

ABBREVIATIONS

DFT discrete Fourier transform

FFT fast Fourier transform

PATS program for the analysis of time series

PSD power spectral density

SFT slow Fourier transform

THEORY AND EQUATIONS

Discrete Fourier Transform

Generalized harmonic analysis begins with the calculation of a Fourier transform, which assumes a definition of a transform pair. For the purposes of this program, the transform pair is given by

$$F(\omega) = \frac{1}{2\pi}\int_{-\infty}^{\infty} f(t)e^{-i\omega t}\,dt \tag{1}$$

$$f(t) = \int_{-\infty}^{\infty} F(\omega)e^{i\omega t}\,d\omega \tag{2}$$

When the integral in equation (1) exists, it defines a function, generally complex, known as the Fourier integral, or transform of $f(t)$. The function $f(t)$ is then known as the inverse Fourier transform of $F(\omega)$, and $f(t)$ and $F(\omega)$ are said to be a transform pair.

The finite Fourier transform is an approximation to equation (1) which assumes that $f(t)$ is identically zero outside the region of definition. If $f(t)$ is known on the interval $-T/2$ to $T/2$ continuously, then the finite Fourier transform of $f(t)$ is given by

$$F_T(\omega) = \frac{1}{2\pi}\int_{-T/2}^{T/2} f(t)e^{-i\omega t}\,dt \tag{3}$$

When $f(t)$ is known at N equally spaced discrete points covering the entire interval T, $F_T(\omega)$ may be approximated at the frequencies

$$\omega_k = \frac{2\pi k}{N\,\Delta t} \qquad (k = 0, 1, 2, \ldots, N/2)$$

by the discrete Fourier transform (DFT) given by

$$\hat{F}_T(\omega_k) = (-1)^k \frac{\Delta t}{2\pi} \sum_{j=0}^{N-1} f(j\,\Delta t) e^{-i2\pi jk/N} \tag{4}$$

where Δt is the time sampling rate. This discrete Fourier transform is the basic relation which must be evaluated in all types of digital time-series analysis.

There are two inherent limitations in using the discrete Fourier transform as an estimate of the true Fourier integral. First, the finite Fourier transform assumes that the function for which the transform is desired is zero outside the region T. This introduces an error in resolution which is discussed in a subsequent section, "Data and Spectral Windows." Second, it can be shown that for N input time points, only $N/2$ unique frequency points will be generated. The highest of these $\omega_\nu = \pi/\Delta t$, which occurs when $k = N/2$, is called the Nyquist or folding frequency and is significant in that any energy present in the data with a frequency above ω_ν will appear erroneously at a lower frequency. This phenomenon is known as aliasing and is to be avoided. Since the Nyquist frequency is a function of the sampling rate, aliasing may be reduced by choosing the sampling frequency at twice the highest frequency for which nonnegligible power occurs or by low pass filtering the signal at the Nyquist frequency.

Assuming that the Nyquist frequency has been properly chosen, the discrete Fourier transform (eq. (4)) may be obtained from calculations of the standard relation

$$z_k = \sum_{j=0}^{N-1} z_j W^{jk} \tag{5}$$

where $W = e^{-i2\pi/N}$ and z_j is a sequence of complex numbers. To evaluate equation (5), N^2 operations are required (N multiply-add operations which must be repeated

N times). Implemented in this form, equations (4) and (5) are referred to herein as the slow Fourier transform (SFT).

The fast Fourier transform (FFT) derives its name from its computational efficiency, which requires that $N = 2^n$, where N is the number of points to be transformed and n is an integer. For this choice of N, the number of operations is reduced from N^2 to $2N \log_2 N$, and considerable time is saved. However, the restriction that N be a power of 2 is often undesirable; consequently, both the FFT and the SFT are implemented in PATS and may be used interchangeably as the application requires.

Spectral Representations

As mentioned in the Introduction, there are several types of harmonic analysis in common usage. Which of these is preferred depends roughly upon whether the signal is steady or transient and whether it is considered random or deterministic. However, all these cases may be analyzed by means of the discrete Fourier transform, which was discussed in the previous section.

Fourier coefficients of periodic functions.- Suppose $f(t)$ is periodic with period p. Then $f(t)$ may be represented by the Fourier series

$$f(t) = \frac{A_0}{2} + \sum_{k=1}^{\infty} \left(A_k \cos \omega_k t + B_k \sin \omega_k t \right) \tag{6}$$

where $\omega_k = 2\pi k/p$ are harmonics of the fundamental frequency of the signal. If N samples of this signal at equal intervals Δt are available for a total record length of $T = N \Delta t$ and if $T = \nu p$, where ν is a positive integer, then employing these values in equation (5) yields

$$\left.\begin{aligned} \hat{A}_k &= \frac{2}{N} \operatorname{Re}\left(z_{\nu k}\right) \\ \hat{B}_k &= -\frac{2}{N} \operatorname{Im}\left(z_{\nu k}\right) \end{aligned}\right\} \qquad (k = 0, 1, 2, \ldots, N/2\nu) \tag{7}$$

where $\hat{}$ indicates an estimate of the required quantity. An estimate of the phase ϕ_k at frequency ω_k can also be obtained from

$$\hat{\phi}_k = \arctan \frac{\hat{B}_k}{\hat{A}_k}$$

Two factors should be particularly noted in this representation: First, the total signal length should be a multiple of the period of the signal. If this is not the case, then the frequencies at which the finite transform is evaluated will not correspond to the fundamental frequencies in the signal, and smearing will result. Second, since the Nyquist frequency occurs at the frequency $\omega = \pi N/\nu p$, the frequency content of the signal should be limited by means of a low-pass filter, because aliasing can cause significant errors in the estimate if the signal contains power above the Nyquist frequency. A thorough discussion of this representation is given in appendix A.

Amplitude spectra of transient functions.- If $f(t)$ is a transient function, that is, it begins at a finite time and dies away after some time, it may be represented by the Fourier integral in equation (1):

$$F(\omega) = \frac{1}{2\pi}\int_{-\infty}^{\infty} f(t)e^{-i\omega t}\,dt$$

If N samples of this signal at equal intervals Δt are available, a spectral estimate may be obtained by employing these values in equation (5). Then, by equation (4),

$$\hat{F}_T(\omega_k) = (-1)^k \frac{\Delta t}{2\pi} z_k \qquad (k = 0, 1, 2, \ldots, N/2) \tag{8}$$

where

$$\omega_k = \frac{2\pi k}{N\,\Delta t}$$

Power spectra of random processes.- If the signal $f(t)$ is not considered to be a deterministic function, but merely one member of the ensemble which comprises a random process, the concept of power spectra must be employed to provide a harmonic representation of the function. Strictly speaking, such a representation is valid only when the random process may be said to be both stationary and ergodic. Briefly, this means that the statistics of the process are independent of time (i.e., no change in the mechanism of generation is present) and that each sample function is representative of the whole ensemble.

When these conditions are satisfied, the Fourier integral representation given by equation (1) does not exist, since the function is not square integrable. However, the

finite Fourier transform given by equation (3) does exist and an estimate of the power spectral density of the random process $f(t)$ may be obtained from

$$\hat{S}_f(\omega) = \frac{\pi}{T}\left|F_T(\omega)\right|^2$$

If N values of the function $f(t)$ exist at equally spaced intervals Δt and these are employed in the standard transform (eq. (5)), then the spectral estimate becomes

$$\hat{S}_f(\omega_k) = \frac{\Delta t}{4\pi N}\left|z_k\right|^2 \tag{9}$$

It will be shown in a later section that the factor $\Delta t/4\pi N$ must be modified for other considerations. However, equation (9) does show the basic dependence of the spectral estimate on the standard transform given by equation (5).

It should be noted that if the random process does not satisfy the condition of stationarity, a representation in terms of power spectra is invalid and, in fact, no general representation of reasonable utility exists. If only the condition of ergodicity is violated, the power spectral representation is valid. However, many sample functions must be collected and an ensemble average taken over them. The reader may find a more detailed discussion of stationarity and ergodicity and their implications in reference 5.

In this section, it has been indicated that the three most widely employed spectral representations may all be estimated from the standard transform given by equation (5). In the next few sections, some particular aspects of this technique will be discussed.

Data and Spectral Windows

One inherent limitation in techniques of spectral estimation is that the data input must always be finite in length. This causes a frequency smearing, or lack of resolution. The phenomenon may be analyzed by investigating the relation between the finite Fourier integral $F_T(\omega)$ and the infinite Fourier integral $F(\omega)$.

To do so, rewrite equation (3) as follows:

$$F_T(\omega) = \frac{1}{2\pi}\int_{-\infty}^{\infty} u_{T/2}(t)\, f(t) e^{-i\omega t}\, dt \tag{10}$$

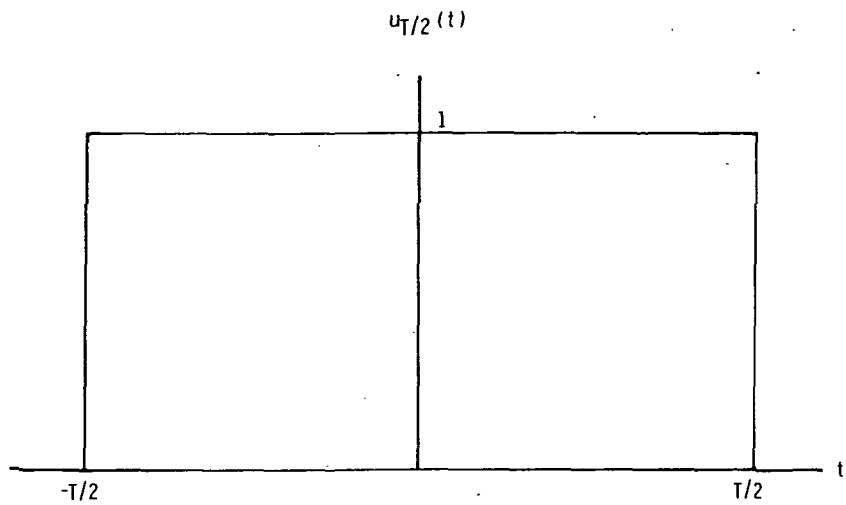


Figure 1.- The boxcar data window.

where $u_{T/2}(t)$ is the "boxcar" data window, as shown in figure 1. From the Fourier frequency convolution theorem, equation (10) may be rewritten as

$$F_T(\omega) = \int_{-\infty}^{\infty} F(\omega')\, U_{T/2}(\omega - \omega')\, d\omega' \tag{11}$$

where $F(\omega)$ is the true transform and the transform of $u_{T/2}(t)$ is given by

$$U_{T/2}(\omega) = \frac{T}{2\pi} \frac{\sin(\omega T/2)}{\omega T/2} \tag{12}$$

Thus, $F_T(\omega)$ is seen to be the weighted average of the values of $F(\omega)$ about $\omega = \omega'$. As can be seen in figure 2, $F_T(\omega)$ is an estimate of $F(\omega)$, the true Fourier transform of $f(t)$. Because of the duality of time-domain multiplication and frequency-domain

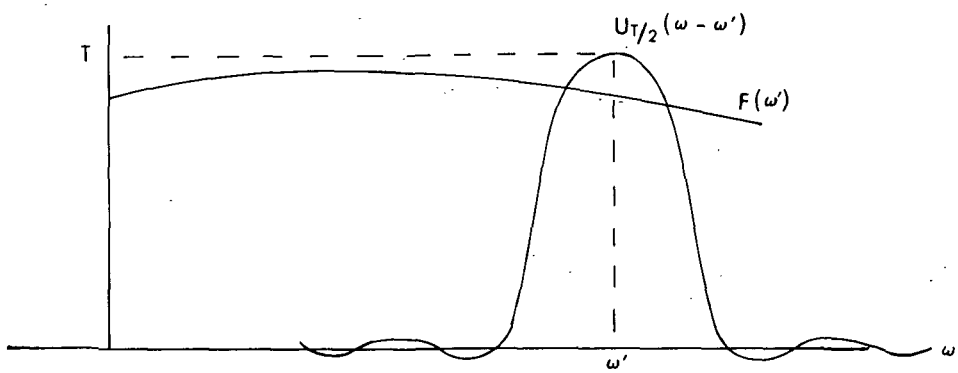


Figure 2.- The average smoothed spectral estimate.

convolution, the finite transform $F_T(\omega)$ at $\omega = \omega'$ is an infinite sum of contributions selected from $F(\omega)$ by $U_{T/2}(\omega)$. The magnitude of these contributions is dependent upon the lobes of $U_{T/2}$ on either side of the maximum, known as side lobes. It is thus desirable to minimize the size of the side lobes of $U_{T/2}$ in order that $F_T(\omega)$ may approximate $F(\omega)$ accurately. It is worthy of note that as $T \to \infty$, the approximation improves. This phenomenon may be seen by reference to equation (12), which shows that the main lobe narrows and the side lobes decrease with increasing T. The finite Fourier transform may thus be considered a smoothed approximation as seen through a window, in this case $U_{T/2}$, which is referred to as the boxcar spectral window because of the characteristic square shape of its transform in the time domain.

These data windows and spectral windows exist because of the finite length of the data over which the user has no control. However, there are data windows for which the side lobes of the corresponding spectral window are lower than for the boxcar window, and the averaging is thus concentrated at points nearer ω'. It is therefore often advantageous to employ one of these windows.

PATS provides three window options in addition to the boxcar. They are the Hann data window given by

$$u_n(t) = \begin{cases} 0 & (t < -T/2) \\ \frac{1}{2}\left(1 + \cos\frac{2\pi t}{T}\right) & (-T/2 \leqq t \leqq T/2) \\ 0 & (t > T/2) \end{cases} \tag{13}$$

the Hamming data window given by

$$u_m(t) = \begin{cases} 0 & (t < -T/2) \\ 0.54 + 0.46\cos\frac{2\pi t}{T} & (-T/2 \leqq t \leqq T/2) \\ 0 & (t > T/2) \end{cases} \tag{14}$$

and the Parzen data window given by

$$u_p(t) = \begin{cases} 1 - 6\left(\frac{2|t|}{T}\right)\left(1 - \frac{2|t|}{T}\right) & (|t| \leqq T/4) \\ 2\left(1 - \frac{2|t|}{T}\right)^3 & (|t| > T/4) \\ 0 & (|t| > T/2) \end{cases} \tag{15}$$

A thorough discussion of relative merits of the Hamming and Hann data windows may be found in reference 3. The Parzen window is, however, unique and warrants some discussion. Reference 2 will show that Parzen windows possess no negative side lobes as do Hamming and Hann windows. This unique feature precludes the presence of negative spectral estimates, which may occur when using either of the latter windows for computing power spectra. This advantage is offset by the complexity of the window and the extra computing time encountered in its use.

Because of the existence of these windows, it is necessary to use a different power spectral estimate from that given by equation (9) in order for the estimate to be power preserving. This new estimate is defined by

$$\tilde{S}_f(\omega_k) = \frac{(\Delta t)^2}{2\pi W_u} \left| z_k \right|^2 \tag{16}$$

where

$$W_u = \int_{-\infty}^{\infty} u_d^2(t)\, dt \tag{17}$$

is a window correction factor. The derivation of this estimate is given in appendix B.

Variation of Estimates

When $f(t)$ is considered to be a random process, the power spectral estimate $\tilde{S}_f(\omega)$ will be a random variable for each frequency ω, because the estimate is calculated from a single sample function while the true spectrum is an average over the entire ensemble. Thus, the estimate will vary about the required value. In order to reduce this variation, PATS uses the concept of "block averaging." The total record length of N_t points is divided into a number of blocks of length N. Then, it is shown in reference 1 that

$$\frac{\mathrm{var}\left[\tilde{S}_f(\omega)\right]}{E^2\left[\tilde{S}_f(\omega)\right]} = \beta \frac{N}{N_t}$$

where E and var indicate the ensemble expectation and variance of the random variable $\tilde{S}_f(\omega)$, respectively, and β is a factor dependent upon the window chosen.

One simple way of assessing this variation is to assume that $\tilde{S}_f(\omega)$ is a chi-square random variable. Then,

$$\frac{\mathrm{var}\left[\tilde{S}_f(\omega)\right]}{\left\{E\left[\tilde{S}_f(\omega)\right]\right\}^2} = \frac{2}{k_{eq}}$$

where k_{eq} is the equivalent number of degrees of freedom of the random variable $\tilde{S}_f(\omega)$, and bounds on the variation may be obtained from the relation (see ref. 5)

$$\frac{k_{eq}\,\tilde{S}_f(\omega)}{\chi^2_{k_{eq}}\left(\frac{\alpha}{2}\right)} < E\left[\tilde{S}_f(\omega)\right] \leqq \frac{k_{eq}\,\tilde{S}_f(\omega)}{\chi^2_{k_{eq}}\left(1 - \frac{\alpha}{2}\right)} \tag{18}$$

with probability $1 - \alpha$, where α is the significance level and

$$\chi^2_{k_{eq}}(\alpha) = b$$

$$\int_b^\infty p\left(\chi^2_{k_{eq}}\right) d\chi^2_{k_{eq}} = \alpha$$

The function $p\left(\chi^2_{k_{eq}}\right)$ is the probability density function for a chi-square random variable with k_{eq} degrees of freedom.

The number of blocks used determines the number of degrees of freedom. For L sequential nonoverlapping blocks, as shown in figure 3, the equivalent number of degrees of freedom k_{eq} is shown in reference 1 to be $\frac{2N_t}{N}$, that is, $\beta = 1$. For (2L - 1) blocks overlapping by 50 percent, as shown in figure 4, reference 1 shows that $k_{eq} \approx \frac{36}{11}\frac{N_t}{N}$, that is, $\beta = \frac{11}{18}$ and is dependent upon the window used. In PATS, k_{eq} is calculated exactly through the use of equations to be found in reference 1.

As can be seen, the number of degrees of freedom available for a fixed record length may be improved slightly by overlapping data blocks by 50 percent. This technique is provided as a program option but is not recommended if the available data are of sufficient length to obtain the desired variance without the use of overlapping.

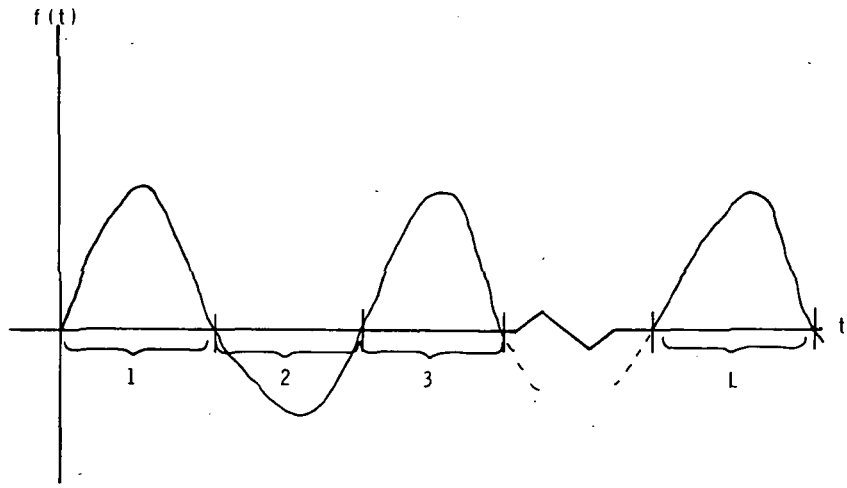


Figure 3.- Nonoverlapping blocks.

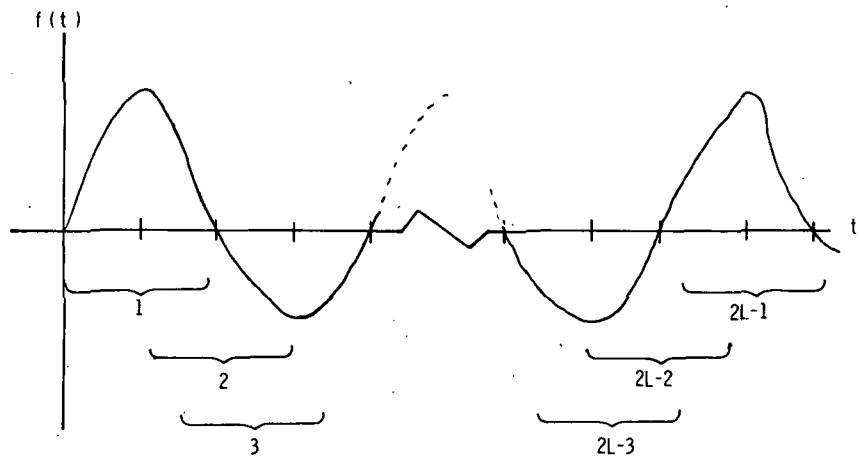


Figure 4.- Blocks overlapping by 50 percent.

One-Third-Octave Power Spectra and General Power Spectra

Power spectra, as differentiated from power spectral densities (PSD), are in general computed by multiplying the PSD by the bandwidth of the estimate. For a sampling rate of Δt and block size of N points, the bandwidth of the estimate is given by $\omega = 2\pi/N\,\Delta t$. Thus, the power spectrum is computed from equation (16) as

$$\hat{P}_f(\omega_k) = \frac{\Delta t}{NW_u}\left|z_k\right|^2 \tag{19}$$

One-third-octave spectra are computed by summing the contributions of the narrow-band spectra given by equation (16) over the 1/3-octave band. Thus, where band m is of width $\Delta\omega_m$, then $\hat{P}_m$, the 1/3-octave power for band m, may be approximated by

$$\hat{P}_m = \sum_{\Delta\omega_m} \hat{P}_f(\omega_k) \tag{20}$$

The 1/3-octave power spectral density is then given by

$$\hat{S}_m = \frac{\hat{P}_m}{2\pi\,\Delta\omega_m} \tag{21}$$

Estimated Cross Power Spectra

The cross power spectral density between two signals $x(t)$ and $y(t)$ is estimated by PATS from the following equation:

$$\tilde{S}_{xy}(\omega_k) = \frac{(\Delta t)^2}{2\pi W_u} z_k z_k' \tag{22}$$

where

$$\left.\begin{aligned} z_k &= \sum_{j=0}^{N-1} x(j\,\Delta t) W^{jk} \\ z_k' &= \sum_{j=0}^{N-1} y(j\,\Delta t) W^{jk} \end{aligned}\right\} \tag{23}$$

Block averaging is used. However, unless the coherence function, to be discussed in a later section, is unity, x and y are unrelated, and techniques for estimation of variance are not currently available. When coherence is near unity, equation (18) may be applied.

Estimated Autocorrelation Functions

The estimated autocorrelation function may be obtained from the PSD estimate as follows:

$$\tilde{R}_f(\tau) = \int_{-\infty}^{\infty} e^{i\omega\tau}\, \tilde{S}_f(\omega)\, d\omega \tag{24}$$

Thus, the estimated autocorrelation function would be computed by applying the inverse FFT to the PSD estimate. This technique is appreciably faster than the method of standard lagged products originated by Blackman and Tukey (ref. 3).

It should be recalled, however, that since only a finite record was utilized in the FFT, a frequency window was introduced in the spectral estimate. As a result, the autocorrelation function computed from equation (24) will be distorted for large values of τ. Thus, it is necessary to introduce a new estimate

$$\tilde{R}_x(\tau) = W_R \int_{-\infty}^{\infty} \tilde{S}_x(\omega) e^{i\omega\tau}\, d\omega \tag{25}$$

where

$$W_R = \frac{\int_{-\infty}^{\infty} u_d{}^2(t)\, dt}{\int_{-\infty}^{\infty} u_d(t)\, u_d(t+\tau)\, dt}$$

Note that the correction factor W_R is a function of the lag τ and the data window $u_d(t)$ chosen. The derivation of this factor may be found in appendix C.

An additional source of error is referred to as circular correlation error, a thorough discussion of which may be found in reference 2. Briefly, because of the periodic nature of the DFT, a correlation function obtained by inverting the power spectrum tacitly assumes that data outside the known interval are repeated periodically. This assumed periodicity introduces errors in the estimates for all values of lag greater than zero, as illustrated in figure 5.

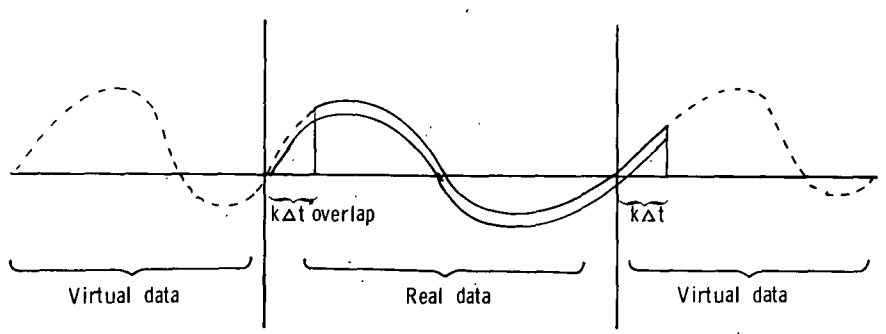


Figure 5.- Illustration of circular correlation error.

For a lag of $k\,\Delta t$, the real and virtual data overlap at k points to introduce an erroneous result. A remedy for this situation is to insert zeros in the last half of the data block, as shown in figure 6. In the situation depicted in figure 6, the virtual data are set to zero; thus, a zero result is produced for the k overlapping points. PATS provides for zero insertion as a selectable computation option.

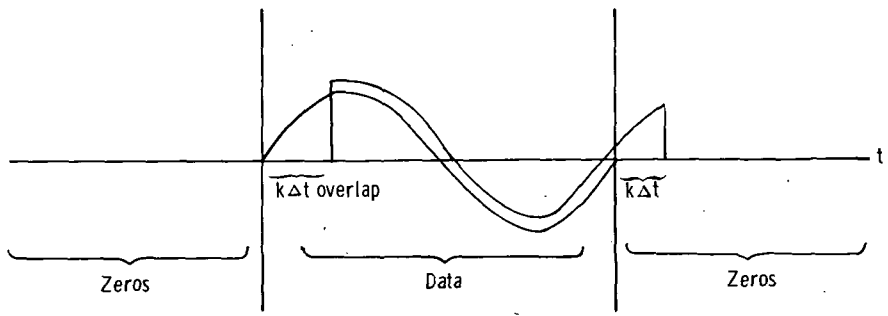


Figure 6.- Illustration of zero insertion for correcting circular error.

Spectral Filtering and Narrow-Band Correlation Functions

Once the Nyquist frequency is chosen and a digital tape generated, a DFT analysis becomes somewhat inflexible. The spectra and correlation functions will contain all the information up to the Nyquist frequency. Often it is desirable to track a narrow band of frequencies, as in the case of time-space correlation studies of structures. This may be accomplished by digital filtering in the frequency domain. If $G(\omega)$ is the transfer function of the desired filter and $X(\omega)$ is the FFT of the input signal, the transform of the filtered signal $Y(\omega)$ is given by $Y(\omega) = X(\omega)\,G(\omega)$. The filtered or smoothed correlation function may then be obtained by application of equations (16) and (24).

PATS permits the user to construct a spectral filter of any general description by selection of a number of points on the desired response curve. A smooth function of frequency is then generated by fitting these points with a set of straight lines and quadratic and cubic curves in such a manner that the result is a continuous function through the first derivative. A thorough explanation of the technique may be found in reference 4. It should be noted that this option can also be employed for prewhitening and postdarkening should the user so desire. A thorough discussion of these filter applications may be found in reference 3.

Estimates of Cross Correlation Functions

Cross correlation functions are computed from cross power spectral densities by inversion through the use of the FFT. That is,

$$R_{xy}(\tau) = W_R \int_{-\infty}^{\infty} \tilde{S}_{xy}(\omega) e^{i\omega t}\, d\omega \tag{26}$$

Zero insertion is optional in the program, although circular errors result if it is not employed. If zero insertion is employed, accurate estimates of both amplitude and phase are obtained with PATS. Spectral filtering for the purpose of smoothing of cross correlation functions, as previously described, is also available. It should be noted that care has been taken so that phase is unaltered by the filtering process.

Transfer and Coherence Functions

The coherence function is a real-valued quantity which may be estimated as

$$\hat{\gamma}^2_{xy}(\omega) = \frac{\left|\tilde{S}_{xy}(\omega)\right|^2}{\tilde{S}_x(\omega)\, \tilde{S}_y(\omega)} \tag{27}$$

where $x(t)$ and $y(t)$ are the input and output of a system, as shown in figure 7. Here $H(\omega)$ is the Fourier transform of the system response $h(t)$. Since

$$\left|\tilde{S}_{xy}(\omega)\right|^2 \leq \tilde{S}_x(\omega)\, \tilde{S}_y(\omega) \tag{28}$$

then $\gamma^2_{xy}(\omega) \leq 1$.

In the event that the system is linear, $\gamma^2_{xy} = 1$ and an estimate of the transfer function of the linear system may then be computed by

$$\tilde{H}(\omega) = \frac{\tilde{S}_{xy}(\omega)}{\tilde{S}_x(\omega)} \tag{29}$$

It should be noted that the estimate of the transfer function is valid only when the coherence is high. Also, the estimation for coherence is highly biased for small statistical accuracy (small number of degrees of freedom). Thus, a large number of blocks should be averaged to get as accurate an estimate of γ^2_{xy} as possible.

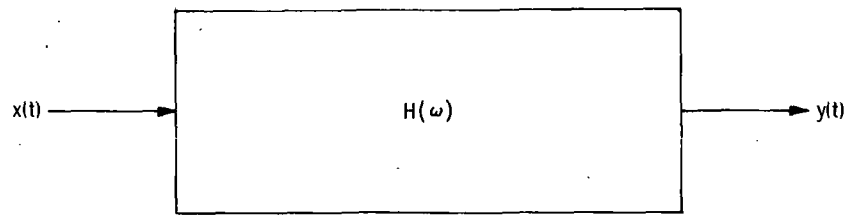


Figure 7.- System response.

Histograms

A histogram is used to obtain estimates of the probability density function of the time data as follows: Consider a block of data as shown in figure 8. An array of amplitude bins is set up by PATS, as shown, and the number of points in each bin f_j is counted from a total sample of N_t points. The resulting histogram is then plotted.

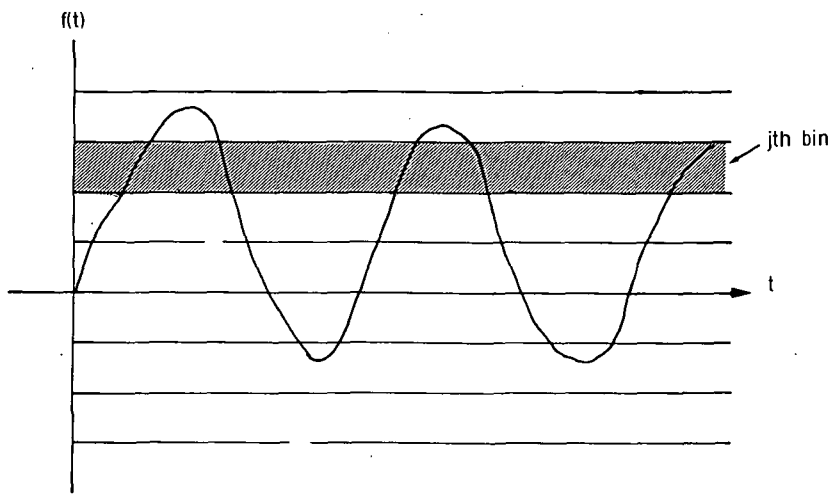


Figure 8.- Amplitude bins for histogram computation.

The hypothesis that the process is Gaussian may also be tested by using the chi-square goodness-of-fit tests. To do so, the Gaussian distribution with the sample mean $\hat{\mu}$ and sample variance $\hat{\sigma}^2$ is generated. The sample mean $\hat{\mu}$ is estimated from the original data as $\hat{\mu} = \sum_{j=1}^{N_t} \frac{x_j}{N_t}$ and the sample variance $\hat{\sigma}^2$ as $\hat{\sigma}^2 = \frac{1}{N_t - 1} \sum_{j=1}^{N_t} \left(x_j - \hat{\mu}\right)^2$.

The bin frequency f_j is then subtracted from the expected frequency $N_t p_j$, where p_j is the Gaussian probability of the jth interval. The sum of the squares of these differences is compared with the χ^2 distribution with $k_{eq} = \left(N_b - 3\right)$ degrees of freedom,

where N_b is the number of bins selected. The effective value of χ^2, or χ_e^2, is computed by

$$\chi_e^2 = \sum_{j=1}^{N_b} \frac{\left(f_j - N_t p_j\right)^2}{N_t p_j} \tag{30}$$

The probability density functions of the χ^2 random variable are given by

$$p\left(\chi^2\right) = 2^{-n/2}\,\Gamma^{-1}\left(\frac{k_{eq}}{2}\right)\left(\chi^2\right)^{\frac{k_{eq}}{2}-1} e^{-\chi^2/2} \tag{31}$$

where n is the number of degrees of freedom. The critical value of χ^2, or χ_c^2, for the α significance level may be found from equation (31). The normality hypothesis is rejected when $\chi_e^2 > \chi_c^2$. The decision to reject the hypothesis, however, is left to the user, as the value of χ_e^2 is quite sensitive to N_b. A detailed description of hypothesis testing using the χ^2 distribution may be found in reference 5.

PROGRAM DESCRIPTION

Operating Environment

PATS was developed for use on the Langley Research Center CDC 6000 Operating System. It is written in CDC FORTRAN and uses some library subroutines written in COMPASS, the CDC assembler level code. The central memory requirement is 60000_8 for compiling and executing the source version presented here and 55000_8 for loading and executing the absolute binary version.

Six files are used by the program: TAPE1, TAPE5=INPUT, TAPE6=OUTPUT, TAPE7, TAPE8, and TAPE9. TAPE1 is a binary file containing the input time series written in one of the three formats described in appendix D. TAPE5 is a binary-coded decimal (BCD) file containing the card input data in NAMELIST and FORTRAN READ formats. It is equivalenced to the input file. TAPE6 is a BCD file containing the output to be printed. It is equivalenced to OUTPUT and is automatically printed. TAPE7 is a binary file containing output values of all spectra, correlations, coherence, and transfer functions computed during execution of the program. TAPE8 and TAPE9 are random-access disk storage files used for temporary storage. TAPE1 will be a magnetic-tape file. TAPE7 may be a magnetic-tape file or a disk file copied to a tape after execution.

Program Specifications

The program is written as an overlay structure with two levels. The main overlay sets up the COMMON storage arrays and calls the primary overlays for multiple-case execution. The first primary overlay reads the card input, checks for errors, prints informational messages, and computes the accuracy measurement of the spectral estimators. The second primary overlay reads the input time series by blocks, computes the transform of each block, and stores the results on random-access disk storage. The third primary overlay calculates the averaged auto power spectra and autocorrelations and calls the plot subroutines. The fourth primary overlay calculates the average cross power spectra, cross correlations, transfer functions, and coherence and calls the plot subroutine. One large array in blank COMMON provides the temporary storage blocks for all overlays. Initial block addresses are assigned in the main overlay for reference by the primary overlay programs. Labeled COMMON blocks hold the input and control parameters used by all the levels.

Appendix E contains a general flow diagram of PATS. Appendix F contains a list of the programs and subprograms used in PATS, with a brief description of the purpose of each. Appendix G contains the Langley Library subroutines used by PATS. Appendix H contains the source listing of PATS.

OPERATING INSTRUCTIONS

Deck Setups for Langley Operating System

The following examples show deck setups for the Langley Operating System.

Deck setup 1.- Purpose is to fetch and execute the absolute binary version. Field length required is 55000_8.

```
JOB card
USER card
FETCH(A4119, ,BINARY, ,PATS)
NOMAP.
LINECNT,10000.
[1]REQUEST,TAPE1,HY.        tape no.,ROL,
REWIND(TAPE1)
SETINDF.
PATS.
```

[1]See Langley Computer Programing Manual for format.

RFL,10000.

DROPFIL(TAPE1)

REWIND(TAPE7)

[1]REQUEST,TAPE99,HY. SAVTP,RIS,your 3 initials, identification

COPYBF(TAPE7,TAPE99)

DROPFIL(TAPE99)

EXIT.

RFL,10000.

DROPFIL(TAPE1)
REWIND(TAPE7)

[1]REQUEST,TAPE99,HY. SAVTP,RIS,your 3 initials, identification

COPYBF(TAPE7,TAPE99)

DROPFIL(TAPE99)

[2]7/8/9

(Data card deck inserted here)

[3]6/7/8/9

Deck setup 2.- Purpose is to fetch, compile, and execute the source version. Field length required is 60000_8.

JOB card

USER card

FETCH(A4119, ,SOURCE)

RUN(S, , ,SCFILE)

LINECNT,10000.

[1]REQUEST,TAPE1,HY. tape no.,ROL,

REWIND(TAPE1)

SETINDF.

I.GO.

---

[1]See Langley Computer Programing Manual for format.

[2]End-of-record card.

[3]End-of-file card.

RFL,10000.

DROPFIL(TAPE1)

REWIND(TAPE7)

[1]REQUEST,TAPE99,HY. SAVTP,RIS,your 3 initials, identification

COPYBF(TAPE7,TAPE99)

DROPFIL(TAPE99)

EXIT.

RFL,10000.

DROPFIL(TAPE1)

REWIND(TAPE7)

[1]REQUEST,TAPE99,HY. SAVTP,RIS,your 3 initials, identification

COPYBF(TAPE7,TAPE99)

DROPFIL(TAPE99)

[2]7/8/9

Mod card deck if any (may be a blank record)

[2]7/8/9

(Data card deck inserted here)

[3]6/7/8/9

Card Input Data Description

The card input parameters are entered via FORTRAN NAMELIST and READ statements with formats for the READ statements as specified. Some parameters have default values noted below. Parameter types are defined as I, integer; R, real; A, alphanumeric.

---

[1]See Langley Computer Programing Manual for format.

[2]End-of-record card.

[3]End-of-file card.

NAMELIST input format:

FORTRAN name	Default value	Parameter type	Description
$INPUT			
ITFMT	2	I	Code for input tape format: 1 tape format 1 2 tape format 2 3 tape format 3 (see appendix D for format descriptions)
NFSKIP	0	I	Number of logical binary files on input tape to be skipped before starting execution of this case
NRSKIP	0	I	Number of logical binary records on input tape to be skipped before starting execution of this case
SN		R	Serial number of input data
DELTAT		R	1/Sampling rate of input data
STARTT	0.0	R	Starting time in seconds at which program is to start processing data from input tape
OFFSCAL	10^6	R	Offscale value for all channels
NCH		I	Number of data channels on input tape (maximum value = 14)
SCALFAC	1.	R	Array of NCH values of scale factors, one for each channel of input data; every point for channel i is multiplied by SCALFAC(i).
NPTOT		I	Total number of data points to be read for each channel
NREAD		I	Number of data points per block to be read for each channel (maximum value is 1024 for INZERO=0, 512 for INZERO=1)
IAUTOSP	0	I	Array of NCH codes for computing auto spectra: 1 compute auto spectrum for channel i 0 do not compute auto spectrum for channel i

FORTRAN name	Default value	Parameter type	Description
IAUTOCO	0	I	Array of NCH codes for computing autocorrelation: 1 compute autocorrelation for channel i 0 do not compute autocorrelation for channel i
IFILTER	0	I	Array of NCH codes for spectral filtering: 1 filter auto and cross spectra for channel i 0 do not filter spectra for channel i (autocorrelation and cross correlation will be computed from filtered spectra)
NCROSS	0	I	Number of pairs of channels to perform cross functions on (maximum value = 20)
ICROSS	0	I	Array of channel numbers for cross functions: ICROSS(1,i) first channel no. for pair i ICROSS(2,i) second channel no. for pair i
ICRSP	0	I	Array of NCROSS codes for computing cross spectra for each pair of channels: 1 compute cross spectrum for pair i 0 do not compute cross spectrum for pair i
ICRCOR	0	I	Array of NCROSS codes for computing cross correlations: 1 compute cross correlation for pair i 0 do not compute cross correlation for pair i
ITRA	0	I	Array of NCROSS codes for computing transfer functions: 1 compute transfer function, $\tilde{H}(\omega) = \tilde{S}_{xy}(\omega)/\tilde{S}_x(\omega)$ -1 compute transfer function, $\tilde{H}(\omega) = \tilde{S}_{xy}(\omega)/\tilde{S}_y(\omega)$ 0 do not compute transfer function for pair i

FORTRAN name	Default value	Parameter type	Description
ICOH	0	I	Array of NCROSS codes for computing coherence: 1 compute coherence function for pair i 0 do not compute coherence for pair i
LAP	0	I	Code for overlapping blocks of input data: 1 overlap data blocks 50 percent 0 no overlap
IWINDOW	1	I	Code for type of data window: 0 boxcar window 1 Hann window 2 Hamming window 3 Parzen window
ITYPESP	2	I	Code for type of spectral output: 1 power spectrum 2 power spectral density 3 amplitude spectrum
NPRINT	100	I	Number of points to be printed from auto or cross spectra
IPLOTA	1	I	Code for auto spectral fanfold plots and/or binary tape output: 1 no output 2 plot and save 1/3-octave spectra only (log scale) 3 plot and save narrow-band spectra only (linear scale) 4 plot and save narrow-band spectra only (log scale) 5 both options 2 and 3 6 both options 2 and 4
IPLOTC	0	I	Code for cross spectral fanfold plots and/or binary tape output: 0 no output 1 plot and save real and imaginary (linear scale) against frequency (linear scale)

FORTRAN name	Default value	Parameter type	Description
IPLOTC	0	I	3 plot and save magnitude and phase (linear) against frequency (linear) 5 plot magnitude (log scale) and phase against frequency (linear) and save magnitude and phase values
F1	0.0	R	Lower limit of frequency to be plotted on narrow-band spectra plots
F2	20000.	R	Upper limit of frequency to be plotted on narrow-band spectra plots (If F1 and F2 are both zero, no narrow-band plots will be made)
LAG1	0	I	Lower limit of the number of time lags to be plotted on correlation plots
LAG2	0	I	Upper limit of the number of time lags to be plotted on correlation plots (If LAG1 and LAG2 are both zero, no correlation plots will be made)
PCTC	90.	R	Percent band to be used for calculation of confidence band and level of significance in chi-square calculation
NBINS	0	I	Number of bins to be used in histograms: 0 no histograms (maximum value=100)
DMAX	0.	R	Array of values of maximum readings for each channel
DMIN	0.	R	Array of values of minimum readings for each channel
INZERO	0	I	Code for zero insertion option: 1 insert NREAD zeros at end of input data block (block size is 2 × NREAD) (this option should be used for runs requesting cross correlations) 0 no zero insertion

FORTRAN name	Default value	Parameter type	Description
NFILTP	0	I	Number of points in input spectral filter: 0 no spectral filter (maximum value=50)
FREQF	0.0	R	Array of values of frequency for spectral filter
WGHTF	0.0	R	Array of values of weights for spectral filter (user should be careful to specify points close together where derivative of filter function changes)

$

Input cards after NAMELIST input:

Card no.	FORTRAN name	Format	Parameter type	Description
1	YLABEL	2A10	A	Array of two words (20 characters) to be written on each plot frame for case identification
2	TRACK	8A10	A	Array of NCH identification words, one unique word for each channel on input tape, eight words per card; more than one card may be needed

Output Description

The contents of the printed output and binary tape output of PATS are described in this section.

Printed output.- The printed output consists of the following items:

(1) Echo of input data

(2) Informational messages about block size, type of Fourier transform to be used, error messages about input data

(3) Accuracy measurement of the spectral estimators

(4) Table of values of spectral filter calculated from input table by SPLINE

(5) Table of number of offscale values read for each channel

(6) For each channel processed,

a. Channel number, channel ID, mean and square root of variance of input data (MEAN and SIGMA) and the root mean square (RMS) as computed from the power spectral density

b. If auto spectral output is desired, a list of NPRINT values of frequency and power of averaged narrow-band spectrum, and 1/3-octave band power spectrum and power spectral density

c. If autocorrelation is desired, a list of time and R_x, with time lags from 0 to N/2 (N=block size)

(7) If histograms are requested, a fanfold plot of bin number against counts, a list of values of occurrences, and a goodness-of-fit test calculation

(8) For each pair of channels processed,

a. If cross spectral output is desired, a list of NPRINT values of frequency, real and imaginary parts, and magnitude and phase of complex narrow-band power spectrum

b. If cross correlation is desired, a list of time lag and $R_{xy}(\tau)$, with time-lag values from -N/2 to N/2 (N=block size)

c. If coherence is desired, a list of NPRINT values of frequency and coherence

d. If transfer function is desired, a list of NPRINT values of frequency and transfer function

(9) Fanfold output – plots of every function computed for which plots are specified will appear in the printed output immediately following the listed values; the plots are limited to 256 points each to conserve line count; the first 256 points between F1 and F2 of each spectrum are plotted; points between LAG1 and LAG2 of correlations are plotted, skipping intermediate points to reduce the number of plotted points to less than 256

Binary tape output.- Every function computed is written onto file TAPE7 when it is computed. All calculated values are written. One record is created on the file in the following format:

Word	Type	Description
1 to 6	A	Label describing function and channels
7	I	Number of points in output function, NP

Word	Type	Description
8	R	First value of independent variable
9	R	First value of dependent variable
10	R	Second value of independent variable
11	R	Second value of dependent variable
.		
.		
.		
NP + 7	R	NPth value of independent variable
NP + 8	R	NPth value of dependent variable

In printed output, a message is written noting the record number and descriptive label for the function.

Restrictions and Limitations

The restrictions and limitations for use of PATS are as follows:

(1) The binary input tape must be positioned at the beginning of the data to be processed before the program starts reading data for the case. This may be accomplished by using control cards before execution and by assigning the correct nonzero values to NFSKIP and NRSKIP for each input case. For tape format 1, it should be positioned at a record with the desired serial number in the second word (may be after the first record). For tape format 2, it should be positioned at the ID record with the desired serial number in the eighth word. For tape format 3, it should be at the record of the file containing the desired serial number. If this condition is not met, a message will be printed and execution stopped. When both NFSKIP and NRSKIP have nonzero values, NFSKIP files are skipped first. For tape format 2 the ID record is checked, the next two records are read, then NRSKIP records are skipped. For tape format 1, no records are read before skipping NRSKIP records.

(2) The version of the program being presented has a maximum block size of 1024. The program storage requirements are 55000_8 for the absolute binary version and 60000_8 for the source version. To change this limit, NMAX must be assigned the desired value in the program MAIN and the dimension of CMAIN changed accordingly.

(3) The number of block averages and the number of individual channels to be processed are restricted by NBCMAX. The product must be less than or equal to 800. To change this limit, assign the desired value to NBCMAX in MAIN and change the dimension of KNDEX accordingly.

(4) The number of data channels on the input tape is limited to 14. To change this limit, change the dimensions of all variables dimensioned 14 in COMMON blocks BLK2, BLK5, and BLK8 in all overlays; assign the correct value to NCHMAX in MAIN; and make CMAIN dimension the larger of 4NMAX+6 or 2NMAX+6+64NCHMAX+512.

(5) When the amplitude spectrum option is selected, no other functions will be calculated.

(6) Filter input function is restricted to 50 points. To change this limit, change the dimensions of FREQF and WGHTF in COMMON block BLK9 in all overlays and change value in test in READIN (two statements after statement number 113).

Error Messages and Remedies

The error messages and suggested remedies are as follows:

(1) NCH GREATER THAN NCHMAX, PROGRAM WILL NOT READ TAPE CORRECTLY. JOB TERMINATED.
To correct, see item 4 in "Restrictions and Limitations."

(2) YOU MAY HAVE CIRCULAR ERROR IN YOUR CORRELATIONS BECAUSE YOU HAVE NOT ASKED FOR ZERO INSERTION.
Job will continue. To correct, change INZERO to 1 and rerun.

(3) BLOCK SIZE TOO LARGE FOR DIMENSIONS PROVIDED.
Job terminated. To correct, see item 2 in "Restrictions and Limitations."

(4) NO 50 PERCENT OVERLAP ON ZERO INSERTION RUNS.
Input value of LAP will be altered to zero and job will continue.

(5) INPUT INDICATES NO CHANNELS TO BE PROCESSED.
Job terminated. Check input data and rerun. PATS resets all computing options other than AUTOSP to zero when ITYPESP=3. See item 5 in "Restrictions and Limitations."

(6) NCHP*NBLK GREATER THAN NBCMAX.
Execution ended. To correct, see item 3 in "Restrictions and Limitations."

(7) INPUT ERROR, NFILTP GT 50.
Execution ended. To correct, see item 6 in "Restrictions and Limitations."

(8) NCH GT 100 NOT ALLOWED.
Execution ended. No correction of program possible.

(9) TAPE NOT POSITIONED AT ID RECORD FOR DESIRED SN.
Execution ended. Correct input deck to position tape correctly and rerun.

(10) TAPE NOT POSITIONED AT DESIRED SN.
Execution ended. Correct input deck to position tape correctly and rerun.

(11) STARTING TIME NOT FOUND ON FILE.
Execution ended. Check tape for correct times.

(12) END OF FILE ENCOUNTERED BEFORE NPTOT POINTS READ.
Check tape for number of points after STARTT; correct input data and rerun.

CONCLUDING REMARKS

This paper has presented a general purpose digital computer program for the harmonic analysis of multiple channels of time-history data. The program is written primarily in CDC FORTRAN and employs the technique of the fast Fourier transform. A complete program listing with descriptions of necessary subroutines is included so that the program may be adapted to any facility. In addition, the philosophy and theory employed by the program are discussed so that the user may make appropriate choices among the options available.

Langley Research Center,
National Aeronautics and Space Administration,
Hampton, Va., April 8, 1974.

APPENDIX A

FINITE FOURIER TRANSFORM OF A PERIODIC SIGNAL

Suppose that $f(t)$ is a signal with period p. Then, it can be represented by the Fourier series

$$f(t) = \frac{A_0}{2} + \sum_{n=1}^{\infty} (A_n \cos \omega_n t + B_n \sin \omega_n t) \tag{A1}$$

where $\omega_n = \frac{2\pi n}{p}$ are harmonics of the fundamental radian frequency $\omega = \frac{2\pi}{p}$ of the signal. Further, suppose that N samples of this signal at equal intervals Δt are available for a total record length of $T = N \Delta t$.

The finite Fourier transform of this signal is given by

$$X_k = \sum_{j=0}^{N-1} f(j \Delta t) e^{-i2\pi jk/N} \qquad (k = 0, 1, 2, \ldots, N/2)$$

at the frequencies $\omega_k = \frac{2\pi k}{T}$. These frequencies will correspond to the harmonic frequencies of the Fourier series if and only if $T = \nu p$, where ν is a positive integer. In this case, the mth harmonic will be equal to the kth frequency at which the finite transform is evaluated when $k = \nu m$.

Now, when $t = \nu p$, it can be shown that the finite Fourier transform of equation (A1) is given by

$$X_k = \frac{A_0}{2} N \delta(k) + N \sum_{n=1}^{\infty} \frac{A_n - jB_n}{2} \delta(k - \nu n) + N \sum_{n=1}^{\infty} \frac{A_n}{2} \sum_{l=1}^{\infty} \left[\delta(k - \nu n + lN) + \delta(k + \nu n - lN)\right]$$

$$- iN \sum_{n=1}^{\infty} \frac{B_n}{2} \sum_{l=1}^{\infty} \left[\delta(k - \nu n + lN) - \delta(k + \nu n - lN)\right]$$

where $\delta(j)$ is the Dirac delta function.

The most interesting of these transforms are those which correspond to harmonics of the fundamental period, that is, $k = \nu m$. For this case,

$$\frac{X_{\nu m}}{N} = \frac{A_m - iB_m}{2} + \frac{1}{2}\sum_{l=1}^{\infty}\left(A_{\frac{lN}{\nu}+m} + A_{\frac{lN}{\nu}-m}\right) - \frac{i}{2}\sum_{l=1}^{\infty}\left(B_{\frac{lN}{\nu}+m} - B_{\frac{lN}{\nu}-m}\right) \tag{A2}$$

The summation terms in equation (A2) involve aliasing of power from higher frequencies. Note that the aliasing depends upon N/ν, the number of points per fundamental period of the signal. Since the Nyquist frequency occurs when $\nu = N/2$, the aliasing may be removed by filtering the signal above $f_\nu = 1/2\ \Delta t = N/2\nu p$. Assume that aliasing has been removed, then

$$\frac{|X_{\nu m}|}{N} = \frac{\sqrt{A_m^2 + B_m^2}}{2}$$

and

$$\tan \phi_m = \frac{B_m}{A_m} = -\frac{I_m(X_{\nu m})}{Re(X_{\nu m})} \tag{A3}$$

As an example, consider the square wave of amplitude M and period p. If the Fourier series representation of this signal is considered, it can be shown that

$$A_n = \frac{2}{p}\int_0^p f(t) \cos \omega_n t\, dt = 0$$

and

$$B_n = \frac{2}{p}\int_0^p f(t) \sin \omega_n t\, dt = \begin{cases} 4M/n\pi & \text{(n odd)} \\ 0 & \text{(n even)} \end{cases}$$

Thus, the square wave admits the Fourier series representation

$$f(t) = \frac{4M}{\pi}\sum_{n=1}^{\infty}\frac{1}{2n-1}\sin\frac{2\pi(2n-1)t}{p} \tag{A4}$$

The magnitude and phase of this representation are given by

$$\frac{\sqrt{A_n^2 + B_n^2}}{2} = \begin{cases} 0 & \text{(n even)} \\ \frac{2M}{n\pi} & \text{(n odd)} \end{cases}$$

and

$$\phi_n = \tan^{-1}\frac{B_n}{A_n} = \tan^{-1}(\infty) = \frac{\pi}{2}$$

To obtain the finite Fourier series representation for this function, assume, without loss of generality, that $\nu = 1$ and N is an even number. Then, since equation (A4) yields $f(0) = f(p/2) = 0$, the finite Fourier transform becomes

$$X_k = \sum_{j=0}^{N-1} f(j\,\Delta t)e^{-i2\pi jk/N}$$

$$= M\sum_{j=1}^{\frac{N}{2}-1}\left(e^{-i2\pi k/N}\right)^j - M\sum_{j=\frac{N}{2}+1}^{N-1}\left(e^{-i2\pi k/N}\right)^j$$

$$= M\left(\frac{1 - e^{-i\pi k}}{1 - e^{-i2\pi k/N}} - 1\right) - M\left(\frac{1 - e^{-i2\pi k}}{1 - e^{-i2\pi k/N}} - \frac{1 - e^{\frac{-i2\pi k}{N}\left(\frac{N}{2}+1\right)}}{1 - e^{-i2\pi k/N}}\right)$$

$$= M\frac{\left(1 - e^{i\pi k}\right)\left(e^{-i2\pi k/N} - e^{-i\pi k}\right)}{1 - e^{-i2\pi k/N}}$$

$$= \begin{cases} 0 & \text{(k even)} \\ -2Mi\cot\frac{\pi k}{N} & \text{(k odd)} \end{cases}$$

Thus,

$$\frac{X_k}{N} = \begin{cases} 0 & \text{(k even)} \\ \frac{-i2M}{N} \cot \frac{\pi k}{N} & \text{(k odd)} \end{cases}$$

From equations (A2) and (A3), it can be seen that when k is even,

$$A_n = 0 \qquad \text{and} \qquad \phi_n = \frac{\pi}{2}$$

Further, when k is odd,

$$-\mathrm{Im}\left(\frac{X_k}{N}\right) = \frac{2M}{N} \cot \frac{\pi k}{N} = \frac{2M}{N}\left[\frac{N}{\pi k} - \sum_{j=1}^{\infty} \frac{2^{2j}|B_{2j}|}{(2j)!}\left(\frac{\pi k}{N}\right)^{2j-1}\right] = \frac{B_k}{2} - \frac{2M}{N} \sum_{j=1}^{\infty} \frac{2^{2j}|B_{2j}|}{(2j)!}\left(\frac{\pi k}{N}\right)^{2j-1} \qquad \text{(A5)}$$

where B_{2j} is a Bernoulli number. The summation which appears in this equation is the aliased term which arises because the signal was not low pass filtered.

APPENDIX B

SPECTRAL ESTIMATION THROUGH USE OF THE FINITE FOURIER TRANSFORM

Let $x(t)$ be an arbitrary stationary random process and define

$$X_T(\omega) = \frac{1}{2\pi}\int_{-T/2}^{T/2} x(t)e^{-i\omega t}\,dt = \frac{1}{2\pi}\int_{-\infty}^{\infty} u_d(t)\,x(t)e^{-i\omega t}\,dt \tag{B1}$$

where $u_d(t)$ is any data window which is zero for $|t| > T/2$. Then, the power spectral estimate becomes

$$\hat{S}_x(\omega) = \frac{\pi}{T}\left|X_T(\omega)\right|^2 = \frac{\pi}{T}\frac{1}{4\pi^2}\int_{-\infty}^{\infty} dt_1 \int_{-\infty}^{\infty} u_d(t_1)\,u_d(t_2)\,x(t_1)\,x(t_2)e^{-i\omega(t_1-t_2)}\,dt_2 \tag{B2}$$

Taking the ensemble expectation of this quantity yields

$$E\left[\hat{S}_x(\omega)\right] = \frac{\pi}{T}\frac{1}{4\pi^2}\int_{-\infty}^{\infty} dt_1 \int_{-\infty}^{\infty} u_d(t_1)\,u_d(t_2)\,R_x(t_1 - t_2)e^{-i\omega(t_1-t_2)}\,dt_2 \tag{B3}$$

where

$$R_x(\tau) = \int_{-\infty}^{\infty} \hat{S}_x(\omega')e^{i\omega'\tau}\,d\omega' \tag{B4}$$

is the autocorrelation of the random process $x(t)$. Employing this relation in equation (B3) gives

$$E\left[\hat{S}_x(\omega)\right] = \frac{\pi}{T}\frac{1}{4\pi^2}\int_{-\infty}^{\infty} dt_1 \int_{-\infty}^{\infty} u_d(t_1)\,u_d(t_2)\,dt_2 \int_{-\infty}^{\infty} \hat{S}_x(\omega')e^{i\omega'\tau}e^{-i\omega'\tau}\,d\omega'$$

$$= \frac{\pi}{T}\int_{-\infty}^{\infty} \hat{S}_x(\omega')\,d\omega' \frac{1}{2\pi}\int_{-\infty}^{\infty} u_d(t_1)e^{i(\omega'-\omega)t_1}\,dt_1 \frac{1}{2\pi}\int_{-\infty}^{\infty} u_d(t_2)e^{-i(\omega'-\omega)t_2}\,dt_2 \tag{B5}$$

Now, define

$$U_d(\omega) = \frac{1}{2\pi}\int_{-\infty}^{\infty} u_d(t)e^{-i\omega t}\,dt \tag{B6}$$

where $U_d(\omega)$ will be real and even if $u_d(t)$ is even. Then equation (B5) becomes

$$E\left[\hat{S}_x(\omega)\right] = \frac{\pi}{T}\int_{-\infty}^{\infty} \hat{S}_x(\omega')\,U_d(\omega - \omega')\,U_d(\omega' - \omega)\,d\omega' = \frac{\pi}{T}\int_{-\infty}^{\infty} \hat{S}_x(\omega')\,U_d^2(\omega' - \omega)\,d\omega' \tag{B7}$$

Thus, the spectral estimate obtained in this way is a smoothed approximation to the actual spectrum as seen through the spectral window characterized by squaring the Fourier transform of the data window.

In order for this estimate to be power preserving, it is necessary for the integral of the mean estimate to be equal to the total power. Integrating equation (B3) yields

$$\int_{-\infty}^{\infty} E\left[\hat{S}_x(\omega)\right] d\omega = \frac{\pi}{T}\frac{1}{4\pi^2}\int_{-\infty}^{\infty} dt_1 \int_{-\infty}^{\infty} u_d(t_1)\,u_d(t_2)\,R_x(t_1 - t_2)\,dt_2 \int_{-\infty}^{\infty} e^{i\omega(t_1 - t_2)}\,d\omega \tag{B8}$$

and since

$$\delta(t_1 - t_2) = \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{i\omega(t_1 - t_2)}\,d\omega \tag{B9}$$

equation (B8) becomes

$$\begin{aligned}\int_{-\infty}^{\infty} E\left[\hat{S}_x(\omega)\right] d\omega &= \frac{\pi}{T}\frac{1}{2\pi}\int_{-\infty}^{\infty} dt_1 \int_{-\infty}^{\infty} u_d(t_1)\,u_d(t_2)\,R_x(t_1 - t_2)\,\delta(t_1 - t_2)\,dt_2 \\ &= \frac{\pi}{T}\frac{1}{2\pi}\int_{-\infty}^{\infty} u_d^2(t)\,R_x(0)\,dt_1 \\ &= \frac{R_x(0)}{2T}\int_{-\infty}^{\infty} u_d^2(t_1)\,dt_1 \end{aligned} \tag{B10}$$

APPENDIX B

Since the total power in the signal is given by $R_x(0)$, define a new estimate $\tilde{S}_x(\omega)$ by the following equation:

$$\int_{-\infty}^{\infty} E\left[\tilde{S}_x(\omega)\right] d\omega = R_x(0) \tag{B11}$$

Clearly, this estimate is related to the old estimate $\hat{S}_x(\omega)$ by

$$\tilde{S}_x(\omega) = \frac{2T}{W_u} \hat{S}_x(\omega) \tag{B12}$$

where

$$W_u = \int_{-\infty}^{\infty} u_d^2(t_1)\, dt_1 \tag{B13}$$

is the window correction factor. Thus, since the estimate $\hat{S}_x(\omega)$ is given by equation (9) as

$$\hat{S}_x(\omega_k) = \frac{\Delta t}{4\pi N} |z_k|^2$$

the desired spectral estimate $\tilde{S}_x(\omega)$ is obtained:

$$\tilde{S}_x(\omega_k) = \frac{(\Delta t)^2}{2\pi W_u} |z_k|^2 \tag{B14}$$

The window correction factors for the various data windows are as follows:

For the boxcar window,

$$W_u = T \tag{B15}$$

For the Hann window,

$$W_u = \frac{3T}{8} \tag{B16}$$

For the Hamming window,

$$W_u = T\left(0.3974 + \frac{0.9936}{\pi}\right) \tag{B17}$$

And for the Parzen window,

$$W_u = T\,\frac{151}{560} \tag{B18}$$

APPENDIX C

AUTOCORRELATION ESTIMATION FROM ESTIMATED POWER SPECTRAL DENSITY

The autocorrelation of a function of time $x(t)$ would normally be estimated as the inverse Fourier transform of the estimated power spectrum; that is,

$$\hat{R}_x(\tau) = \int_{-\infty}^{\infty} \tilde{S}_x(\omega) e^{i\omega\tau}\, d\omega \tag{C1}$$

Thus,

$$E\left[\hat{R}_x(\tau)\right] = \int_{-\infty}^{\infty} E\left[\tilde{S}_x(\omega)\right] e^{i\omega\tau}\, d\omega \tag{C2}$$

Now, it can be shown from the equations of appendix B that

$$E\left[\tilde{S}_x(\omega)\right] = \frac{2\pi}{W_u} \int_{-\infty}^{\infty} S_x(\omega')\, U_d{}^2(\omega' - \omega)\, d\omega'$$

Thus, equation (C2) becomes

$$E\left[\hat{R}_x(\tau)\right] = \frac{2\pi}{W_u} \int_{-\infty}^{\infty} d\omega' e^{i\omega'\tau}\, S_x(\omega') \int_{-\infty}^{\infty} d\omega\, U_d{}^2(\omega' - \omega) e^{-i(\omega' - \bar{\omega})\tau} \tag{C3}$$

Now, by setting $\omega_0 = \omega' - \omega$, equation (C3) yields

$$E\left[\hat{R}_x(\tau)\right] = \frac{2\pi}{W_u} \int_{-\infty}^{\infty} d\omega' e^{i\omega'\tau}\, S_x(\omega') \int_{-\infty}^{\infty} d\omega_0\, U_d{}^2(\omega_0) e^{-i\omega_0\tau}$$

$$= \frac{2\pi}{W_u} R_x(\tau) \int_{-\infty}^{\infty} d\omega_0\, U_d{}^2(\omega_0) e^{-i\omega_0\tau} \tag{C4}$$

Recall the definition (from eq. (B6))

$$U_d(\omega) = \frac{1}{2\pi} \int_{-\infty}^{\infty} u_d(t) e^{-i\omega t}\, dt$$

Further, since $u_d(t)$ is real and even for all windows considered in this report, $U_d(\omega)$ is real also. Thus,

$$U_d^2(\omega) = U_d(\omega)\, U_d^*(\omega)$$

$$= \frac{1}{4\pi^2} \int_{-\infty}^{\infty} dt \int_{-\infty}^{\infty} dt'\, u_d(t)\, u_d(t') e^{-i\omega(t-t')}$$

and

$$\int_{-\infty}^{\infty} d\omega_0\, U_d^2(\omega_0) e^{-i\omega_0 \tau} = \frac{1}{2\pi} \int_{-\infty}^{\infty} dt \int_{-\infty}^{\infty} dt'\, u_d(t)\, u_d(t') \frac{1}{2\pi} \int_{-\infty}^{\infty} d\omega_0 e^{-i\omega_0 (t-t'+\tau)} \tag{C5}$$

However,

$$\frac{1}{2\pi} \int_{-\infty}^{\infty} d\omega e^{-i\omega(t-t'+\tau)} = \delta(t - t' + \tau)$$

Thus, equation (C5) becomes

$$\int_{-\infty}^{\infty} d\omega_0\, U_d^2(\omega_0) e^{-i\omega_0 \tau} = \frac{1}{2\pi} \int_{-\infty}^{\infty} dt\, u_d(t)\, u_d(t + \tau)$$

and

$$E\left[\hat{R}_x(\tau)\right] = \frac{\int_{-\infty}^{\infty} u_d(t)\, u_d(t + \tau)\, dt}{\int_{-\infty}^{\infty} u_d^2(t)\, dt} R_x(\tau) \tag{C6}$$

Therefore, in order to have an unbiased estimate of the autocorrelation, it is necessary to define the new estimate (eq. (25)):

$$\tilde{R}_x(\tau) = W_R \int_{-\infty}^{\infty} \tilde{S}_x(\omega) e^{i\omega\tau}\, d\omega$$

where

$$W_R = \frac{\int_{-\infty}^{\infty} u_d^2(t)\, dt}{\int_{-\infty}^{\infty} u_d(t)\, u_d(t+\tau)\, dt}$$

is again a window correction factor.

Note that for the boxcar window,

$$u_{T/2}(t) = \begin{cases} 1 & (|t| \leq T/2) \\ 0 & (\text{otherwise}) \end{cases}$$

$$\int_{-\infty}^{\infty} u_{T/2}^2(t)\, dt = T$$

and

$$\int_{-\infty}^{\infty} u_{T/2}(t)\, u_{T/2}(t+\tau)\, dt = \begin{cases} T\left(1 - \frac{|\tau|}{T}\right) & (|\tau| < T) \\ 0 & (\text{otherwise}) \end{cases}$$

Therefore,

$$W_R = \begin{cases} \left(1 - \frac{|\tau|}{T}\right)^{-1} & (|\tau| < T) \\ 0 & (\text{otherwise}) \end{cases}$$

APPENDIX D

BINARY INPUT TAPE FORMATS

Tape Format 1

Data digitized by analog-to-digital conversion equipment at Langley is edited, reduced to engineering units, and put on a digital computer tape. The computer program which does this is called the Adtran Quantity Pass and its standard output data tape is called Adtran Output Tape. The CDC Adtran Output Tape is explicitly blocked and the actual end-of-file mark is used to indicate the end of writing on tape.

All tapes will be written in the binary parity using the standard CDC FORTRAN 2.0 input/output statements. There will be between 11 and 110 FORTRAN 2.0 logical words per frame. These frames will be blocked into larger physical records. A file of data will be completely defined by serial number. New serial numbers will always begin in a new physical record. If a physical record is not complete, it will be filled with 999999 (six 9's). The end of writing on the tape will be indicated by an end-of-file mark. The frame format is as follows:

Word	Type	Contents and Description
1	Floating	The number of channels of data in this frame; less than 40 for continuous data and less than 100 for commutated data
2		Serial number; the input card format for serial number should be 6 digits wide
3 4		Words 3 and 4 are the primary engineering identification, for example, test and run; they would be represented on input card formats by no more than 6 digits apiece
5 6		Words 5 and 6 are additional engineering identification
7 8		Words 7 and 8 are Greenwich Mean Time and are used only for telemetry data; for ground facilities, word 8 may be ground facilities
9		Frame count which starts at 1 for each new serial number

APPENDIX D

Word	Type	Contents and Description
10	Floating	Elapsed time in seconds; processing will be controlled by elapsed time within a file; the increments in elapsed time may not be constant
11		Data channel 1
12		Data channel 2
13		Data channel 3
.		.
.		.
.		.
N + 10		Data channel N, where N is the number given in logical word 1

The relationship between frames and records is shown below.

Number of channels, N	Words per frame	Frames per record	Words per record
$1 \leqq N \leqq 10$	20	25	500
$10 < N \leqq 20$	30	17	510
$20 < N \leqq 30$	40	12	480
$30 < N \leqq 40$	50	10	500
*$30 < N \leqq 100$	110	4	440

*Commutated.

As an example, to read a 12-channel frame, a physical record of 510 words is read. The time of the first frame is in word 10, the time of the second frame is in word 40, . . ., the time of the 17th frame is in word 490.

Tape Format 2

The tape is a FORTRAN written, binary-parity, multifile tape with a flexible yet efficient format. Each file contains four basic record types (ID, NAMES, UNITS, and DATA) and consists of a continuous unique test (or run). The ID record contains non-repetitive information such as run or test number, date, time bias, and record blocking factors. The NAMES and UNITS records contain data channel names and engineering units, respectively. The DATA records themselves contain the engineering data. In addition, each record begins with a KEY word denoting the record type followed by a word containing the record size. Thus, all information necessary to operate on any file is

available within the first four records of the file. The formats for the records in each file are as follows:

Record 1 ID Record

Word	Name	Format	Description
1	KEY	A	ID
2	NN	I	Number of remaining words in the record = 19
3	IWD	I	Number of words of unblocked data in a data record
4	KCH	I	Number of words of blocked data in a data record
5	NFR	I	Number of frames in a data record (blocking factor)
6	ID(1)	A	Name for first ID parameter = SERIAL
7	ID(2)	A	UNITS for first ID parameter = NUMBER
8	ID(3)	F	First ID parameter = the serial number
9	ID(4)	A	NAME(2) second ID parameter = TEST
10	ID(5)	A	UNITS(2) second ID parameter = NUMBER
11	ID(6)	F	Second parameter = the test number
12	ID(7)	A	NAME(3) = DATE
13	ID(8)	A	UNITS(3) = DAYS, YR-MONTH-DAY or UNKNOWN
14	ID(9)	F	PARAMETER(3) = YEAR × 10000 + MONTH × 100 + DAY
15	ID(10)	A	NAME(4) = BIAS
16	ID(11)	A	UNITS(4) = SECONDS
17	ID(12)	F	PARAMETER(4) = GMT time bias
18	ID(13)	A	NAME(5) = ENGR ID
19	ID(14)	*I	UNITS(5) = 2
20-21	ID(15-16)	A	PARAMETER(5) = Engineering identification (two words)

*When the UNITS word for a parameter contains an integer less than 12, the parameter is defined to be alphanumeric data of that many words in length.

APPENDIX D

Record 2 NAMES Record

Word	Name	Format	Description
1	KEY	A	NAMES
2	NN	I	Number of unblocked parameters plus number of blocked parameters
3 NN + 2	NAMES	A	Names for unblocked data parameters followed by names for blocked data

Each parameter including time will have a name.

Record 3 UNITS Record

Word	Name	Format	Description
1	KEY	A	UNITS
2	NN	I	Number of unblocked parameters plus number of blocked parameters
3 NN + 2	UNITS	A	Units for unblocked data followed by UNITS for blocked data

The UNITS are not always necessary and will sometimes be blank.

Record 4 through EOF, DATA Records

Word	Name	Format	Description
1	KEY	A	Data
2	NN	I	Number of remaining words on the record (IWD + KCH * NFR)
3	XDATA(1)	F	First word of IWD words of UNBLOCKED data (FRAME COUNT, e.g.)
4 + IWD ZDATA	ZDATA(I,J)	F	Blocked data I parameter, J frames

The data records are optimally packed to approach, but not exceed, 512 words per record. The record size is determined as follows:

SIZE = NFR * KCH + IWD + 2

where

NFR = (510 - IWD)/KCH

NFR is the blocking factor (integer)
IWD is the number of nonrepeated words in the record
KCH is the number of data channels

For example, a test with 9 recorded channels and only one word of unblocked data per record would have 507 words in each data record as follows:

KCH = 9
IWD = 1
NFR = (510 - 1)/9 = 56

Therefore,

SIZE = 56 * 9 + 1 + 2 = 507

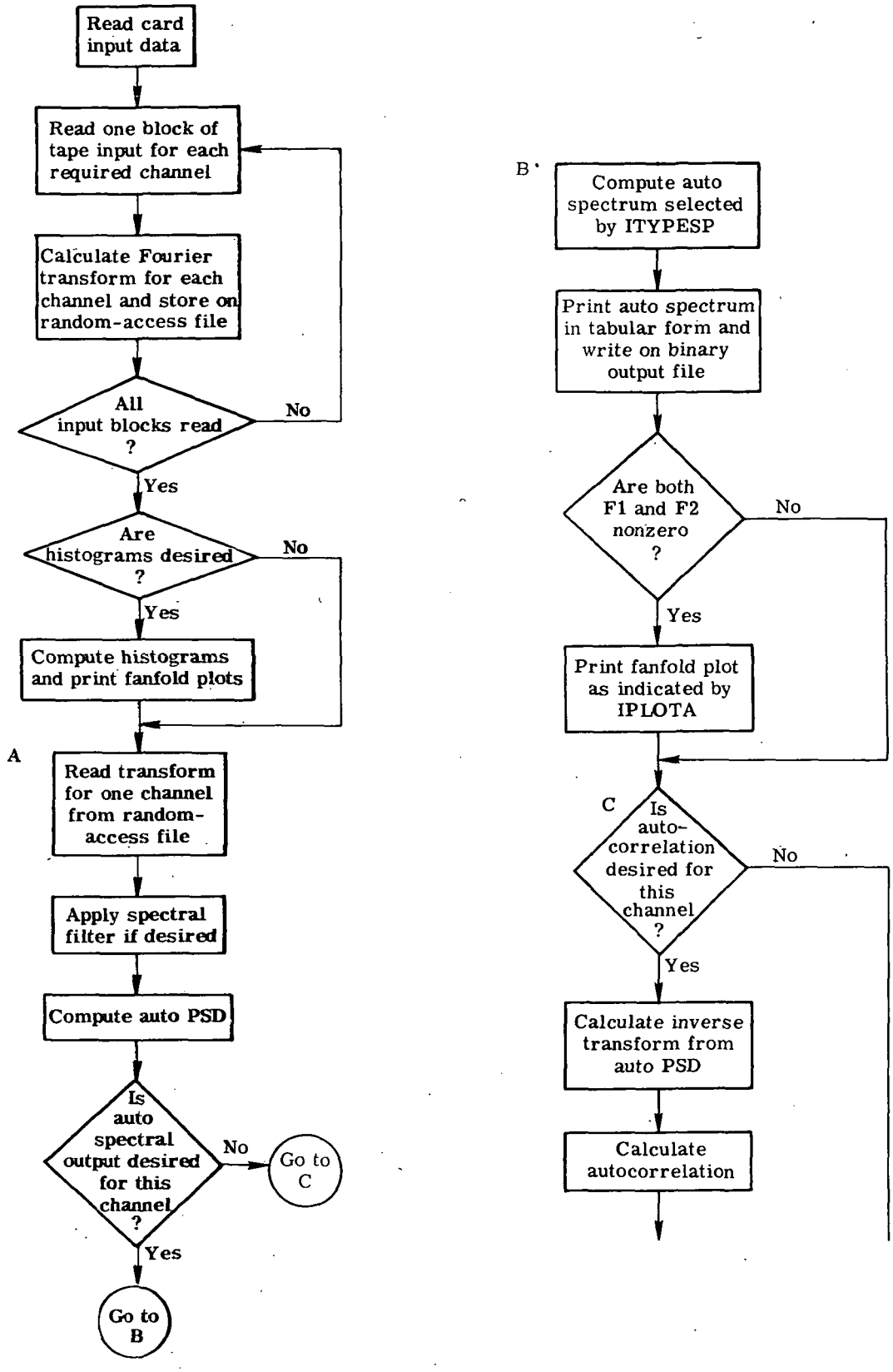
Tape Format 3

The binary tape is written by using subroutine RECOUT. The data passed to RECOUT at each time point are

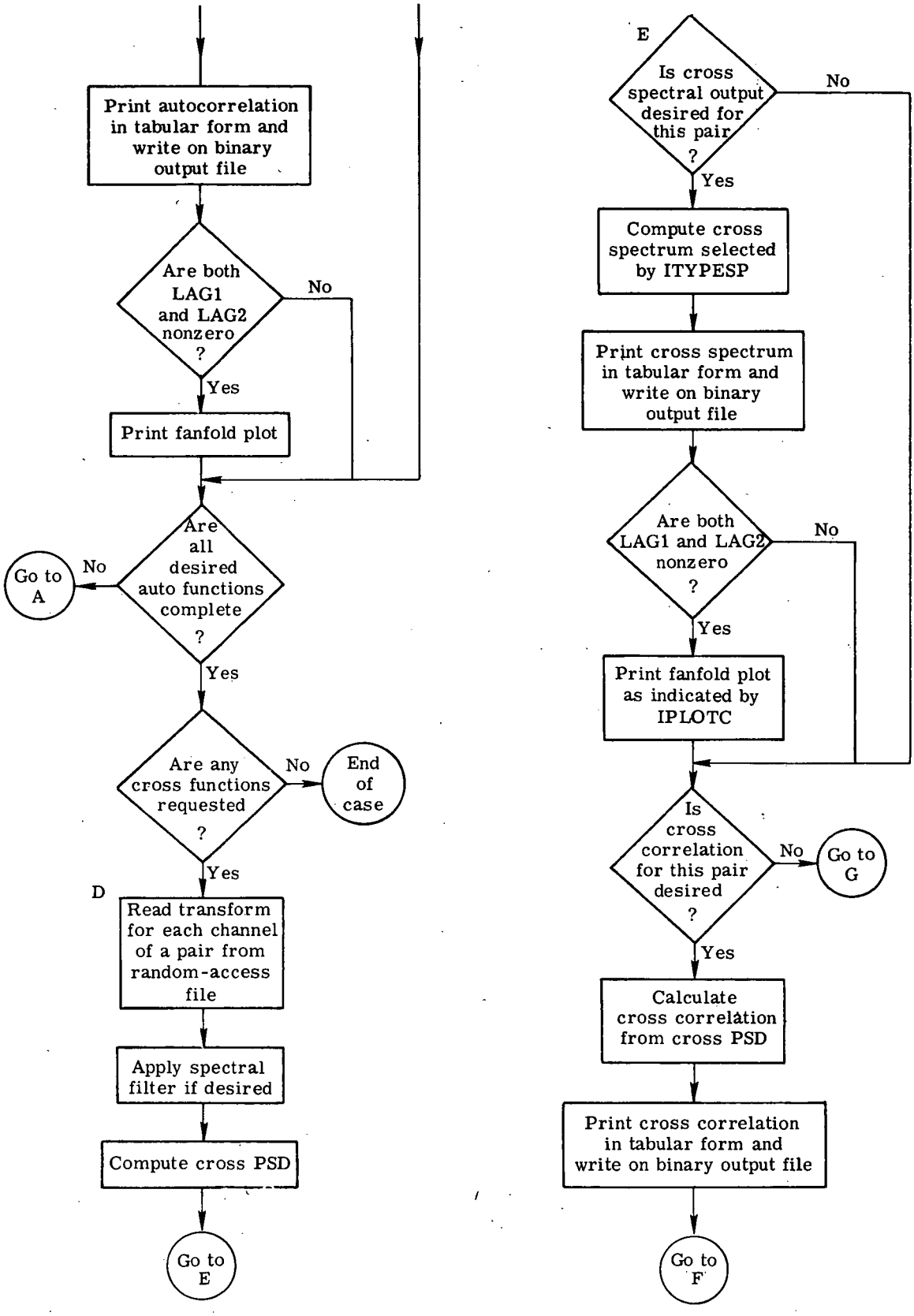
Word	Contents
1	Serial number
2	Time
3	Data channel 1
4	Data channel 2
.	.
.	.
.	.
NCH + 2	Data channel NCH

All words are in the floating-point mode.

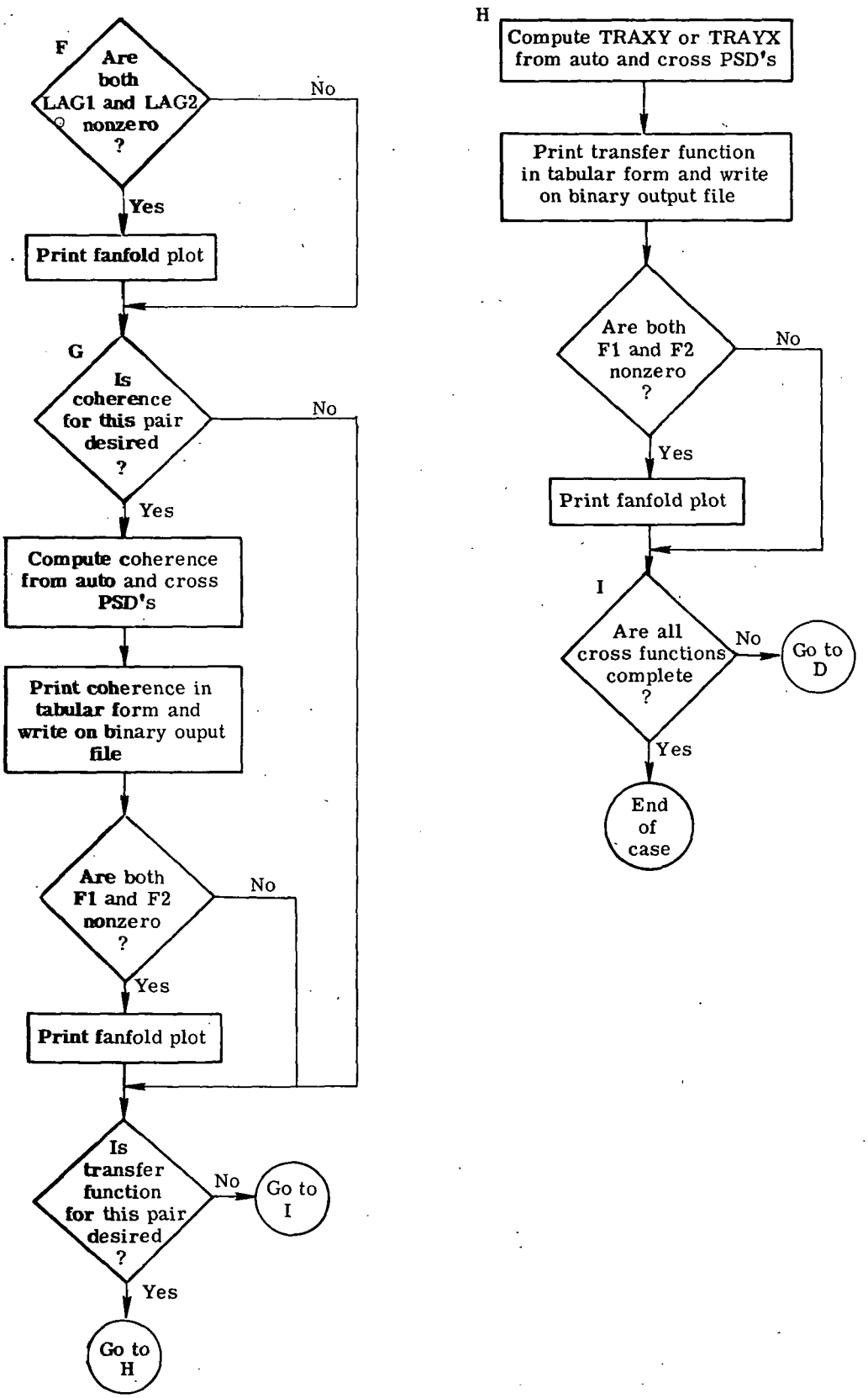
APPENDIX E – FLOWCHART FOR PATS



APPENDIX E


Print autocorrelation in tabular form and write on binary output file
Are both LAG1 and LAG2 nonzero ?
No
Yes
Print fanfold plot
Are all desired auto functions complete ?
No
Go to A
Yes
Are any cross functions requested ?
No
End of case
Yes
D
Read transform for each channel of a pair from random-access file
Apply spectral filter if desired
Compute cross PSD
Go to E
E
Is cross spectral output desired for this pair ?
No
Yes
Compute cross spectrum selected by ITYPESP
Print cross spectrum in tabular form and write on binary output file
Are both LAG1 and LAG2 nonzero ?
No
Yes
Print fanfold plot as indicated by IPLOTC
Is cross correlation for this pair desired ?
No
Go to G
Yes
Calculate cross correlation from cross PSD
Print cross correlation in tabular form and write on binary output file
Go to F

APPENDIX E


F
Are both LAG1 and LAG2 nonzero ?
No
Yes
Print fanfold plot
G
Is coherence for this pair desired ?
No
Yes
Compute coherence from auto and cross PSD's
Print coherence in tabular form and write on binary ouput file
Are both F1 and F2 nonzero ?
No
Yes
Print fanfold plot
Is transfer function for this pair desired ?
No
Go to I
Yes
Go to H
H
Compute TRAXY or TRAYX from auto and cross PSD's
Print transfer function in tabular form and write on binary output file
Are both F1 and F2 nonzero ?
No
Yes
Print fanfold plot
I
Are all cross functions complete ?
No
Go to D
Yes
End of case

APPENDIX F

PROGRAMS AND SUBPROGRAMS USED BY PATS

The programs and subprograms written specifically for PATS are given in the following list with a brief description of the purpose of each.

MAIN	Sets storage array dimensions, sets up random-access files 8 and 9, calls overlays for input, computations, and output
PLOTNB	Sets up arrays for plotting narrow-band spectra on fanfold References FANFOLD
FANFOLD	Plots an array in printed output with heading, max, min, and scale; the ordinate is across the page; the abscissa (index number in the array) is down the page, one point per line; up to 256 points per plot may be plotted
FOURT	Computes the Cooley-Tukey fast Fourier transform for an array of complex numbers; the number of points is arbitrary, although the subroutine operates much faster on powers of 2
READIN	Reads NAMELIST and FORTRAN READ input, checks for input errors, and prints informational messages, including accuracy measurement of spectral estimators References CSQ
CSQ	Computes value of chi-square for given level of significance and number of degrees of freedom References ITR2, FUNC
FUNC	Function subprogram used by CSQ to evaluate the chi-square probability function
BLOCKS	Calls subroutines to read data from binary input tape and perform Fourier transforms for given number of blocks of data References READTPE, TRAN
READTPE	Reads one block of data from binary input tape for selected channels and stores the data on random-access file 9; if overlap option is selected, each block after first takes the last half of the previous block and fills the rest of the block with new data References WRITMS, READMS, RECIN

TRAN	For each channel of data selected, this subroutine reads one block of input data from random-access file 9, counts occurrences for histograms, windows the data, extends the block with zeros if zero insertion option is selected, performs Fourier transform, and stores the results on random-access file 8 References READMS, WRITMS, HANNING, HAMMING, PARZEN, FOURT
HANNING	Weights the input array by the Hann data window
HAMMING	Weights the input array by the Hamming data window
PARZEN	Weights the input array by the Parzen data window
AUTOSP	Sets up storage arrays for subroutine AUTO and calls for histograms if selected; calls SPLINE to evaluate the spectral filter weighting function References AUTO, NORMAL, SPLINE
AUTO	For each selected channel of input data, computes the mean and variance of analyzed data (including overlap if used), reads all transforms for this channel from random-access file 8 and averages the amplitude spectra or PSD, applies the spectral filter, prints results, and calls selected plot routine; auto PSD is stored on random-access file 9; 1/3-octave spectra are calculated from the narrow-band spectra and printed; if autocorrelation is selected, the inverse transform of the auto PSD is performed and the result printed and plotted on fanfold References READMS, WRITMS, PLOTNB, BANDS, PLOTB, FOURT, ASCALE, FANFOLD
NORMAL	For each selected channel of input data, calls FANFOLD to plot histogram data, calculates chi-square for goodness-of-fit test, and prints the results References FANFOLD, PFUN
PFUN	Function used by NORMAL to calculate probability density function of a normally distributed random variable
PLOTB	Calls FANFOLD to plot 1/3-octave spectrum References FANFOLD
BANDS	Integrates narrow-band spectrum for 1/3-octave power spectrum References BNDSUM
BNDSUM	Computes sum of given array of complex numbers
CROSSP	Sets up storage arrays for CROSS References CROSS

CROSS	For each pair of channels: reads transforms for both channels for all blocks from random-access file 8, averages the products for cross PSD, prints the desired results, and calls for selected plots; the cross correlation is computed from the inverse transform of the cross PSD, and the results are printed and plotted on fanfold; coherence and transfer function are calculated from the auto PSD's stored on random-access file 9 and the results are printed and plotted on fanfold References READMS, PLOTNB, ENCODE, FOURT, ASCALE, FANFOLD
SPLINE	Fits a smooth curve to a set of input data points and evaluates the function at evenly incremented intervals over a given range References SIMEQ

APPENDIX G

LANGLEY LIBRARY SUBROUTINES

The Langley Library subroutines used by PATS are ASCALE, GAMMF, ITR2, OPENMS, READMS, RECIN, SIMEQ, and WRITMS. The subroutine RECOUT is not used by PATS but must be used separately to generate input data in tape format 3. Usage descriptions of all these subroutines are given in this appendix.

Subroutine ASCALE

Language: FORTRAN

Purpose: To compute a scaling factor for an array of numbers to be plotted over a certain area and find the minimum data value within the array.

Use: CALL ASCALE(ARRAY,S,N,K,DV), where

ARRAY	Name of the array containing the floating-point values to be scaled
S	Length (floating-point inches) over which the data are to be plotted (usually the length of one of the axes)
N	Number of data values in ARRAY from which points are to be plotted in accordance with K
K	Interleave factor which specifies the sequence in which data are stored: 1 indicates that values are stored sequentially 2 indicates that values are stored in every other location in the array
DV	Number of divisions per inch of the plotting paper to be used (should be 10.0, 20.0, 25.0, or 25.4)

Restrictions: The array must be dimensioned to include storage space for two extra elements per interleave factor. For example: N = 100, K = 1, DIMENSION ARRAY (102); N = 75, K = 3, DIMENSION ARRAY (231).

Method: This routine scans the elements in the array to find the minimum and maximum. ASCALE computes an adjusted minimum (origin value) and stores it in ARRAY((N*K)+1) and computes a scale factor and stores it in ARRAY((N*K)+1+K). The scale factor will be a power of 10 × (2,4,5, or 10). The data in the array may be scaled to floating-point inches by using a formula similar to the following: SV = (AE-MV)/SF, where SV is the scaled value, AE is the present value of array element, MV is either the minimum value or the value desired at the origin, and SF is the scale factor computed by the subroutine.

Storage: 262_8 locations for the CDC 6000 series.

Subprograms used: ALOG, ALOG10.

Other coding information: Example: DIMENSION ORD(102),ABS(204);CALL ASCALE(ORD,10.,100,1,10.); CALL ASCALE(ABS,15.,100,2,10.).

Subroutine date: September 3, 1970.

APPENDIX G

Function GAMMF

Language: FORTRAN

Purpose: To compute the incomplete gamma function

$$\Gamma(A,X) = \int_X^\infty e^{-\mu} \mu^{A-1} \, d\mu$$

If X = 0, then the complete gamma function is obtained.

Use: Y = GAMMF(A,X), where GAMMF(A,X) is defined as the integral from X to ∞ of exp(-μ) times μ to the (A-1)th power dμ.

Restrictions: X ≧ 0; when X = 0, A is not a nonpositive integer. The following subprograms are called by GAMMF: GSERES, GCHEB, GFRAC, GAMNEG.

Method: The method was originated by the AEC Computing and Applied Mathematics Center, Courant Institute of Mathematical Sciences, New York University.

(a) If A = 0,

$$\Gamma(0,X) = E_1(X) = -\left[\nu + \log(X) + \sum_{n=0}^{\infty} \frac{(-X)^n}{n.n!}\right]$$

(b) If A = -N, for some positive integer N,

$$\Gamma(-N,X) = \frac{(-1)^N}{N!}\left[E_1(X) - e^{-X} \sum_{j=0}^{N-1} \frac{(-1)^j j!}{X}\right]$$

(c) If X = 0,

$$\Gamma(A,0) = \int_0^\infty e^{-\mu} \mu^{A-1} \, d\mu = \Gamma(A)$$

which is the complete gamma function.

A rational Chebyshev approximation is used:

(d) For $A \neq 0$, $X < \sqrt{A+1}$,

$$\Gamma(A,X) = \Gamma(A) - X^A \sum_{n=0}^{\infty} \frac{(-X)^n}{(A+n)n!}$$

(e) For $A \neq 0$, $X \geqq \sqrt{|A|+1}$,

$$\Gamma(A,X) = e^{-X} X^A \left(\frac{1}{X+} \frac{1-A}{1+} \frac{1}{X+} \frac{2-A}{1+} \frac{2}{X+} \cdots\right)$$

APPENDIX G

Accuracy: Complete gamma function:

Test 1: A = 0.1(0.1)0.9 by formulas as in reference (a)

Test 2: A = 1.1(0.1)1.9 and 10.0(10.0)110.0

Incomplete gamma function:

Test 1: A = 1.0(0.1)2.0, X = 0.1(0.1)0.9 by formulas in reference (a)

All test results as compared with table entries of reference (a) were good to about 10 decimal places.

Reference: (a) Davis, Philip J.: Gamma Function and Related Functions. Handbook of Mathematical Functions, Milton Abramowitz and Irene A. Stegun, eds., Nat. Bur. Stand. Appl. Math. Ser. 55, U.S. Dep. Com., June 1964, pp. 253-293.

Storage: GAMMF 610_8 locations.

Coding information: GAMMF itself is a branching function which according to the values of A and X calls the following functions:

(a) GSERIES(A,X), which computes

$$\sum_{n=0}^{\infty} \frac{(-X)^n}{(A+n)n!}$$

(b) GCHEB(A), which computes by a rational Chebyshev approximation $\Gamma(A)$

(c) GFRAC(A,X), which computes the continued function for $\Gamma(A,X)$

(d) GAMNEG(IA,X), which computes $\Gamma(A,X)$ when A is a negative integer IA

(Because of the representation of numbers in the CDC 6600, of $A = -N \pm \epsilon$, where $\epsilon > 1.E - 10$, then A is taken to be a negative integer.)

Subprograms used: System library functions EXP, ALOG.

Function date: August 1, 1968.

APPENDIX G.

Subroutine ITR2

Language: FORTRAN

Purpose: Given F(X) = 0, to find a value for X within a given relative error, epsilon, in a given interval (a,b).

Use: CALL ITR2(X,A,B,DELTX,FOFX,E1,E2,MAXI,ICODE), where

X	The root
A	The lower bound on X; this value is used by ITR2 as an initial guess
B	The upper bound on X; this value is used by ITR2 as a final guess if the entire interval is scanned
DELTX	ΔX, the size of the scanning interval
FOFX	The name of a function subprogram to evaluate F(X)
E1	Relative error criterion
E2	Absolute error criterion
MAXI	A maximum iteration count supplied by the user
ICODE	An integer supplied by ITR2 as an error code; this code should be tested by the user on return to the calling program: 0 normal return 1 maximum iterations are exceeded 2 DELTX = 0 or negative 3 a root cannot be found within the given bounds 4 A > B

Restrictions: Make A < B, ΔX positive. A function subprogram with a single argument X must be written by the user to evaluate F(X). The name of this subprogram, FOFX, must appear in an EXTERNAL statement of the calling program.

Method: The given function F(X) is evaluated at a given starting point a and at intervals of a specified ΔX thereafter, up to and including a specified end point b. A change of sign of the function across a ΔX interval indicates a possible root in that interval. The interval is then halved successively toward F(X) = 0 until the prescribed accuracy is satisfied. The given function F(X) is evaluated once for each halving step.

If the given function is expected to have more than one root between the prescribed starting and end points, it is suggested that a sufficiently small value of ΔX be given so that no more than one root is present within a ΔX interval. A normal return is given upon the location of the first root from the starting point a. Additional roots must be located by new entries into the subroutine using a new starting point a which is just beyond the previous root.

APPENDIX G

Accuracy: The iteration process is continued until either of two convergence criteria is satisfied. These criteria are

If $|X_i| > \epsilon_1$,

$$\left|\frac{X_i - X_{i-1}}{X_i}\right| \leqq \epsilon_1$$

If $X_i \leqq \epsilon_1$,

$$|X_i - X_{i-1}| \leqq \epsilon_2$$

Storage: 260_8 locations.

Subroutine date: August 1, 1968.

APPENDIX G

Subroutine OPENMS

Language: COMPASS

Purpose: To open a random-access file.

Use: CALL OPENMS(U,IX,L,P), where

U The logical unit number

IX The first word address of the index

L The length of the index

P 0 for numbered indexing; 1 for named indexing

Restrictions: OPENMS must be the first operation on a random-access file. The file must be a disk file. For n index entries, the length of the index must be at least 2n + 1 if using named indexing, whereas the index length must be at least n + 1 for numbered indexing.

Method: OPENMS sets the first word in the index to a positive number for numbered indexing or to a negative number for named indexing. The random-access bit, index address, and index length are set by OPENMS into the FET of the file for system communication. If the file already exists, the master index is read into central memory.

Storage: 103_8 locations.

Subprograms used: System library subprograms GETBA, SIO$, SYSTEM.

Error messages: (1) UNASSIGNED MEDIUM FILE XXXXXX

(2) FILE DOES NOT RESIDE ON A RANDOM ACCESS DEVICE, XXXXXX

(3) INDEX BUFFER IS OF INSUFFICIENT LENGTH XXXXXX

XXXXXX is the file name. Termination is abnormal in each case.

Subroutine date: March 29, 1971.

APPENDIX G

Subroutine READMS

Language: COMPASS

Purpose: To read a record on a random-access file.

Use: CALL READMS(U,FWA,N,I), where

U	The logical unit number
FWA	The central memory address of the first word of the record
N	The number of words of the record to be transferred
I	The record number or record name depending upon the indexing mode set by the initial call to OPENMS

Restrictions: The file must have been opened by a call to OPENMS.

Method: The disk address of the record is determined using the index. If n words are requested to be transferred and there are m words in the record, where $m \leq n$, m words are transferred. If $m > n$, n words are transferred.

Storage: 131_8 locations.

Subprograms used: System library subprograms GETBA, SYSTEM, SIO$.

Error messages: (1) UNASSIGNED MEDIUM FILE XXXXXXX

(2) FILE WAS NOT OPENED BY A CALL TO SUBROUTINE OPENMS

(3) RECORD NAME REFERRED TO IN CALL IS NOT IN THE FILE INDEX

(4) *READ PARITY ERROR*

(5) SPECIFIED INDEX IN THIS MASS STORAGE CALL .GT. MASTER INDEX OR IS ZERO

Termination is abnormal.

Subroutine date: March 29, 1971.

APPENDIX G

Subroutine RECIN

Language: COMPASS

Purpose: To read binary records written by the subroutine RECOUT(J1.1).

Use: 1. Type 1 – Individual elements (not arrays):

CALL RECIN(LUN,IT,ICOUNT,L1,L2,. . .LN), where

LUN	Logical unit number
IT	Type, equal to 1
ICOUNT	Location reserved by the user; RECIN will store the following information in this location: 0, end-of-file; nonzero, number of words actually in the logical record; if the end-of-file flag was written by a call to RECOUT with IEOF = 1, then end-of-file testing must be done by testing ICOUNT for 0; if the end-of-file was written by an END FILE statement, then testing for end-of-file must be done by the IF(EOF,LUN) statement
L1,L2,. . .LN	Individual list elements

2. Type 2 – Arrays:

CALL RECIN(LUN,IT,ICOUNT,ARRAY,IFIRST,ILAST,INC), where

LUN	Logical unit number
IT	Type, equal to 2
ICOUNT	0, end-of-file; nonzero, number of words actually in the logical record (See ICOUNT under type 1)
ARRAY	Array name
IFIRST	First subscript
ILAST	Last subscript
INC	Increment

Examples: 1. CALL RECIN(1,1,K,A,B,ARRAY(1),ARRAY(2)).
Read a record from logical unit 1 into A, B, ARRAY(1), and ARRAY(2). Note that if the record contained only three words, K would equal 3 and ARRAY(2) would be unaltered.

2. CALL RECIN(1,2,K,ARRAY,1,39,2).
Read 20 words from logical unit 1 into ARRAY(1), ARRAY(3), . . ., ARRAY(39).

Restrictions: If RECIN is used on a file, the only other FORTRAN statements which may be used on that file are REWIND and IF(EOF,i).

The buffer size must be at least 2001_8.

RECIN must be used to read files written by RECOUT and only by RECOUT.

Method: RECIN reads into a central memory buffer physical records written by RECOUT, then passes to the user the requested logical record via a list giving the elements of the desired logical record. RECIN is analogous to a FORTRAN binary READ statement.

Storage: 301_8 locations.

Other coding information: Day file diagnostics and their meaning:

1. UNASSIGNED FILE MEDIUM FILE TAPEnn – No FET exists for this file. Every file has a file environment table that contains information describing the file to the system. This error would probably result because the file was not defined in the PROGRAM card or the user accidentally overwrote portions of his program.
2. BAD TYPE – The IT parameter was not 1 or 2.
3. UNCHECKED END FILE – The program attempted to read past EOF without testing for EOF.
4. READ/WRITE SEQUENCE ERROR – An attempt was made to read after writing.

Subroutine date: September 22, 1968.

APPENDIX G

Subroutine RECOUT

Language: COMPASS

Purpose: To write short binary records on a disk or tape in an optimum manner to increase peripheral processor and central processor efficiency. These records are to be read by RECIN(I1.1).

Use: RECOUT may be used for either tape or disk files.

1. Type 1 – Individual elements (not arrays):

   CALL RECOUT(LUN,IT,IEOF,L1,L2,. . .,LN), where

LUN	Logical unit number
IT	Type, equal to 1
IEOF	Equal to 1 if an end-of-file flag is desired, otherwise it must be zero. There are two methods by which the user may end his file. One method is to call RECOUT with IEOF = 1 when the last data record is written. This will cause an end-of-file flag (a short length record of less than 512_{10} CM words) to be written. RECIN is programed to sense this and will set ICOUNT = 0 when sensed. If this method is used, the user must set IEOF = 1 when outputting his last data record since RECOUT should not be called with an empty list. For all other calls to RECOUT, IEOF must be set to 0. The other method of ending the file is to use the END FILE statement. This is the most convenient way of ending the file
L1,L2,. . .LN	Individual list elements

2. Type 2 – Arrays:

   CALL RECOUT(LUN,IT,IEOF,ARRAY,IFIRST,ILAST,INC), where

LUN	Logical unit number
IT	Type, equal to 2
IEOF	Equal to 1 if an end-of-file desired; equal to 0 if no end-of-file (see explanation under type 1)
ARRAY	Array name
IFIRST	First subscript
ILAST	Last subscript
INC	Increment

Examples: 1. CALL RECOUT(1,1,0,A,B,ARRAY(1),ARRAY(2)).
Write a record on logical unit 1 containing A, B, ARRAY(1), ARRAY(2).

2. CALL RECOUT(1,2,0,ARRAY,1,20,1).
Write a record containing ARRAY(1) through ARRAY(20). This is equivalent to WRITE(1) (ARRAY(I), I = 1, 20).

Restrictions: If RECOUT is used on a file, the only other FORTRAN statements which may be used on that file are REWIND and END FILE.

The buffer size must be at least 2001_8. A normal FORTRAN buffer is this size.

Files written with RECOUT must be read with RECIN.

If the list to be written in a logical record is larger than 511_{10} CM words, then RECOUT offers no advantage and should not be used.

If the programer wishes to write a file containing multifiles using RECOUT, then he must end each file by setting IEOF = 1 and not by using the END FILE statement. Consequently, he should then test for end-of-file in RECIN by testing ICOUNT for zero.

Method: Under the CDC SCOPE 3.0 operating system, each binary write commanded by the FORTRAN statement WRITE(LUN). . . causes one or more physical records to be output to either a disk or tape file. If the logical record size written by the programer is small and the number of records processed is large, then excessive usage of I/O routines and equipment results. To decrease this I/O time, RECOUT blocks binary data into an optimum record size (512_{10} CM words) in a central memory buffer before transmitting it to the actual disk or tape file.

Storage: 320_8 locations.

Other coding information: Day file diagnostics and their meaning:

1. UNASSIGNED FILE MEDIUM FILE TAPEnn – No FET exists for the file. Every file has a file environment table that contains information describing the file to the system. This error would probably result because the file was not defined in the PROGRAM card or the user accidentally overwrote portions of his program.
2. BAD TYPE – The IT parameter was not 1 or 2.
3. BUFFER TOO SMALL – The buffer size was less than 2001_8.
4. BAD PARAM COUNT – The number of parameters in the call was illegal.
5. WRITE/READ SEQUENCE ERROR – A write request was made after a read request.

Source: CDC.

Subroutine date: September 23, 1968.

APPENDIX G

Subroutine SIMEQ

Language: FORTRAN

Purpose: SIMEQ solves the matrix equation AX = B where A is a square coefficient matrix and B is a matrix of constant vectors. The solution to a set of simultaneous equations and the determinant may be obtained. If the user wants the determinant only, use DETEV for savings in time and storage.

Use: CALL SIMEQ (A, N, B, M, DETERM, IPIVOT, NMAX, ISCALE)

A	A two-dimensional array of the coefficients.
N	The order of A; $1 \leqq N \leqq NMAX$.
B	A two-dimensional array of the constant vectors B. On return to calling program, X is stored in B.
M	The number of column vectors in B.
DETERM	Gives the value of the determinant by the following formula: $DET(A) = \left(10^{100}\right)^{ISCALE}(DETERM)$
IPIVOT	A one-dimensional array of temporary storage used by the routine.
NMAX	The maximum order of A as stated in dimension statement of calling program.
ISCALE	A scale factor computed by subroutine to keep results of computation within the floating-point word size of the computer.

Restrictions: Arrays A, B, and IPIVOT are dimensioned with variable dimensions in the subroutine. The maximum size of these arrays must be specified in a DIMENSION statement of the calling program as: A (NMAX, NMAX), B (NMAX, M), IPIVOT (NMAX). The original matrices, A and B, are destroyed. They must be saved by the user if there is further need for them. The determinant is set to zero for a singular matrix.

Method: Jordan's method is used through a succession of elementary transformations: $l_n, l_{n-1}, \ldots, l_1$. If these transformations are applied to a matrix B of constant vectors, the result is X where AX = B. Each transformation is selected so that the largest element is used in the pivotal position.

Accuracy: Total pivotal strategy is used to minimize the rounding errors; however, the accuracy of the final results depends upon how well-conditioned the original matrix is.

Reference: (a) Fox, L.: An Introduction to Numerical Linear Algebra. Oxford Univ. Press, c.1965.

Storage: 432_8 locations.

Subroutine date: August 1, 1968.

APPENDIX G

Subroutine WRITMS

Language: COMPASS

Purpose: To write a record on a random-access file.

Use: CALL WRITMS(U,FWA,N,I), where

U	Logical unit number
FWA	Central memory address of the first word of the record
N	Number of central memory words to be transferred
I	Record number or record name depending upon the indexing mode set by the initial call to OPENMS

Restrictions: The file must have been opened by a call to OPENMS.

Method: The specified record is written on the file and an address entered in the index to reference the record.

Storage: 102_8 locations.

Subprograms used: System library subprograms GETBA, SYSTEM, SIO$.

Error messages: (1) UNASSIGNED MEDIUM FILE XXXXXXX

(2) FILE WAS NOT OPENED BY A CALL TO SUBROUTINE OPENMS

(3) INDEX BUFFER IS OF INSUFFICIENT LENGTH

Subroutine date: March 29, 1971.

APPENDIX H

SOURCE LISTING OF PATS

```
      OVERLAY(PATS,0,0)
      PROGRAM MAIN(INPUT,OUTPUT,TAPE1,TAPE5=INPUT,TAPE6=OUTPUT,TAPE7,
     1TAPE8,TAPE9)
      COMMON CMAIN(4102)
C
C     CMAIN MUST BE DIMENSIONED 4*NMAX+6
C
C     KNDEX MUST BE DIMENSIONED NBCMAX+2
      DIMENSION INDEX(129),KNDEX(8021
      COMMON/BLK1/STARTT,ITFMT,NBLK,IPOW2,NCH,NPRINT,IPLOTA,IPLOTC,OFFSC
     1AL,DELTAT,SN,NRSKIP,LAP,NCROSS,ICROSS(2,20),NCHP,YLABEL(2),IWINDOW
     1,F1,F2,ITYPESP,WCON,NFSKIP,IFF,LAG1,LAG2
      COMMON/BLK2/ICH(14),CHSUM(14),NOFF(14),CHSUMSQ(14),SIGMA(14),RMS(1
     114),MEAN(14),SCALFAC(14),CHSUM1(14),TRACK(14),ICHAN(14),IFILTER(14)
      COMMON/BLK3/NPT,TMAX,NPTO2,NSPCT,DELF,N64,NPTO128
     1,INZERO,NREAD,NPTOT
      COMMON/BLK4/NMAX,NCHMAX,NBCMAX
      COMMON/BLK5/PCTC,NBINS,DMAX(14),DMIN(14),DBIN(14),BINS(100,14)
     1,CHISQC
      COMMON/BLK6/ISAVE64,IRI,IXPLOT,IDATA,IZ,ISPECT
      COMMON/BLK7/IWORD(11)
      COMMON/BLK8/IAUTOSP(14),IAUTOCO(14),ICRSP(20),ICRCOR(20),ITRA(20),
     1ICOH(20)
      COMMON/BLK9/NFILTP,FREQF(50),WGHTF(50)
      COMMON/BLK10/NRCRD7
      DATA NRCRD7/0/
C
      NMAX=1024
      NCHMAX=14
      NBCMAX=800
      IZ=IRI=1
      IXPLOT=IRI+NMAX+4
      ISPECT=ISAVE64=IXPLOT+NMAX+2
      IDATA=ISAVE64+64*NCHMAX
      PATS=4HPATS
      CALL OPENMS(9,INDEX,129,0)
      CALL OPENMS(8,KNDEX,NBCMAX+2,0)
      PRINT 1000
 1000 FORMAT(1H1//////////23X1H*43(2H *)/23X1H*85X1H*/23X1H*8X*P R O G R
     1 A M   F O R   A N A L Y S I S   O F   T I M E   S E R I E S*8X1H*
     2/23X1H*85X1H*/ 23X1H*5X*C.G.BROWN, T.J.BROWN, AND J.C.HARDIN FOR A
     3COUSTICS DIVISION, NASA-LRC, 1973*5X1H*/23X1H*85X1H* /23X1H*43(24
     4*))
C
C     READ NAMELIST AND CARD INPUT
C
  100 CALL OVERLAY(PATS,1,0)
C
C     READ ONE BLOCK OF INPUT TIME SERIES DATA FOR NCH CHANNELS FROM
C     BINARY TAPE AND STORE ON RANDOM ACCESS FILE
```

APPENDIX H

```
C     COMPUTE FOURIER TRANSFORM FOR NCH CHANNELS AND STORE ON RANDOM
C     ACCESS FILE
C     REPEAT UNTIL ALL DATA IS READ AND PROCESSED.
C
      CALL OVERLAY(PATS,2,0)
C
C     AVERAGE NBLK POWER SPECTRA FOR EACH CHANNEL FOR AUTO SPECTRA AND
C     APPLY SPECTRAL FILTER. COMPUTE AUTOCORRELATION. PRINT FANFOLD PLOTS
C
      CALL OVERLAY(PATS,3,0)
C
C     TO COMPUTE CROSS SPECTRA
C     GET NBLK TRANSFORMS FOR BOTH OF THE CHANNELS IN EACH SELECTED PAIR
C     AVERAGE THE PRODUCTS FOR NARROW BAND SPECTRA. COMPUTE CROSSCORRELATIONS,
C     COHERENCE, AND TRANSFER FUNCTIONS. PRINT FANFOLD PLOTS.
C
      IF(NCROSS.GT.0) CALL OVERLAY(PATS,4,0)
C
      GO TO 100
      END
      SUBROUTINE PLOTNB(YLABEL,FRAMEL,NF,XPLOT,RI,NSPCT,ILOG,F1,F2,PLABE
     1L,NP,IFF,ISEARCH)
      DIMENSION YLABEL(2),FRAMEL(1),XPLOT(1),RI(1),PLABEL(5),FFID(5),
     1IDEN(6)
C     ILOG - CODE FOR TYPE OF SCALE FOR AXES
C     = 0 BOTH SCALES LINEAR
C     = 1 HORIZ. SCALE LOG, VERT. SCALE LINEAR
C     = 2 HORIZ. SCALE LINEAR, VERT. SCALE LOG
C     = 3 BOTH SCALES LOG
C
C     OPTIONS 1 AND 3 ARE NOT USED BY PATS
C
      ILOGP1=ILOG+1
      IF(ISEARCH.EQ.0) GO TO 6
      J1=1 $ J2=NSPCT
      DO 1 I=1,NSPCT
      J=I
      IF(XPLOT(I).GE.F1) GO TO 2
    1 CONTINUE
    2 J1=J $ IF(ILOG.EQ.1.OR.ILOG.EQ.3) J1=MAXO(J1,2)
      DO 3 I=J1,NSPCT
      J=NSPCT-I+J1
      IF(XPLOT(J).LE.F2) GO TO 4
    3 CONTINUE
    4 J2=J
      NFW=NF/10 $ IF(NFW*10.LT.NF) NFW=NFW+1
      NFP=NF/10 $ IF(NFP*10.LT.NP) NFP=NFP+1
      IF(J1.LT.J2) GO TO 5
      PRINT 900, (PLABEL(I),I=1,NFP),(FRAMEL(I),I=1,NFW)
  900 FORMAT(*0BANDWIDTH FOR PLOTS TOO NARROW*/* NO PLOT GENERATED FOR *
     1I0A10)
```

```
      RETURN
    5 NPLOT=J2-J1+1
      IF(NPLOT.GE.NSPCT) GO TO 100
C     MOVE PLOTTING REGION TO BEGINNING OF ARRAY
      DO 4002 I=J1,J2
      J=I-J1+1
 4002 XPLOT(J)=XPLOT(I)
    6 CONTINUE
      DO 4003 I=J1,J2
      J=I-J1+1
      RI(J)=RI(I)
 4003 CONTINUE
  100 CONTINUE
  103 GO TO (105,105,106,106),ILOGP1
C     LINEAR VERTICAL SCALE
  105 CALL ASCALE(RI,10.,NPLOT,1,10.)
      YMIN=RI(NPLOT+1) $ YMAX=YMIN+10.*RI(NPLOT+2)
      GO TO 107
C     LOG VERTICAL SCALE
  106 SMAX=-100.
      DO 108 I=1,NPLOT $ IF(RI(I)) 112,112,113
  112 RI(I)=-100. $ GO TO 108
  113 RI(I)=ALOG10(RI(I)) $ SMAX=AMAX1(RI(I),SMAX)
  108 CONTINUE
      IMAX=SMAX
      IF(IMAX.LT.SMAX) IMAX=IMAX+1
      IMIN=IMAX-5
      YMIN=RI(NPLOT+1)=IMIN $ YMAX=IMAX $ RI(NPLOT+2)=.5
      DO 114 I=1,NPLOT
      IF(RI(I).LT.YMIN) RI(I)=YMIN
  114 CONTINUE
  107 CONTINUE
      IF(IFF.EQ.0) GO TO 24
      IF(NPLOT.GT.256) NPLOT=256
      FFID(1)=YLABEL(1) $ FFID(2)=YLABEL(2) $ FFID(3)=FRAMEL(1)
      FFID(4)=FRAMEL(2) $ FFID(5)=FRAMEL(3)
      IF(ILOG.GT.1) ENCODE(54,901,IDEN) PLABEL
      IF(ILOG.LE.1) ENCODE(60,902,IDEN) PLABEL
  902 FORMAT(5X,5A10,5X)
      CALL FANFOLD(RI,NPLOT,1,1,NPLOT,IDEN,1H*,1.,YMAX,YMIN,FFID,NFW+2,
     1132,0,XPLOT,9HFREQUENCY)
  901 FORMAT(*LOG *5A10)
   24 RETURN
      END
      SUBROUTINE FANFOLD(PLOT,NPT,K,NF,NMAX,IDEN,CHAR,PNORM,PMAX,PMIN,
     1YLABEL,NYL,LINE,IWRITEX,XARRAY,XLABEL)
      DIMENSION PLOT(NMAX,1),CHAR(1),IDEN(5,1),PNORM(1),YLABEL(2)
     1,PMAX(1 ),PMIN(1 ),PSCALE(10),PLINE(124),XARRAY(1)
      IF(LINE-120) 7,7,8
    7 NCHAR=104 $ GO TO 9
    8 NCHAR=124
```

```
  9 CONTINUE
    IF(NF.GT.10) NF=10
    PRINT 900,(YLABEL(I),I=1,NYL)
900 FORMAT(*1*48X8A10)
    IF(IWRITEX.EQ.0) GO TO 10
    PRINT 907, (XLABEL,J=1,6),(J,XARRAY(J),J=1,NPT,K)
907 FORMAT(//41X*THE *A10,* CODE SCALE IS AS FOLLOWS*/5X*CODE  *A10,4(
   16X*CODE  *A10)/(I8,E14.5,I8,E14.5,I8,E14.5,I8,E14.5,I8,E14.5))
 10 CONTINUE
    NSKIP=0
    IF(NPT/K-15) 16,16,17
 16 NSKIP=2 $ GO TO 13
 17 IF(NPT/K-25) 18,18,13
 18 NSKIP=1
 13 CONTINUE
    DO 1 I=1,NF
    IF(PMIN(I).NE.PMAX(I)) GO TO 1
    PMIN(I)=PMAX(I)=PLOT(1,I)
    DO 2 J=1,NPT,K
    PMIN(I)=AMIN1(PMIN(I),PLOT(J,I))
    PMAX(I)=AMAX1(PMAX(I),PLOT(J,I))
  2 CONTINUE
  1 PSCALE(I)=(PMAX(I)-PMIN(I))*PNORM(I)/(NCHAR-4.)
    PRINT 901,(I,CHAR(I),(IDEN(J,I),J=1,5),PNORM(I),PMAX(I),PMIN(I),
   1PSCALE(I),I=1,NF)
901 FORMAT(///58X*PLOT DESCRIPTION*/* FUNCTION*67X*SCALE*/*   NO.*4X*C
   1HARACTER*5X*IDENTIFICATION*37X*FACTOR*9X*MAXIMUM*8X*MINIMUM*6X*RES
   2OLUTION*/(I5,9X,A1,5X,5A10,4E15.5))
    PLINE(2)=1H( $ PLINE(NCHAR)=1H) $ PLINE(1)=1H
    IF(NCHAR.EQ.124) PRINT 902
    IF(NCHAR.EQ.104) PRINT 905
902 FORMAT(                                  //20X*1    1    2    2    3
   1 3    4    4    5    5    6    6    7    7    8    8    9    9    1
   20   10   11   11   12*/10X*0....5....0....5....0....5....0....5....0....5...
   3.0....5....0....5....0....5....0....5....0....5....0....5....0....5....0....
   +5....0....5....0*)
905 FORMAT(                                  //20X*1    1    2    2    3
   1 3    4    4    5    5    6    6    7    7    8    8    9    9    1
   20*/10X*0....5....0....5....0....5....0....5....0....5....0....5....0....5...
   3.0....5....0....5....0....5....0....5....0....5....0*)
    IF(NSKIP.EQ.0) GO TO 20
    DO 21 I=1,NSKIP
 21 PRINT 908
908 FORMAT(1H )
 20 CONTINUE
    DO 3 J=1,NPT,K
    DO 5 I=4,NCHAR
  5 PLINE(I-1)=1H
    DO 4 I=1,NF
    P=(PLOT(J,I)-PMIN(I))/PSCALE(I)
    IP=P+3.5
```

APPENDIX H

```
      IF(IP.LE.2..OR.IP.GE.NCHAR) GO TO 4
      IF(PLINE(IP).NE.1H ) GO TO 6
      PLINE(IP)=CHAR(I)
      GO TO 4
    6 PLINE(IP)=1HX
    4 CONTINUE
      PRINT 903, J,(PLINE(I),I=1,NCHAR)
  903 FORMAT(I8,124A1)
      IF(NSKIP.EQ.0) GO TO 3
      DO 19 I=1,NSKIP
   19 PRINT 908
    3 CONTINUE
      IF(NCHAR.EQ.124) PRINT 904
      IF(NCHAR.EQ.104) PRINT 906
  904 FORMAT(10X*0....5....0....5....0....5....0....5....0....5....0....5....0....
     15....0....5....0....5....0....5....0....5....0....5....0....5....0....5....0
     3*/                              20X*1     1    2    2    3    3    4
     1 4    5    5    6    6    7    7    8    8    9    9   10   10   1
     21   11   12*)
  906 FORMAT(10X*0....5....0....5....0....5....0....5....0....5....0....5....0....
     15....0....5....0....5....0....5....0....5....0....5....0*/20X*1     1    2
     2 2    3    3    4    4    5    5    6    6    7    7    8    8
     39    9   10*)
      RETURN
      END
      SUBROUTINE FOURT(DATA,NN,NDIM,ISIGN,IFORM,WORK)                    0002
C                                                                        0003
C     THE COOLEY-TUKEY FAST FOURIER TRANSFORM IN USASI BASIC FORTR       0004
C                                                                        0005
C     TRANSFORM(K1,K2,...) = SUM(DATA(J1,J2,...)*EXP(ISIGN*2*PI*SQ       0006
C     *((J1-1)*(K1-1)/NN(1)+(J2-1)*(K2-1)/NN(2)+...))), SUMMED FOR       0007
C     J1, K1 BETWEEN 1 AND NN(1), J2, K2 BETWEEN 1 AND NN(2), ETC.       0008
C     THERE IS NO LIMIT TO THE NUMBER OF SUBSCRIPTS.  DATA IS A          0009
C     MULTIDIMENSIONAL COMPLEX ARRAY WHOSE REAL AND IMAGINARY            0010
C     PARTS ARE ADJACENT IN STORAGE, SUCH AS FORTRAN IV PLACES THE       0011
C     IF ALL IMAGINARY PARTS ARE ZERO (DATA ARE DISGUISED REAL), S       0012
C     IFORM TO ZERO TO CUT THE RUNNING TIME BY UP TO FORTY PERCENT       0013
C     OTHERWISE, IFORM = +1.  THE LENGTHS OF ALL DIMENSIONS ARE          0014
C     STORED IN ARRAY NN, OF LENGTH NDIM.  THEY MAY BE ANY POSITIV       0015
C     INTEGERS, THO THE PROGRAM RUNS FASTER ON COMPOSITE INTEGERS,       0016
C     ESPECIALLY FAST ON NUMBERS RICH IN FACTORS OF TWO.  ISIGN IS       0017
C     OR -1.  IF A -1 TRANSFORM IS FOLLOWED BY A +1 ONE (OR A +1         0018
C     BY A -1) THE ORIGINAL DATA REAPPEAR, MULTIPLIED BY NTOT (=NN       0019
C     NN(2)*...).  TRANSFORM VALUES ARE ALWAYS COMPLEX, AND ARE RE       0020
C     IN ARRAY DATA, REPLACING THE INPUT.  IN ADDITION, IF ALL           0021
C     DIMENSIONS ARE NOT POWERS OF TWO, ARRAY WORK MUST BE SUPPLIE       0022
C     COMPLEX OF LENGTH EQUAL TO THE LARGEST NON 2**K DIMENSION.         0023
C     OTHERWISE, REPLACE WORK BY ZERO IN THE CALLING SEQUENCE.           0024
C     NORMAL FORTRAN DATA ORDERING IS EXPECTED, FIRST SUBSCRIPT VA       0025
C     FASTEST.  ALL SUBSCRIPTS BEGIN AT ONE.                             0026
C                                                                        0027
```

APPENDIX H

```
C     RUNNING TIME IS MUCH SHORTER THAN THE NAIVE NTOT**2, BEING      0028
C     GIVEN BY THE FOLLOWING FORMULA.  DECOMPOSE NTOT INTO            0029
C     2**K2 * 3**K3 * 5**K5 * ....  LET SUM2 = 2*K2, SUMF = 3*K3 +    0030
C     + ... AND NF = K3 + K5 + ....  THE TIME TAKEN BY A MULTI-       0031
C     DIMENSIONAL TRANSFORM ON THESE NTOT DATA IS T = T0 + NTOT*(T     0032
C     T2*SUM2+T3*SUMF+T4*NF).  ON THE CDC 3300 (FLOATING POINT ADD     0033
C     OF SIX MICROSECONDS), T = 3000 + NTOT*(500+43*SUM2+68*SUMF+      0034
C     320*NF) MICROSECONDS ON COMPLEX DATA.  IN ADDITION, THE          0035
C     ACCURACY IS GREATLY IMPROVED, AS THE RMS RELATIVE ERROR IS       0036
C     BOUNDED BY 3*2**(-B)*SUM(FACTOR(J)**1.5), WHERE B IS THE NUM     0037
C     OF BITS IN THE FLOATING POINT FRACTION AND FACTOR(J) ARE THE     0038
C     PRIME FACTORS OF NTOT.                                           0039
C                                                                      0040
C     THE DISCRETE FOURIER TRANSFORM PLACES THREE RESTRICTIONS UPO     0048
C     DATA.                                                            0049
C     1.  THE NUMBER OF INPUT DATA AND THE NUMBER OF TRANSFORM VAL     0050
C     MUST BE THE SAME.                                                0051
C     2.  BOTH THE INPUT DATA AND THE TRANSFORM VALUES MUST REPRES     0052
C     EQUISPACED POINTS IN THEIR RESPECTIVE DOMAINS OF TIME AND        0053
C     FREQUENCY.  CALLING THESE SPACINGS DELTAT AND DELTAF, IT MUS     0054
C     TRUE THAT DELTAF=2*PI/(NN(I)*DELTAT).  OF COURSE, DELTAT NEE     0055
C     BE THE SAME FOR EVERY DIMENSION.                                 0056
C     3.  CONCEPTUALLY AT LEAST, THE INPUT DATA AND THE TRANSFORM      0057
C     REPRESENT SINGLE CYCLES OF PERIODIC FUNCTIONS.                   0058
C                                                                      0059
C     EXAMPLE 1.  THREE-DIMENSIONAL FORWARD FOURIER TRANSFORM OF A     0060
C     COMPLEX ARRAY DIMENSIONED 32 BY 25 BY 13 IN FORTRAN IV.          0061
C     DIMENSION DATA(32,25,13),WORK(50),NN(3)                          0062
C     COMPLEX DATA                                                     0063
C     DATA NN/32,25,13/                                                0064
C     DO 1 I=1,32                                                      0065
C     DO 1 J=1,25                                                      0066
C     DO 1 K=1,13                                                      0067
C  1  DATA(I,J,K)=COMPLEX VALUE                                        0068
C     CALL FOURT(DATA,NN,3,-1,1,WORK)                                  0069
C                                                                      0070
C     EXAMPLE 2.  ONE-DIMENSIONAL FORWARD TRANSFORM OF A REAL ARRA     0071
C     LENGTH 64 IN FORTRAN II.                                         0072
C     DIMENSION DATA(2,64)                                             0073
C     DO 2 I=1,64                                                      0074
C     DATA(1,I)=REAL PART                                              0075
C  2  DATA(2,I)=0.                                                     0076
C     CALL FOURT(DATA,64,1,-1,0,0)                                     0077
C                                                                      0078
      DIMENSION DATA(2),NN(1),IFACT(32),WORK(1)                        0079
      WI=1.00                                                          0080
      WR=1.00                                                          0081
      WSTPR=1.00                                                       0082
      WSTPI=1.00                                                       0083
      TWOPI=6.283185307                                                0084
      IF(NDIM-1)920,1,1                                                0085
```

```
1       NTOT=2                                                          0086
        DO 2 IDIM=1,NDIM                                                0087
        IF(NN(IDIM))920,920,2                                           0088
2       NTOT=NTOT*NN(IDIM)                                              0089
C                                                                       0090
C       MAIN LOOP FOR EACH DIMENSION                                    0091
C                                                                       0092
        NP1=2                                                           0093
        DO 910 IDIM=1,NDIM                                              0094
        N=NN(IDIM)                                                      0095
        NP2=NP1*N                                                       0096
        IF(N-1)920,900,5                                                0097
C                                                                       0098
C       FACTOR N                                                        0099
C                                                                       0100
5       M=N                                                             0101
        NTWO=NP1                                                        0102
        IF=1                                                            0103
        IDIV=2                                                          0104
10      IQUOT=M/IDIV                                                    0105
        IREM=M-IDIV*IQUOT                                               0106
        IF(IQUOT-IDIV)50,11,11                                          0107
11      IF(IREM)20,12,20                                                0108
12      NTWO=NTWO+NTWO                                                  0109
        M=IQUOT                                                         0110
        GO TO 10                                                        0111
20      IDIV=3                                                          0112
30      IQUOT=M/IDIV                                                    0113
        IREM=M-IDIV*IQUOT                                               0114
        IF(IQUOT-IDIV)60,31,31                                          0115
31      IF(IREM)40,32,40                                                0116
32      IFACT(IF)=IDIV                                                  0117
        IF=IF+1                                                         0118
        M=IQUOT                                                         0119
        GO TO 30                                                        0120
40      IDIV=IDIV+2                                                     0121
        GO TO 30                                                        0122
50      IF(IREM)60,51,60                                                0123
51      NTWO=NTWO+NTWO                                                  0124
        GO TO 70                                                        0125
60      IFACT(IF)=M                                                     0126
C                                                                       0127
C       SEPARATE FOUR CASES--                                           0128
C          1. COMPLEX TRANSFORM OR REAL TRANSFORM FOR THE 4TH, 5TH,E    0129
C             DIMENSIONS.                                               0130
C          2. REAL TRANSFORM FOR THE 2ND OR 3RD DIMENSION.  METHOD--    0131
C             TRANSFORM HALF THE DATA, SUPPLYING THE OTHER HALF BY C    0132
C             JUGATE SYMMETRY.                                          0133
C          3. REAL TRANSFORM FOR THE 1ST DIMENSION, N ODD.  METHOD--    0134
C             TRANSFORM HALF THE DATA AT EACH STAGE, SUPPLYING THE O    0135
C             HALF BY CONJUGATE SYMMETRY.                               0136
```

```
C         4. REAL TRANSFORM FOR THE 1ST DIMENSION, N EVEN.  METHOD-
C            TRANSFORM A COMPLEX ARRAY OF LENGTH N/2 WHOSE REAL PAR
C            ARE THE EVEN NUMBERED REAL VALUES AND WHOSE IMAGINARY
C            ARE THE ODD NUMBERED REAL VALUES.  SEPARATE AND SUPPLY
C            THE SECOND HALF BY CONJUGATE SYMMETRY.
C
70      NON2=NP1*(NP2/NTWO)
        ICASE=1
        IF(IDIM-4)71,90,90
71      IF(IFORM)72,72,90
72      ICASE=2
        IF(IDIM-1)73,73,90
73      ICASE=3
        IF(NTWO-NP1)90,90,74
74      ICASE=4
        NTWO=NTWO/2
        N=N/2
        NP2=NP2/2
        NTOT=NTOT/2
        I=3
        DO 80 J=2,NTOT
        DATA(J)=DATA(I)
80      I=I+2
90      I1RNG=NP1
        IF(ICASE-2)100,95,100
95      I1RNG=NPO*(1+NPPEV/2)
C
C       SHUFFLE ON THE FACTORS OF TWO IN N.  AS THE SHUFFLING
C       CAN BE DONE BY SIMPLE INTERCHANGE, NO WORKING ARRAY IS NEEDE
C
100     IF(NTWO-NP1)600,600,110
110     NP2HF=NP2/2
        J=1
        DO 150 I2=1,NP2,NON2
        IF(J-I2)120,130,130
120     I1MAX=I2+NON2-2
        DO 125 I1=I2,I1MAX,2
        DO 125 I3=I1,NTOT,NP2
        J3=J+I3-I2
        TEMPR=DATA(I3)
        TEMPI=DATA(I3+1)
        DATA(I3)=DATA(J3)
        DATA(I3+1)=DATA(J3+1)
        DATA(J3)=TEMPR
125     DATA(J3+1)=TEMPI
130     M=NP2HF
140     IF(J-M)150,150,145
145     J=J-M
        M=M/2
        IF(M-NON2)150,140,140
150     J=J+M
```

```
C                                                                       0188
C     MAIN LOOP FOR FACTORS OF TWO.  PERFORM FOURIER TRANSFORMS OF      0189
C     LENGTH FOUR, WITH ONE OF LENGTH TWO IF NEEDED.  THE TWIDDLE       0190
C     W=EXP(ISIGN*2*PI*SQRT(-1)*M/(4*MMAX)).  CHECK FOR W=ISIGN*SQ      0191
C     AND REPEAT FOR W=ISIGN*SQRT(-1)*CONJUGATE(W).                     0192
C                                                                       0193
      NON2T=NON2+NON2                                                   0194
      IPAR=NTWO/NP1                                                     0195
310   IF(IPAR-2)350,330,320                                             0196
320   IPAR=IPAR/4                                                       0197
      GO TO 310                                                         0198
330   DO 340 I1=1,I1RNG,2                                               0199
      DO 340 J3=I1,NON2,NP1                                             0200
      DO 340 K1=J3,NTOT,NON2T                                           0201
      K2=K1+NON2                                                        0202
      TEMPR=DATA(K2)                                                    0203
      TEMPI=DATA(K2+1)                                                  0204
      DATA(K2)=DATA(K1)-TEMPR                                           0205
      DATA(K2+1)=DATA(K1+1)-TEMPI                                       0206
      DATA(K1)=DATA(K1)+TEMPR                                           0207
340   DATA(K1+1)=DATA(K1+1)+TEMPI                                       0208
350   MMAX=NON2                                                         0209
360   IF(MMAX-NP2HF)370,600,600                                         0210
370   LMAX=MAX0(NON2T,MMAX/2)                                           0211
      IF(MMAX-NON2)405,405,380                                          0212
380   THETA=-TWOPI*FLOAT(NON2)/FLOAT(4*MMAX)                            0213
      IF(ISIGN)400,390,390                                              0214
390   THETA=-THETA                                                      0215
400   WR=COS(THETA)                                                     0216
      WI=SIN(THETA)                                                     0217
      WSTPR=-2.*WI*WI                                                   0218
      WSTPI=2.*WR*WI                                                    0219
405   DO 570 L=NON2,LMAX,NON2T                                          0220
      M=L                                                               0221
      IF(MMAX-NON2)420,420,410                                          0222
410   W2R=WR*WR-WI*WI                                                   0223
      W2I=2.*WR*WI                                                      0224
      W3R=W2R*WR-W2I*WI                                                 0225
      W3I=W2R*WI+W2I*WR                                                 0226
420   DO 530 I1=1,I1RNG,2                                               0227
      DO 530 J3=I1,NON2,NP1                                             0228
      KMIN=J3+IPAR*M                                                    0229
      IF(MMAX-NON2)430,430,440                                          0230
430   KMIN=J3                                                           0231
440   KDIF=IPAR*MMAX                                                    0232
450   KSTEP=4*KDIF                                                      0233
      DO 520 K1=KMIN,NTOT,KSTEP                                         0234
      K2=K1+KDIF                                                        0235
      K3=K2+KDIF                                                        0236
      K4=K3+KDIF                                                        0237
      IF(MMAX-NON2)460,460,480                                          0238
```

```
460   U1R=DATA(K1)+DATA(K2)                                          0239
      U1I=DATA(K1+1)+DATA(K2+1)                                      0240
      U2R=DATA(K3)+DATA(K4)                                          0241
      U2I=DATA(K3+1)+DATA(K4+1)                                      0242
      U3R=DATA(K1)-DATA(K2)                                          0243
      U3I=DATA(K1+1)-DATA(K2+1)                                      0244
      IF(ISIGN)470,475,475                                           0245
470   U4R=DATA(K3+1)-DATA(K4+1)                                      0246
      U4I=DATA(K4)-DATA(K3)                                          0247
      GO TO 510                                                      0248
475   U4R=DATA(K4+1)-DATA(K3+1)                                      0249
      U4I=DATA(K3)-DATA(K4)                                          0250
      GO TO 510                                                      0251
480   T2R=W2R*DATA(K2)-W2I*DATA(K2+1)                                0252
      T2I=W2R*DATA(K2+1)+W2I*DATA(K2)                                0253
      T3R=WR*DATA(K3)-WI*DATA(K3+1)                                  0254
      T3I=WR*DATA(K3+1)+WI*DATA(K3)                                  0255
      T4R=W3R*DATA(K4)-W3I*DATA(K4+1)                                0256
      T4I=W3R*DATA(K4+1)+W3I*DATA(K4)                                0257
      U1R=DATA(K1)+T2R                                               0258
      U1I=DATA(K1+1)+T2I                                             0259
      U2R=T3R+T4R                                                    0260
      U2I=T3I+T4I                                                    0261
      U3R=DATA(K1)-T2R                                               0262
      U3I=DATA(K1+1)-T2I                                             0263
      IF(ISIGN)490,500,500                                           0264
490   U4R=T3I-T4I                                                    0265
      U4I=T4R-T3R                                                    0266
      GO TO 510                                                      0267
500   U4R=T4I-T3I                                                    0268
      U4I=T3R-T4R                                                    0269
510   DATA(K1)=U1R+U2R                                               0270
      DATA(K1+1)=U1I+U2I                                             0271
      DATA(K2)=U3R+U4R                                               0272
      DATA(K2+1)=U3I+U4I                                             0273
      DATA(K3)=U1R-U2R                                               0274
      DATA(K3+1)=U1I-U2I                                             0275
      DATA(K4)=U3R-U4R                                               0276
520   DATA(K4+1)=U3I-U4I                                             0277
      KMIN=4*(KMIN-J3)+J3                                            0278
      KDIF=KSTEP                                                     0279
      IF(KDIF-NP2)450,530,530                                        0280
530   CONTINUE                                                       0281
      M=MMAX-M                                                       0282
      IF(ISIGN)540,550,550                                           0283
540   TEMPR=WR                                                       0284
      WR=-WI                                                         0285
      WI=-TEMPR                                                      0286
      GO TO 560                                                      0287
550   TEMPR=WR                                                       0288
      WR=WI                                                          0289
```

```
      WI=TEMPP                                                          0290
560   IF(M-LMAX)565,565,410                                             0291
565   TEMPR=WR                                                          0292
      WR=WR*WSTPR-WI*WSTPI+WR                                           0293
570   WI=WI*WSTPR+TEMPR*WSTPI+WI                                        0294
      IPAR=3-IPAR                                                       0295
      MMAX=MMAX+MMAX                                                    0296
      GO TO 360                                                         0297
C                                                                       0298
C     MAIN LOOP FOR FACTORS NOT EQUAL TO TWO.  APPLY THE TWIDDLE F      0299
C     W=EXP(ISIGN*2*PI*SQRT(-1)*(J2-1)*(J1-J2)/(NP2*IFP1)), THEN        0300
C     PERFORM A FOURIER TRANSFORM OF LENGTH IFACT(IF), MAKING USE       0301
C     CONJUGATE SYMMETRIES.                                             0302
C                                                                       0303
600   IF(NTWO-NP2)605,700,700                                           0304
605   IFP1=NON2                                                         0305
      IF=1                                                              0306
      NP1HF=NP1/2                                                       0307
610   IFP2=IFP1/IFACT(IF)                                               0308
      J1RNG=NP2                                                         0309
      IF(ICASE-3)612,611,612                                            0310
611   J1RNG=(NP2+IFP1)/2                                                0311
      J2STP=NP2/IFACT(IF)                                               0312
      J1RG2=(J2STP+IFP2)/2                                              0313
612   J2MIN=1+IFP2                                                      0314
      IF(IFP1-NP2)615,640,640                                           0315
615   DO 635 J2=J2MIN,IFP1,IFP2                                         0316
      THETA=-TWOPI*FLOAT(J2-1)/FLOAT(NP2)                               0317
      IF(ISIGN)625,620,620                                              0318
620   THETA=-THETA                                                      0319
625   SINTH=SIN(THETA/2.)                                               0320
      WSTPR=-2.*SINTH*SINTH                                             0321
      WSTPI=SIN(THETA)                                                  0322
      WR=WSTPR+1.                                                       0323
      WI=WSTPI                                                          0324
      J1MIN=J2+IFP1                                                     0325
      DO 635 J1=J1MIN,J1RNG,IFP1                                        0326
      I1MAX=J1+I1RNG-2                                                  0327
      DO 630 I1=J1,I1MAX,2                                              0328
      DO 630 I3=I1,NTOT,NP2                                             0329
      J3MAX=I3+IFP2-NP1                                                 0330
      DO 630 J3=I3,J3MAX,NP1                                            0331
      TEMPR=DATA(J3)                                                    0332
      DATA(J3)=DATA(J3)*WR-DATA(J3+1)*WI                                0333
630   DATA(J3+1)=TEMPR*WI+DATA(J3+1)*WR                                 0334
      TEMPR=WR                                                          0335
      WR=WR*WSTPR-WI*WSTPI+WR                                           0336
635   WI=TEMPR*WSTPI+WI*WSTPR+WI                                        0337
640   THETA=-TWOPI/FLOAT(IFACT(IF))                                     0338
      IF(ISIGN)650,645,645                                              0339
645   THETA=-THETA                                                      0340
```

```
650   SINTH=SIN(THETA/2.)                                              0341
      WSTPR=-2.*SINTH*SINTH                                            0342
      WSTPI=SIN(THETA)                                                 0343
      KSTEP=2*N/IFACT(IF)                                              0344
      KRANG=KSTEP*(IFACT(IF)/2)+1                                      0345
      DO 698 I1=1,I1RNG,2                                              0346
      DO 698 I3=I1,NTOT,NP2                                            0347
      DO 690 KMIN=1,KRANG,KSTEP                                        0348
      J1MAX=I3+J1RNG-IFP1                                              0349
      DO 680 J1=I3,J1MAX,IFP1                                          0350
      J3MAX=J1+IFP2-NP1                                                0351
      DO 680 J3=J1,J3MAX,NP1                                           0352
      J2MAX=J3+IFP1-IFP2                                               0353
      K=KMIN+(J3-J1+(J1-I3)/IFACT(IF))/NP1HF                           0354
      IF(KMIN-1)655,655,665                                            0355
655   SUMR=0.                                                          0356
      SUMI=0.                                                          0357
      DO 660 J2=J3,J2MAX,IFP2                                          0358
      SUMR=SUMR+DATA(J2)                                               0359
660   SUMI=SUMI+DATA(J2+1)                                             0360
      WORK(K)=SUMR                                                     0361
      WORK(K+1)=SUMI                                                   0362
      GO TO 680                                                        0363
665   KCONJ=K+2*(N-KMIN+1)                                             0364
      J2=J2MAX                                                         0365
      SUMR=DATA(J2)                                                    0366
      SUMI=DATA(J2+1)                                                  0367
      OLDSR=0.                                                         0368
      OLDSI=0.                                                         0369
      J2=J2-IFP2                                                       0370
670   TEMPR=SUMR                                                       0371
      TEMPI=SUMI                                                       0372
      SUMR=TWOWR*SUMR-OLDSR+DATA(J2)                                   0373
      SUMI=TWOWR*SUMI-OLDSI+DATA(J2+1)                                 0374
      OLDSR=TEMPR                                                      0375
      OLDSI=TEMPI                                                      0376
      J2=J2-IFP2                                                       0377
      IF(J2-J3)675,675,670                                             0378
675   TEMPR=WR*SUMR-OLDSR+DATA(J2)                                     0379
      TEMPI=WI*SUMI                                                    0380
      WORK(K)=TEMPR-TEMPI                                              0381
      WORK(KCONJ)=TEMPR+TEMPI                                          0382
      TEMPR=WR*SUMI-OLDSI+DATA(J2+1)                                   0383
      TEMPI=WI*SUMR                                                    0384
      WORK(K+1)=TEMPR+TEMPI                                            0385
      WORK(KCONJ+1)=TEMPR-TEMPI                                        0386
680   CONTINUE                                                         0387
      IF(KMIN-1)685,685,686                                            0388
685   WR=WSTPR+1.                                                      0389
      WI=WSTPI                                                         0390
      GO TO 690                                                        0391
```

```
686   TEMPR=WR                                                          0392
      WR=WR*WSTPR-WI*WSTPI+WR                                           0393
      WI=TEMPR*WSTPI+WI*WSTPR+WI                                        0394
690   TWOWR=WR+WR                                                       0395
      IF(ICASE-3)692,691,692                                            0396
691   IF(IFP1-NP2)695,692,692                                           0397
692   K=1                                                               0398
      I2MAX=I3+NP2-NP1                                                  0399
      DO 693 I2=I3,I2MAX,NP1                                            0400
      DATA(I2)=WORK(K)                                                  0401
      DATA(I2+1)=WORK(K+1)                                              0402
693   K=K+2                                                             0403
      GO TO 698                                                         0404
C                                                                       0405
C     COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N ODD, BY CO      0406
C     JUGATE SYMMETRIES AT EACH STAGE.                                  0407
C                                                                       0408
695   J3MAX=I3+IFP2-NP1                                                 0409
      DO 697 J3=I3,J3MAX,NP1                                            0410
      J2MAX=J3+NP2-J2STP                                                0411
      DO 697 J2=J3,J2MAX,J2STP                                          0412
      J1MAX=J2+J1RG2-IFP2                                               0413
      J1CNJ=J3+J2MAX+J2STP-J2                                           0414
      DO 697 J1=J2,J1MAX,IFP2                                           0415
      K=1+J1-I3                                                         0416
      DATA(J1)=WORK(K)                                                  0417
      DATA(J1+1)=WORK(K+1)                                              0418
      IF(J1-J2)697,697,696                                              0419
696   DATA(J1CNJ)=WORK(K)                                               0420
      DATA(J1CNJ+1)=-WORK(K+1)                                          0421
697   J1CNJ=J1CNJ-IFP2                                                  0422
698   CONTINUE                                                          0423
      IF=IF+1                                                           0424
      IFP1=IFP2                                                         0425
      IF(IFP1-NP1)700,700,610                                           0426
C                                                                       0427
C     COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N EVEN, BY C      0428
C     JUGATE SYMMETRIES.                                                0429
C                                                                       0430
700   GO TO (900,800,900,701),ICASE                                     0431
701   NHALF=N                                                           0432
      N=N+N                                                             0433
      THETA=-TWOPI/FLOAT(N)                                             0434
      IF(ISIGN)703,702,702                                              0435
702   THETA=-THETA                                                      0436
703   SINTH=SIN(THETA/2.)                                               0437
      WSTPR=-2.*SINTH*SINTH                                             0438
      WSTPI=SIN(THETA)                                                  0439
      WR=WSTPR+1.                                                       0440
      WI=WSTPI                                                          0441
      IMIN=3                                                            0442
```

```
      JMIN=2*NHALF-1                                                    0443
      GO TO 725                                                         0444
710   J=JMIN                                                            0445
      DO 720 I=IMIN,NTOT,NP2                                            0446
      SUMR=(DATA(I)+DATA(J))/2.                                         0447
      SUMI=(DATA(I+1)+DATA(J+1))/2.                                     0448
      DIFR=(DATA(I)-DATA(J))/2.                                         0449
      DIFI=(DATA(I+1)-DATA(J+1))/2.                                     0450
      TEMPR=WR*SUMI+WI*DIFR                                             0451
      TEMPI=WI*SUMI-WR*DIFR                                             0452
      DATA(I)=SUMR+TEMPR                                                0453
      DATA(I+1)=DIFI+TEMPI                                              0454
      DATA(J)=SUMR-TEMPR                                                0455
      DATA(J+1)=-DIFI+TEMPI                                             0456
720   J=J+NP2                                                           0457
      IMIN=IMIN+2                                                       0458
      JMIN=JMIN-2                                                       0459
      TEMPR=WR                                                          0460
      WR=WR*WSTPR-WI*WSTPI+WR                                           0461
      WI=TEMPR*WSTPI+WI*WSTPR+WI                                        0462
725   IF(IMIN-JMIN)710,730,740                                          0463
730   IF(ISIGN)731,740,740                                              0464
731   DO 735 I=IMIN,NTOT,NP2                                            0465
735   DATA(I+1)=-DATA(I+1)                                              0466
740   NP2=NP2+NP2                                                       0467
      NTOT=NTOT+NTOT                                                    0468
      J=NTOT+1                                                          0469
      IMAX=NTOT/2+1                                                     0470
745   IMIN=IMAX-2*NHALF                                                 0471
      I=IMIN                                                            0472
      GO TO 755                                                         0473
750   DATA(J)=DATA(I)                                                   0474
      DATA(J+1)=-DATA(I+1)                                              0475
755   I=I+2                                                             0476
      J=J-2                                                             0477
      IF(I-IMAX)750,760,760                                             0478
760   DATA(J)=DATA(IMIN)-DATA(IMIN+1)                                   0479
      DATA(J+1)=0.                                                      0480
      IF(I-J)770,780,780                                                0481
765   DATA(J)=DATA(I)                                                   0482
      DATA(J+1)=DATA(I+1)                                               0483
770   I=I-2                                                             0484
      J=J-2                                                             0485
      IF(I-IMIN)775,775,765                                             0486
775   DATA(J)=DATA(IMIN)+DATA(IMIN+1)                                   0487
      DATA(J+1)=0.                                                      0488
      IMAX=IMIN                                                         0489
      GO TO 745                                                         0490
780   DATA(1)=DATA(1)+DATA(2)                                           0491
      DATA(2)=0.                                                        0492
      GO TO 900                                                         0493
```

APPENDIX H

```
C                                                                       0494
C     COMPLETE A REAL TRANSFORM FOR THE 2ND OR 3RD DIMENSION BY         0495
C     CONJUGATE SYMMETRIES.                                             0496
C                                                                       0497
800   IF(I1RNG-NP1)805,900,900                                          0498
805   DO 860 I3=1,NTOT,NP2                                             0499
      I2MAX=I3+NP2-NP1                                                  0500
      DO 860 I2=I3,I2MAX,NP1                                            0501
      IMIN=I2+I1RNG                                                     0502
      IMAX=I2+NP1-2                                                     0503
      JMAX=2*I3+NP1-IMIN                                                0504
      IF(I2-I3)820,820,810                                              0505
810   JMAX=JMAX+NP2                                                     0506
820   IF(IDIM-2)850,850,830                                             0507
830   J=JMAX+NPO                                                        0508
      DO 840 I=IMIN,IMAX,2                                              0509
      DATA(I)=DATA(J)                                                   0510
      DATA(I+1)=-DATA(J+1)                                              0511
840   J=J-2                                                             0512
850   J=JMAX                                                            0513
      DO 860 I=IMIN,IMAX,NPO                                            0514
      DATA(I)=DATA(J)                                                   0515
      DATA(I+1)=-DATA(J+1)                                              0516
860   J=J-NPO                                                           0517
C                                                                       0518
C     END OF LOOP ON EACH DIMENSION                                     0519
C                                                                       0520
900   NPO=NP1                                                           0521
      NP1=NP2                                                           0522
910   NPREV=N                                                           0523
920   RETURN                                                            0524
      END                                                               0525
      OVERLAY(PATS,1,0)
      PROGRAM READIN
     COMMON/BLK1/STARTT,ITFMT,NBLK,IPOW2,NCH,NPRINT,IPLOTA,IPLOTC,OFFSC
    1AL,DELTAT,SN,NRSKIP,LAP,NCROSS,ICROSS(2,20),NCHP,YLABEL(2),IWINDOW
    1,F1,F2,ITYPESP,NCON,NFSKIP,IFF,LAG1,LAG2
     COMMON/BLK2/ICH(14),CHSUM(14),NOFF(14),CHSUMSQ(14),SIGMA(14),RMS(1
    14),MEAN(14),SCALFAC(14),CHSUM1(14),TRACK(14),ICHAN(14),IFILTER(14)
     COMMON/BLK3/NPT,TMAX,NPTO2,NSPCT,DELF,N64,NPTO128
    1,INZERO,NREAD,NPTOT
     COMMON/BLK4/NMAX,NCHMAX,NBCMAX
     COMMON/BLK5/PCTC,NBINS,DMAX(14),DMIN(14),DBIN(14),BINS(100,14)
    1,CHISQC
     COMMON/BLK8/IAUTOSP(14),IAUTOCO(14),ICRSP(20),ICRCOR(20),ITRA(20),
    1ICOH(20)
     COMMON/BLK9/NFILTP,FREQF(50),WGHTF(50)
     NAMELIST/INPUT/ITFMT,NFSKIP,NRSKIP,SN,DELTAT,STARTT,OFFSCAL,NCH,
    1SCALFAC,NPTOT,NREAD,IAUTOSP,IAUTOCO,NCROSS,ICROSS,ICRSP,ICRCOR,
    2ITRA,ICOH,LAP,IWINDOW,ITYPESP,NPRINT,IPLOTA,IPLOTC,F1,F2,LAG1,LAG2
    3    ,PCTC,NBINS,DMAX,DMIN,INZERO,NFILTP,FREQF,WGHTF,IFILTER
```

APPENDIX H

```
      DATA NFILTP,FREQF,WGHTF,IFILTER/0,100*0.,14*0/
      DATA IAUTOSP,IALTOCO,ICRSP,ICRCUR,ITRA,ICOH/108*0/
      DATA IFF,LAG1,LAG2/1,0,0/
      DATA NFSKIP/0/
      DATA ITEMT,STARTT,ICH,NRSKIP,SCALFAC,OFFSCAL,LAP,NCROSS,ICROSS,
     1NPRINT,IPLOTA,IPLOTC,IWINDOW,F1,F2,ITYPESP/
     222,0.,15*0,14*1.,1.E+6,42*0,100,1,0,1,0.,20000.,2/
      DATA ID,CHAN/2HID,4HCHAN/
      DATA PCTC/90./
      DATA INZERO,NBINS,DMAX,DMIN/2*0,26*0./
C
C     READ AND PRINT NAMELIST INPUT
C
      READ INPUT
      IF(EOF,5) 1,2
    1 STOP
C
C     READ AND PRINT FORMATTED CARD INPUT
C
    2 PRINT INPUT
      READ 5002, YLABEL
      READ 5002, (TRACK(I),I=1,NCH)
 5002 FORMAT(8A10)
      PRINT 5001, YLABEL
      PRINT 5003, (I,TRACK(I),I=1,NCH)
 5001 FORMAT(///* CASE ID*5X1H*2A10,1H*)
 5003 FORMAT(///* CHANNEL     ID*/3X*NO.*/(I5,4XA10))
      PRINT 5004
 5004 FORMAT(*1*)
C
C     CHECK INPUT DATA, PRINT ERROR MESSAGES
C
      NPT=NREAD
      NBLK=NPTOT/NPT
      IF(NCH.LE.NCHMAX) GO TO 101
      PRINT 103, NCHMAX
      STOP 103
  103 FORMAT(///* NCH GT*I3,*, PROGRAM WILL NOT READ TAPE CORRECTLY. JOB
     1 TERMINATED*)
  101 CONTINUE
      IF(NFSKIP) 310,310,311
  311 DO 312 IFSKIP=1,NFSKIP
  313 READ(1) SKIPREC
      IF(EOF,1) 312,313
  312 CONTINUE $ PRINT 904, NFSKIP
  904 FORMAT(//6H * * *,I5,20H FILES SKIPPED * * *)
  310 IF(ITYPESP.LT.3) GO TO 90
      NCROSS=0
      DO 91 I=1,NCH
      IF(IAUTOCO(I).GT.0) IAUTOCO(I)=0
   91 CONTINUE
```

```
      PRINT 92
 92 FORMAT(//* AMPLITUDE SPECTRUM OPTION CHOSEN*/* ONLY AUTO SPECTRUM
     1WILL BE CALCULATED*)
 90 CONTINUE
      ICORR=0
      DO 116 I=1,NCH
      IF(IAUTOCO(I).EQ.0) GO TO 116
      ICORR=1
      IF(IAUTOSP(I).EQ.0) IAUTOSP(I)=-1
116 CONTINUE
      IF(NCROSS) 118,118,108
108 DO 110 I=1,NCROSS
      J1=ICROSS(1,I) $ J2=ICROSS(2,I)
      IF(IAUTOSP(J1).EQ.0) IAUTOSP(J1)=-1
      IF(IAUTOSP(J2).EQ.0) IAUTOSP(J2)=-1
      IF(ICRCOR(I).NE.0) GO TO 117
      IF(ITRA(I).NE.0) GO TO 211
      IF(ICOH(I).NE.0) GO TO 211
      GO TO 110
117 ICORR=1
211 IF(ICRSP(I).EQ.0) ICRSP(I)=-1
110 CONTINUE
118 IF(ICORR.EQ.1.AND.INZERO.EQ.0) PRINT 907
907 FORMAT(//107H * * * YOU MAY HAVE CIRCULAR ERROR IN YOUR CORRELATIO
    1NS BECAUSE YOU HAVE NOT ASKED FOR ZERO INSERTION * * *)
      IF(INZERO.EQ.0) GO TO 109
      NPT=2*NPT
      PRINT 901, NPT
901 FORMAT(//* ZERO INSERTION INCLUDED FOR ALL CHANNELS. BLOCK SIZE IS
    1*I7)
109 DO 35 IK=1,NCH
      NOFF(IK)=0
      CHSUM(IK)=CHSUMSQ(IK)=0.
      CHSUM1(IK)=0.
 35 CONTINUE
      IPOW2=0 $ NTEMP=NPT
203 IF(NTEMP-1) 200,200,202
202 IPOW2=IPOW2+1 $ NTEMP=NTEMP/2 $ GO TO 203
200 IF(2**IPOW2.NE.NPT) GO TO 201
      PRINT 207
207 FORMAT(//* BLOCK SIZE IS A POWER OF TWO. FAST FOURIER TRANSFORM WI
    1LL BE USED*)
      IF(NPT.LE.NMAX) GO TO 205
      PRINT 206 $ STOP 207
201 PRINT 204
204 FORMAT(//* BLOCK SIZE NOT A POWER OF TWO. SLOW FOURIER TRANSFORM W
    1ILL BE USED*)
      IF(NPT.LE.NMAX) GO TO 205
      PRINT 206
206 FORMAT(//* BLOCK SIZE TOO LARGE FOR DIMENSIONS PROVIDED. JOB TERMI
    1NATED*)
```

```
      STOP 206
  205 CONTINUE
      NPT02=NPT/2.
      NSPCT=NPLOT=NPTC2 $ IF(NPRINT.GT.NPT02) NPRINT=NPT02
      TMAX=NPT*DELTAT
      DELF=1./TMAX
      NPT0128=NPT/128
      IF(INZERO.EQ.0.OR.LAP.EQ.0) GO TO 111
      PRINT 902
      LAP=0
  902 FORMAT(///* NO 50 PERCENT OVERLAP ON ZERO INSERTION RUNS*/* INPUT
     1DATA ALTERED, LAP=0 *)
  111 CONTINUE
      IF(LAP.NE.0) NBLK=2*NBLK-1
      NCHP=0
      DO 80 I=1,NCH
      IF(IAUTOSP(I)) 81,80,81
   81 NCHP=NCHP+1
      ICHAN(NCHP)=I
   80 CONTINUE
      IF(NCHP) 82,82,83
   82 PRINT 84
   84 FORMAT(///* INPUT INDICATES NO CHANNELS TO BE PROCESSED, CASE ENDED
     1*)
      STOP 101
   83 CONTINUE
      IF(NCHP*NBLK.LE.NBCMAX) GO TO 306
      PRINT 307, NCHP,NBLK,NBCMAX
  307 FORMAT(///* NO. OF CHANNELS TO BE PROCESSED (NCHP) =*I5/* NO. OF BL
     1OCKS (NBLK) =*I5//33H NCHP * NBLK GREATER THAN NBCMAX=,I5//* EXECU
     2TION ENDED.  CHANGE DIMENSIONS TO FIT YOUR CASE AND RERUN*)
      STOP 07
  306 CONTINUE
      N64=NCHP*64
C
C     COMPUTE ACCURACY MEASUREMENT OF SPECTRAL ESTIMATORS
C
      IF(LAP.EQ.0) NDOF=2*NBLK $ IF(LAP.NE.0) NDOF=1.6364*(NBLK-1)
      CALL CSQ((100.+PCTC)/2.,NDOF,BU,BL,ICODE)
      BL=100./BL $ BU=100./BU
      PRINT 903, PCTC,BL,BU
  903 FORMAT(///58H    * * * ACCURACY MEASUREMENT OF SPECTRAL ESTIMATORS
     1* * *//* ASSUMING NORMALITY OF DATA, USER CAN BE*F5.0,*  CERTAIN*/
     2/* THAT THE SPECTRAL ESTIMATE IS WITHIN THE BOUNDS OF *//F5.0*  PE
     3RCENT AND*F5.0,*  PERCENT OF THE TRUE SMOOTHED SPECTRUM*)
C
C     COMPUTE CRITICAL VALUE OF CHISQUARE FOR NORMALITY TEST
C
      IF(NBINS.GT.100) NBINS=100
      IF(NBINS.EQ.0) GO TO 113
      DO 112 I=1,NCHP
```

```
      K=ICHAN(I)
      DBIN(I)=(DMAX(K)-DMIN(K))/NBINS
      DO 112 J=1,NBINS
  112 BINS(J,I)=0.
      NBINSM3=NBINS-3
      CALL CSQ(PCTC,NBINSM3,BL,BU,ICODE)
      CHISQC=NBINSM3/BL
  113 CONTINUE
      IF(NFILTP.EQ.0) GO TO 114
      IF(NFILTP.LE.50) GO TO 115
      PRINT 905
  905 FORMAT(///55H * * * INPUT ERROR, NFILTP GT 50. EXECUTION ENDED * *
     1 *)
      STOP 06
  115 PRINT 906
  906 FORMAT(//126H * * * SPECTRAL FILTERING OPTION SELECTED. CORRELATIO
     1NS WILL BE CALCULATED FROM FILTERED SPECTRA FOR CHANNELS SPECIFIED
     2. * * *)
  114 CONTINUE
C
C     COMPUTE DATA WINDOW CORRECTION FACTOR FOR SPECTRA
C
      CAPT=NREAD*DELTAT $ IWINDP1=IWINDOW+1
      GO TO (300,301,302,303),IWINDP1
  300 WCON=CAPT $ GO TO 304
  301 WCON=CAPT*.375 $ GO TO 304
  302 WCON=CAPT*.7136727029 $ GO TO 304
  303 WCON=CAPT*.2696428571
  304 CONTINUE
C
C     CORRECT PLOT LIMITS
C
      IF(F1.LT.0.) F1=0.
      FMAX=NPTO2*DELF
      IF(F2.GT.FMAX) F2=FMAX
      IF(LAG1.LT.-NSPCT) LAG1=-NSPCT
      IF(LAG2.GE.NSPCT) LAG2=NSPCT-1
      IF(LAG2.GE.LAG1) GO TO 305
      L1=LAG2 $ LAG2=LAG1 $ LAG1=L1
  305 CONTINUE
      RETURN
      END
      SUBROUTINE CSQ(P,N,BL,BU,ICODE)
      EXTERNAL FUNC
      COMMON/PROBF/A,Q
      A=N/2.
      Q=P/100.
C     FIND UPPER BOUND
      CALL ITR2(CHISQ,.001,1000.,1.,FUNC,1.E-6,1.E-6,100,ICODE)
      IF(ICODE-3) 1,1,2
    2 BU=1.E+12
```

```
      GO TO 3
    1 BU=N/CHISQ
    3 Q=1.-Q
C     FIND LOWER BOUND
      CALL ITP2(CHISQ,.001,1000.,1.,FUNC,1.E-6,1.E-6,100,ICODE)
      IF(ICODE-3) 4,4,5
    5 BL=0.
      GO TO 6
    4 BL=N/CHISQ
    6 RETURN
      END
      FUNCTION FUNC(CHISQ)
      COMMON/PROBF/A,Q
      DIMENSION H60(15)
      DATA H60/-.0118,-.0067,-.0033,-.0010,.0001,2*.0006,.0002,-.0003,
     1-.0006,-.0005,.0002,.0017,.0043,.0062/
      IF(A-15) 1,1,2
    1 X=CHISQ/2.
      FUNC=Q-GAMMF(A,X)/GAMMF(A,0.)
      GO TO 3
    2 X=((CHISQ/(2.*A))**(1./3.)-(1.-1./(9.*A)))/SQRT(1./(9.*A))
      IX=X/.5 $ IF(IX*.5.GT.X) IX=IX-1 $ IX=IX+8 $ IF(IX) 6,6,7
    7 IF(IX-15) 8,8,6
    8 IF(IX.LT.1) GO TO 6
      XREM=X-(IX-8)*.5
      IF(XREM) 9,9,10
    9 HNU=(30./A)*H60(IX)
      GO TO 11
   10 HNU=(30./A)*(H60(IX)+XREM*(H60(IX+1)-H60(IX))/.5)
      GO TO 11
    6 HNU=0.
   11 CONTINUE
      X=X+HNU
      AX=ABS(X)
      X2=AX*AX $ X3=AX*X2 $ X4=AX*X3
      PMINUS1=-.5*(1.+.196854*AX+.115194*X2+.000344*X3+.019527*X4)**(-4.
     1)
      IF(X) 4,5,5
    4 FUNC=Q-(1.+PMINUS1)
      GO TO 3
    5 FUNC=Q+PMINUS1
    3 RETURN
      END
      OVERLAY(PATS,2,0)
      PROGRAM BLOCKS
      COMMON CMAIN(1)
      COMMON/BLK1/STARTT,ITFMT,NBLK,IPOW2,NCH,NPRINT,IPLOTA,IPLOTC,OFFSC
     1AL,DELTAT,SN,NRSKIP,LAP,NCROSS,ICROSS(2,20),NCHP,YLABEL(2),IWINDOW
     1,F1,F2,ITYPESP,NCCN,NFSKIP,IFF,LAG1,LAG2
      COMMON/BLK2/ICH(14),CHSUM(14),NOFF(14),CHSUMSQ(14),SIGMA(14),RMS(1
     14),MEAN(14),SCALFAC(14),CHSUM1(14),TRACK(14),ICHAN(14)
```

APPENDIX H

```
      COMMON/BLK3/NPT,TMAX,NPT02,NSPCT,DELF,N64,NPT0128
     1,INZERO,NREAD
      COMMON/BLK5/PCTC,NBINS,DMAX(14),DMIN(14),DBIN(14),BINS(100,14)
     1,CHISQC
      COMMON/BLK6/ISAVE64,IRI,IXPLOT,IDATA,IZ,ISPECT
      COMMON/BLK7/II(11)
      COMMON/BLK8/IAUTOSP(14),IAUTOCO(14),ICRSP(20),ICRCOR(20),ITRA(20),
     1ICOH(20)
      DO 1 NB=1,NBLK
      CALL READTPE(NB,CMAIN(IDATA),CMAIN(ISAVE64))
      CALL TRAN(NB,CMAIN(IZ),CMAIN(ISAVE64),CMAIN(ISPECT))
    1 CONTINUE
      RETURN
      END
      SUBROUTINE READTPE(NB,DATA,SAVE64)
      DIMENSION DATA(1),SAVE64(1)
      COMMON/BLK1/START,ITEMT,NBLK,IFOW2,NCH,NPRINT,IPLOTA,IPLOTC,OFFSC
     1AL,DELTAT,SN,NRSKIP,LAP,NCROSS,ICROSS(2,20),NCHP,YLABEL(2),IWINDOW
     1,F1,F2,ITYPESP,KCON,NFSKIP,IFF,LAG1,LAG2
      COMMON/BLK2/ICH(14),CHSUM(14),NOFF(14),CHSUMSQ(14),SIGMA(14),RMS(1
     14),MEAN(14),SCALFAC(14),CHSUM1(14),TRACK(14),ICHAN(14)
      COMMON/BLK3/NPT,TMAX,NPT02,NSPCT,DELF,N64,NPT0128
     1,INZERO,NREAD
      COMMON/BLK7/NN,IWD,KCH,NFR,II,KK,N1,JJ,IREC,NREC,ILOC
      IF(ITEMT.EQ.3) GO TO 4000
      IF(NB-1) 12,12,11
   12 GO TO (3001,3010),ITEMT
C     ADTRAN FORMAT PARAMETERS SET UP
 3001 IF(NRSKIP) 105,105,106
  106 DO 107 I=1,NRSKIP
  107 READ(1)
  105 CONTINUE
      IF(NCH.GT.10) GO TO3002
      KCH=20
      NFR=25
      GO TO 3003
 3002 IF(NCH.GT.20) GO TO 3004
      KCH=30
      NFR=17
      GO TO 3003
 3004 IF(NCH.GT.30) GO TO 3005
      KCH=40
      NFR=12
      GO TO 3003
 3005 IF(NCH.GT.40) GO TO 3006
      KCH=50
      NFR=10
      GO TO 3003
 3006 IF(NCH.GT.100) GO TO 3007
      KCH=110
      NFR=4
```

```
      GO TO 3003
 3007 PRINT 3008
 3008 FORMAT(//* NCH GT 100 NOT ALLOWED*)
      STOP 02
 3003 IWD=9
      NN=KCH*NFR
      GO TO 3011
C     ADTRAN INTERFACE FORMAT ID RECORD
 3010 READ(1) KEY,NN,IWD,KCH,NFR,ID1,ID2,TSN
      READ(1)
      READ(1)
      IF(TSN.EQ.SN) GO TO 3011
      PRINT 3012
 3012 FORMAT(//* TAPE NOT POSITIONED AT ID RECORD FOR DESIRED SN*)
      STOP 3012
 3011 IF(NRSKIP) 3009,3009,108
  108 DO 109 I=1,NRSKIP
  109 READ(1)
C
C     FIND STARTING TIME ON TAPE
C
 3009 CONTINUE
      GO TO (3013,3014), ITEMT
 3013 READ(1) (DATA(I),I=1,NN)
      IF(EOF,1) 3,2
    2 IF(SN.EQ.DATA(2)) GO TO 3015
      PRINT 3016
 3016 FORMAT(//* TAPE NOT POSITIONED AT DESIRED SN*)
      STOP 04
 3014 READ(1) KEY,NN,(DATA(I),I=1,NN)
      IF(EOF,1) 3,3015
 3015 DO 1 J=1,NFR
      II=(J-1)*KCH+IWD+1
      JJ=J
      IF(DATA(II)-STARTT) 1,4,4
    1 CONTINUE
      GO TO 3009
    3 PRINT 900, STARTT
  900 FORMAT(//* STARTING TIME*E12.5,* NOT FOUND ON TAPE*)
      STOP
    4 KK=0
      N1=JJ
C
C     START READING DATA TO BE PROCESSED
C
      GO TO 13
   11 N1=JJ+1 $ KK=0
      IF(N1.GT.NFR) GO TO 15
      BACKSPACE 1
      GO TO 16
   15 N1=1 $ II=IWD+1
```

```
   16 IF(ITFMT.EQ.1) READ(1) (DATA(I),I=1,NN)
      IF(ITFMT.EQ.2) READ(1) KEY,NN,(DATA(I),I=1,NN)
      IF(EOF,1) 3063,3066
 3066 CONTINUE
      IF(LAP) 13,13,3062
  206 ILOC=I1-1
      IREC=IREC1-1
      GO TO 14
   13 IREC=0
   14 ILOC=0
   10 CONTINUE
      DO 5 J=N1,NFR
      KK=KK+1
      IF(KK-NREAD) 6,6,7
    6 ILOC=ILOC+1
      IJ=(ILOC-1)*NCHP
      DO 30 IKK=1,NCHP
      IK=ICHAN(IKK)
      X=DATA(II+IK)*SCALFAC(IK)
      IF(ABS( X)-OFFSCAL) 33,34,34
   33 CHSUM(IK)=CHSUM(IK)+X
      SAVE64(IJ+IKK)=X
      GO TO 30
   34 SAVE64(IJ+IKK)=CHSUM(IK)/(KK-1+(NB-1)*NREAD)
      CHSUM(IK)=CHSUM(IK)*(KK+(NB-1)*NREAD)/((KK+(NB-1)*NREAD)+1.)
      NOFF(IK)=NOFF(IK)+1
   30 CONTINUE
      IF(ILOC-64) 31,32,32
   32 ILOC=0 $ IREC=IREC+1
      CALL WRITMS(9,SAVE64,N64,IREC)
   31 CONTINUE
      JJ=J
      II=II+KCH
    5 CONTINUE
      IF(KK-NPT) 8,7,7
    8 CONTINUE
      IF(ITFMT.EQ.1) READ(1) (DATA(I),I=1,NN)
      IF(ITFMT.EQ.2) READ(1) KEY,NN,(DATA(I),I=1,NN)
      IF(EOF,1) 3063,3064
 3063 PRINT 3065
      STOP 3065
 3065 FORMAT(//* END OF FILE ENCOUNTERED BEFORE NPTOT POINTS READ, EXECU
     1TION ENDED*)
 3064 CONTINUE
      N1=1
      II=IWD+1
      GO TO 10
C
C     ONE BLOCK READ, READY TO BE PROCESSED BY FFT
C
    7 CONTINUE
```

```
      IF(ILOC.EQ.0) RETURN
      IREC=IREC+1
      CALL WRITMS(9,SAVE64,ILOC*NCHP,IREC)
      RETURN
C
C     RECIN FORMAT
C
 4000 NREC=NREAD/64 $ NCHP2=NCH+2 $ I1=1
      I=0
      IF(NB-1) 4001,4001,3029
 4001 IF(NRSKIP.EQ.0) GO TO 3050
      DO 3051 IRSKIP=1,NRSKIP
 3051 CALL RECIN(1,2,NN,DATA,1,NCHP2,1)
      PRINT 3052, NRSKIP
 3052 FORMAT(//I10,* RECORDS SKIPPED*)
 3050 CONTINUE
      I=I+1
      CALL RECIN(1,2,NN,DATA,1,NCHP2,1)
      IF(EOF,1) 3,3027
 3027 IF(DATA(2)-STARTT) 3050,3022,3022
 3022 IF(DATA(1).EQ.SN) GO TO 3035
      PRINT 3016
      STOP 05
 3035 I1=2
      PRINT 901, I
  901 FORMAT(//* STARTT FOUND AT RECORD*I5)
      KK=1
      DO 3023 IKK=1,NCHP
      IJ=ICHAN(IKK)
      XX=DATA(IJ+2)*SCALFAC(IJ)
      IF(ABS(XX)-OFFSCAL) 3030,3031,3031
 3031 XX=0. $ NOFF(IJ)=NOFF(IJ)+1
 3030 CHSUM(IJ)=CHSUM(IJ)+XX
 3023 SAVE64(IKK)=XX
      IREC1=1
      NREC=NREAD/64 $ GO TO 3060
 3029 IF(LAP) 3061,3061,3062
 3061 KK=0 $ I1=1 $ IREC1=1 $ NREC=NREAD/64 $ GO TO 3060
 3062 KK=0
      IWRITE=0
      IREC=NPTO2/64
      NREC=NREAD/64
      ILOC=(NPTO2-IREC*64)*NCHP
      IREC=IREC+1 $ NWORDS=N64 $ NLEFT=N64-ILOC
 4018 IF(IREC-NREC) 4016,4016,4017
 4017 NWORDS=(NREAD-NREC*64)*NCHP
      NLEFT=NWORDS-ILOC
 4016 CALL READMS(9,SAVE64,NWORDS,IREC)
      IREC=IREC+1
      KK=KK+NLEFT/NCHP
      IF(ILOC.EQ.0) GO TO 4014
```

```
      DO 4006 I=1,NLEFT
4006  SAVE64(I)=SAVE64(I+ILOC)
      IF(KK-NPTO2) 4007,4012,4013
4007  IF(IREC-NREC) 4009,4009,4010
4010  NWORDS=(NREAD-NREC*64)*NCHP
      IF(NWORDS.LT.ILOC) ILOC=NWORDS
4009  CALL READMS(9,SAVE64(NLEFT+1),ILOC,IREC)
      KK=KK+ILOC/NCHP
4014  IF(KK-NPTO2) 4011,4020,4021
4011  IWRITE=IWRITE+1
      CALL WRITIN(9,SAVE64,N64,IWRITE)
      GO TO 4018
4012  IF(NLEFT.LT.N64) GO TO 4019
4022  IWRITE=IWRITE+1
      CALL WRITIN(9,SAVE64,N64,IWRITE)
      I1=1
4015  IREC1=IWRITE+1
      KK=NPTO2
      GO TO 3060
4013  I1=NLEFT/NCHP-(KK-NPTO2)+1
      GO TO 4015
4019  I1=NLEFT/NCHP+1
      GO TO 4015
4020  IF(ILOC+NLEFT.EQ.N64) GO TO 4022
      I1=(ILOC+NLEFT)/NCHP+1
      GO TO 4015
4021  I1=(NLEFT+ILOC)/NCHP-(KK-NPTO2)+1
      GO TO 4015
3060  IF(ITFMT.LT.3) GO TO 206
      IF(IREC1.GT.NREC) GO TO 3046
      DO 3024 IREC=IREC1,NREC
      DO 3025 ILOC=I1,64
      CALL RECIN(1,2,NN,DATA,1,NCHP2,1)
      IF(EOF,1) 3063,3034
3034  KK=KK+1
      IJ=(ILOC-1)*NCHP
      DO 3026 IKK=1,NCHP
      IK=ICHAN(IKK)
      XX=DATA(IK+2)*SCALFAC(IK)
      IF(ABS(XX)-OFFSCAL) 3032,3033,3033
3033  XX=CHSUM(IK)/(KK-1+(NB-1)*NREAD)
      NOFF(IK)=NOFF(IK)+1
3032  CHSUM(IK)=CHSUM(IK)+XX
3026  SAVE64(IJ+IKK)=XX
3025  CONTINUE
      I1=1
3024  CALL WRITMS(9,SAVE64,N64,IREC)
      IF(NREC*64.EQ.NREAD) RETURN
3046  NLEFT=NREAD-64*NREC
      NREC=NREC+1
      DO 3042 ILOC=I1,NLEFT
```

```
      CALL RECIN(1,2,NN,DATA,1,NCHP2,1)
      IF(EOF,1) 3065,3041
 3041 KK=KK+1
      IJ=(ILOC-1)*NCHP
      DO 3042 IKK=1,NCHP
      IK=ICHAN(IKK)
      XX=DATA(IK+2)
      IF(XX-OFFSCAL) 3043,3044,3044
 3044 XX=CHSUM(IK)/(KK-1+(NB-1)*NREAD)
      NOFF(IK)=NOFF(IK)+1
 3043 CHSUM(IK)=CHSUM(IK)+XX
 3042 SAVE64(IJ+IKK)=XX
      CALL WRITMS(9,SAVE64,NLEFT*NCHP,NREC)
      RETURN
      END
      SUBROUTINE TRAN(NB,Z,SAVE64,SPECT)
      DIMENSION Z(1),SAVE64(1),SPECT(1)
      COMPLEX Z
      COMMON/BLK1/STARTT,ITEMT,NBLK,IPOW2,NCH,NPRINT,IPLOTA,IPLOTC,OFFSC
     1AL,DELTAT,SN,NRSKIP,LAP,NCROSS,ICROSS(2,20),NCHP,YLABEL(2),IWINDOW
     1,F1,F2,ITYPESP
      COMMON/BLK2/ICH(14),CHSUM(14),NOFF(14),CHSUMSQ(14),SIGMA(14),RMS(1
     14),MEAN(14),SCALFAC(14),CHSUM1(14),TRACK(14),ICHAN(14)
      COMMON/BLK3/NPT,TMAX,NPTO2,NSPCT,DELF,N64,NPTO128
     1,INZERO,NREAD
      COMMON/BLK5/PCTC,NBINS,DMAX(14),DMIN(14),DBIN(14),BINS(100,14)
      DO 19 JJ=1,NCHP
C
C     READ INPUT DATA BLOCK FROM RANDOM ACCESS FILE
C
      J=ICHAN(JJ)
      NREC=0
      IK=0
      IF(NREAD.LT.64) GO TO 21
      DO 20 I=64,NREAD,64
      NREC=NREC+1
      CALL READMS(9,SAVE64,N64,NREC)
      DO 20 IJ=1,64
      IK=I-64+IJ
      IL=(IJ-1)*NCHP+JJ
   20 Z(IK)=SAVE64(IL)
      IF(NREC*64.EQ.NREAD) GO TO 10
   21 NLEFT=NREAD-IK
      NREC=NREC+1
      CALL READMS(9,SAVE64,NLEFT*NCHP,NREC)
      DO 11 IJ=1,NLEFT
      IL=(IJ-1)*NCHP+JJ
   11 Z(IJ+IK)=SAVE64(IL)
   10 CONTINUE
C
C     COMPUTE BLOCK MEAN AND COUNTS FOR HISTOGRAM
```

```
C
      BLKMEAN=0.
      DO 22 I=1,NREAD
      CHSUMSQ(J)=CHSUMSQ(J)+Z(I)**2
   22 BLKMEAN=BLKMEAN+Z(I)
      CHSUM1(J)=BLKMEAN+CHSUM1(J)
      BLKMEAN=BLKMEAN/NREAD
      IF(NBINS.EQ.0) GO TO 40
      DMN=DMIN(J) $ DEL=DBIN(JJ)
      DO 41 I=1,NREAD
      IBIN=(Z(I)-DMN)/DEL+1
      IF(IBIN.LT.1) IBIN=1
      IF(IBIN.GT.NBINS) IBIN=NBINS
   41 BINS(IBIN,JJ)=BINS(IBIN,JJ)+1
   40 CONTINUE
      DO 23 I=1,NREAD
   23 Z(I)=Z(I)-BLKMEAN
C
C     APPLY DATA WINDOW
C
      IF(IWINDOW.EQ.0) GO TO 33
      GO TO (30,31,32),IWINDOW
   30 CALL HANNING(Z,NREAD)
      GO TO 33
   31 CALL HAMMING(Z,NREAD)
      GO TO 33
   32 CALL PARZEN(Z,NREAD)
   33 CONTINUE
C
C     INSERT ZEROS
C
      IF(INZERO.EQ.0) GO TO 24
      DO 35 I=1,NPTO2
   35 Z(I+NPTO2)=C.
C
C     COMPUTE FOURIER TRANSFORM AND STORE ON RANDOM ACCESS FILE
C
   24 CALL FOURT(Z,NPT,1,1,0,SPECT)
      IJ=(NB-1)*NCHP+JJ
      CALL WRITMS(8,Z,NPT,IJ)
   19 CONTINUE
      RETURN
      END
      SUBROUTINE HANNING(Z,NPT)
      COMPLEX Z(1)
      DATA PI/3.1415926535898/
      DO 1 I=1,NPT
      D=.5*(1.-COS(2.*PI*(I-1.)/NPT))
    1 Z(I)=Z(I)*D
      RETURN
      END
```

APPENDIX H

```
      SUBROUTINE HAMMING(Z,NPT)
      COMPLEX Z(1) $ DATA PI/3.1415926535898/
      PION=PI/NPT $ NP2=NPT+2
      DO 1 I=1,NPT
      T=(I+I-NP2)*PION
      D=.54+.46*COS(T)
    1 Z(I)=Z(I)*D
      RETURN
      END
      SUBROUTINE PARZEN(Z,NPT)
      COMPLEX Z(1)
      TM=NPT/2 $ TMM1=TM-1
      TMO2=TM/2.
      DO 1 I=1,NPT
      J=I-TMM1
      IJ=IABS(J)
      IF(IJ-TMO2) 2,2,3
    2 D=1.-6.*J*J*(1.-IJ)
      GO TO 1
    3 D=2.*(1.-IJ)**3
    1 Z(I)=Z(I)*D
      RETURN
      END
      OVERLAY(PATS,3,C)
      PROGRAM AUTOSP
      COMMON CMAIN(1)
      COMMON/BLK1/STARTT,ITEMT,NBLK,IPOW2,NCH,NPRINT,IPLOTA,IPLOTC,OFFSC
     1AL,DELTAT,SN,NRSKIP,LAP,NCROSS,ICROSS(2,20),NCHP,YLABEL(2),IWINDOW
     1,F1,F2,ITYPESP,WCON,NFSKIP,IFF,LAG1,LAG2
      COMMON/BLK2/ICH(14),CHSUM(14),NOFF(14),CHSUMSQ(14),SIGMA(14),RMS(1
     114),MEAN(14),SCALFAC(14),CHSUM1(14),TRACK(14),ICHAN(14)
      COMMON/BLK3/NPT,TMAX,NPTO2,NSPCT,DELF,No4,NPTO128
     1,INZERO,NREAD
      COMMON/BLK5/PCTC,NBINS,DMAX(14),DMIN(14),DBIN(14),BINS(100,14)
      COMMON/BLK6/ISAVE64,IRI,IXPLOT,IDATA,IZ,ISPECT
      COMMON/BLK8/IAUTOSP(14),IAUTOCO(14),ICRSP(20),ICRCOR(20),ITRA(20),
     1ICOH(20)
      COMMON/BLK9/NFILTP,FREQF(50),WGHTF(50)
C
C     COMPUTE AND PRINT SPECTRAL FILTER

      IF(NFILTP.LE.0) GO TO 1
      CALL SPLINE(FREQF,WGHTF,NFILTP,NSPCT,CMAIN(IRI),DELF)
      NREC=NBLK*NCHP+1
      CALL WRITMS(8,CMAIN(IRI),NSPCT,NREC)
      DO 2 I=1,NSPCT
    2 CMAIN(IXPLOT+I-1)=(I-1)*DELF
      I1=IXPLOT-1 $ I2=IRI-1
      PRINT 900, (I,CMAIN(I1+I),CMAIN(I2+I),I=1,NSPCT)
  900 FORMAT(//*1SPECTRAL FILTER WEIGHTING FUNCTION*//5X*I*3X*FREQUENCY*
     16X*WEIGHT*3(7X*I*3X*FREQUENCY*6X*WEIGHT*)/(I6,2E13.5,I6,2E13.5,I6,
```

```
     22E13.5,I6,2E13.5))
    1 CONTINUE
      CALL AUTO(CMAIN(IZ),CMAIN(ISPECT),CMAIN(IRI),CMAIN(IXPLOT))
C
C     CALL AUTO FUNCTION ROUTINE WITH COMPUTED BLOCK ADDRESSES
C
      IF(NBINS.GT.0) CALL NORMAL
      RETURN
      END
      SUBROUTINE AUTO(Z,SPECT,RI,XPLOT)
      DIMENSION SPECT(1),XPLOT(1),RI(1),Z(1)
      COMPLEX Z,SPECT
      COMMON/BLK1/STARTT,ITEMT,NBLK,IPOW2,NCH,NPRINT,IPLOTA,IPLOTC,OFFSC
     1AL,DELTAT,SN,NRSKIP,LAP,NCROSS,ICROSS(2,20),NCHP,YLABEL(2),IWINDOW
     1,F1,F2,ITYPESP,WCON,NFSKIP,IFF,LAG1,LAG2
      COMMON/BLK2/ICH(14),CHSUM(14),NOFF(14),CHSUMSQ(14),SIGMA(14),RMS(1
     14),MEAN(14),SCALFAC(14),CHSUM1(14),TRACK(14),ICHAN(14),IFILTER(14)
      COMMON/BLK3/NPT,TMAX,NPTO2,NSPCT,DELF,N64,NPTO128
     1,INZERO,NREAD
      COMMON/BLK8/IAUTOSP(14),IAUTOCO(14),ICRSP(20),ICRCOR(20),ITRA(20),
     1ICOH(20)
      COMMON/BLK9/NFILTP
      COMMON/BLK10/NRCRD/
      DIMENSION PLABEL(6),PPLOT(27),BAND(27),IDEN(5)
      REAL MEAN
      COMPLEX ZZ
      PRINT 912, (TRACK(J),NOFF(J),J=1,NCH)
  912 FORMAT(//* NO. OF OFFSCALE VALUES FOR EACH CHANNEL*/(1X,A10,T10))
C
C     COMPUTE MEAN AND VARIANCE OF EACH CHANNEL
C
      DO 36 IKK=1,NCHP
      IK=ICHAN(IKK)
      MEAN(IK)=CHSUM1(IK)/(NBLK*NREAD)
      SECMOM=CHSUMSQ(IK)/(NBLK*NREAD)
      SIGMASQ=(SECMOM-MEAN(IK)**2 )
      IF(SIGMASQ) 38,37,37
   38 PRINT 913, IK
  913 FORMAT(* NEGATIVE SIGMASQ FOR CH.*I5)
      SIGMA(IK)=0.
      GO TO 36
   37 SIGMA(IK)=SQRT(SIGMASQ)
   36 CONTINUE
C
C     START OF LOOP FOR COMPUTING AUTO SPECTRA AND CORRELATIONS
C
      DO 50 JJ=1,NCHP
      J=ICHAN(JJ)
      DO 53 I=1,NSPCT
   53 SPECT(I)=0.
C     AVERAGE NBLK SPECTRA FOR ONE CHANNEL
```

APPENDIX H

```
      DO 54 IBLK=1,NBLK
      IJ=(IBLK-1)*NCHP+JJ
      CALL READMS(8,Z,NPT,IJ)
      GO TO (80,80,81),ITYPESP
   80 DO 91 I=1,NSPCT
   91 SPECT(I)=SPECT(I)+(REAL(Z(I))**2+AIMAG(Z(I))**2)
      GO TO 54
   81 DO 82 I=1,NSPCT
      ZZ=Z(I)
      AMP=CABS(ZZ) $ ARG=0. $IF(AMP.NE.0.) ARG=ATAN2(AIMAG(ZZ),REAL(ZZ))
   82 SPECT(I)=SPECT(I)+CMPLX(AMP,ARG)
   54 CONTINUE
C
C     COMPUTE CORRECTED AUTO SPECTRUM AND CALCULATED VARIANCE
C
      SUM=0.
      GO TO (83,83,84),ITYPESP
   84 CON=2./(NBLK*NREAD)
      DO 85 I=1,NSPCT
      AMP=REAL(SPECT(I))*CON
      ARG=AIMAG(SPECT(I))*57.2957795/NBLK
      SPECT(I)=CMPLX(AMP,ARG)
   85 CONTINUE
      RMS(J)=0.
      GO TO 86
   83 CON=DELTAT*DELTAT/(6.28318308*WCON*NBLK)
      DO 59 I=1,NSPCT
      SPECT(I)=SPECT(I)*CON
   59 SUM=SUM+SPECT(I)
      RMS(J)=SQRT(SUM*DELF*12.56637062)
   86 CONTINUE
C     APPLY SPECTRAL FILTER
      IF(NFILTP.LE.0.OR.IFILTER(J).EQ.0) GO TO 58
      CALL READMS(8,RI,NSPCT,NBLK*NCHP+1)
      DO 112 I=1,NSPCT
  112 SPECT(I)=SPECT(I)*RI(I)
C
C     PRINT AUTO SPECTRUM AND SET UP PLOTTING ARRAYS
C
   58 PRINT 1001, J,TRACK(J),MEAN(J),SIGMA(J),RMS(J)
 1001 FORMAT(///*1CHANNEL*I3,3XA10,          3X*MEAN=*E12.5,3X*SIGMA=*
     1E12.5,3X*PMS=*E12.5)
 1002 FORMAT(///* AVERAGE OF*I3,* TRANSFORMS*//5X*I*3X*FREQUENCY*6X*POWE
     1R*3(3X*I*3X*FREQUENCY*6X*POWER*))
      IF(ITYPESP.EQ.3) GO TO 87
      DO 60 I=1,NSPCT
      XPLOT(I)=(I-1)*DELF
      AMP=SPECT(I)*2.
      SPECT(I)=RI(I)=AMP
   60 CONTINUE
  911 FORMAT(4(I6,2E13.5))
```

```
      CALL WRITMS(9,RI,NSPCT,JJ)
      IF(IAUTOSP(J).LE.0) GO TO 100
      GO TO (66,67,87),ITYPESP
   66 ENCODE(60,901,PLABEL) DELF,TRACK(J)
  901 FORMAT(*AUTO POWER SPECTRUM (BANDWIDTH=*E9.2,*)*9X,A10)
      NPLABEL=41
      DO 69 I=1,NSPCT
   69 RI(I)=RI(I)*DELF
      PRINT 1005,(PLABEL(I),I=1,5)
      GO TO 68
   67 ENCODE(60,902,PLABEL) TRACK(J)
  902 FORMAT(*AUTO POWER SPECTRAL DENSITY*3X,A10,20X)
      NPLABEL=27
      PRINT 1006,(PLABEL(I),I=1,3)
 1005 FORMAT(//20X7H * * * ,5A10,6H * * *)
 1006 FORMAT(//20X7H * * * ,3A10,6H * * *)
      GO TO 68
   87 DO 88 I=1,NSPCT
      XPLOT(I)=(I-1)*DELF
   88 RI(I)=SPECT(I)
      ENCODE(60,903,PLABEL) TRACK(J)
  903 FORMAT(* AMPLITUDE SPECTRUM AND PHASE*5X,A10,16X)
      PRINT 1009, PLABEL(1),PLABEL(2)
 1009 FORMAT(//20X7H * * * ,2A10,6H * * *)
      NPLABEL=18
      PRINT 1007, NBLK
 1007 FORMAT(///* AVERAGE OF*I5,* TRANSFORMS*//5X*I*3X*FREQUENCY*4X*AMPL
     1ITUDE*4X*PHASE*2( 9X*I*3X*FREQUENCY*4X*AMPLITUDE*4X*PHASE*))
      PRINT 1008, (I,XPLOT(I),SPECT(I),I=1,NPRINT)
 1008 FORMAT(I6,2E13.5,F8.2,I10,2E13.5,F8.2,I10,2E13.5,F8.2)
      GO TO 89
   68 CONTINUE
      PRINT 1002, NBLK
      PRINT 911, (I,XPLOT(I),RI(I),I=1,NPRINT)
   89 CONTINUE
      NRCRD7=NRCRD7+1
      WRITE(7) PLABEL,NSPCT,(XPLOT(I),RI(I),I=1,NSPCT)
      PRINT 1010, NRCRD7,PLABEL
 1010 FORMAT(//21H * * * * * RECORD NO.,I5, * ON TAPE7 CONTAINS *6A10,10
     1H * * * * *)
C     PLOT FANFOLD PLOTS AND COMPUTE 1/3 O.B. SPECTRUM
      GO TO (70,70,62,63,62,63),IPLOTA
   62 ILOG=0 $ GO TO 61
   63 ILOG=2
   61 CALL PLOTNB(YLABEL,TRACK(J),10,XPLOT,RI,NSPCT,ILOG,F1,F2,PLABEL,
     1NPLABEL,IFF,1)
      IF(ITYPESP.LT.3) GO TO 70
      ILOG=0
      DO 90 I=1,NSPCT
   90 RI(I)=AIMAG(SPECT(I))
      ENCODE(50,904,PLABEL)
```

```
  904 FORMAT(*PHASE ANGLE, DEGREES*30X)
      NPLABEL=20
      CALL PLOTNB(YLABEL,TRACK(J),10,XPLOT,RI,NSPCT ,ILOG,F1,F2,PLABEL,
     1NPLABEL,0,0)
      GO TO 100
   70 CONTINUE
      IF(ITYPESP.EQ.3) GO TO 100
C
C     COMPUTE 1/3 OCTAVE BAND SPECTRA
C
      GO TO (100,65,100,100,65,65),IPLOTA
   65 CALL BANDS(DELF,NSPCT,SPECT,PPLOT,BAND,NPP,IERR,ITYPESP)
      IF(NPP.LE.0) GO TO 100
      CALL PLOTB(YLABEL,TRACK(J),10,BAND,PPLOT,NPP,PLABEL,NPLABEL,IFF)
   64 CONTINUE
      GO TO (72,71),ITYPESP
   71 ENCODE(60,73,PLABEL) TRACK(J)
   73 FORMAT(*1/3 OB AUTO POWER SPECTRAL DENSITY*6X,A10,10X)
      GO TO 74
   72 ENCODE(60,75,PLABEL) TRACK(J)
   75 FORMAT(*1/3 OB AUTO POWER SPECTRUM*4X,A10,20X)
   74 CONTINUE
      NRCRD7=NRCRD7+1
      WRITE(7) PLABEL,NPP,(BAND(I),PPLOT(I),I=1,NPP)
      PRINT 1010, NRCRD7,PLABEL
  100 IF(IAUTOCO(J).LE.0) GO TO 50
C
C     COMPUTE AUTOCORRELATION
C
      DO 101 I=2,NSPCT
      Z(I)=SPECT(I)
  101 Z(I+NSPCT)=SPECT(NSPCT-I+2)
      Z(1)=Z(NSPCT+1)=SPECT(1)
      IF(ITYPESP.GT.1) GO TO 113
      DO 114 I=1,NPT
  114 Z(I)=Z(I)/DELF
  113 CONTINUE
      CALL FOURT(Z,NPT,1,-1,0,SPECT)
      CON=3.1415926535898/TMAX
      IF(INZERO) 1014,1014,1011
 1014 DO 110 I=1,NSPCT
  110 SPECT(I)=Z(I)*CON
      GO TO 1012
 1011 DO 1013 I=1,NSPCT
 1013 SPECT(I)=Z(I)*CON*NPT/(NPT-I-I+2.)
 1012 CONTINUE
C
C     PRINT AUTOCORRELATION
C
      DO 102 I=1,NPT
      IM1=I-1
```

```
      RI(I)=SPECT(I)
  102 XPLOT(I)=I*1
      PRINT 1003, TRACK(J),(XPLOT(I),RI(I),I=1,NSPCT)
 1003 FORMAT(//*1AUTO CORRELATION ,*A10//8(7X*I*0X*RX*)/(2X,F5.0,E11.3,
     1F5.0,E11.3,F5.0,E11.3,F5.0,E11.3,F5.0,E11.3,F5.0,E11.3,F5.0,E11.3,
     2F5.0,E11.3))
      ENCODE(60,1004,PLABEL) TRACK(J)
 1004 FORMAT(*AUTO CORRELATION  *,A10,32X)
      NRCRD7=NRCRD7+1
      WRITE(7) PLABEL,NSPCT,(XPLOT(I),RI(I),I=1,NSPCT)
      PRINT 1010, NRCRD7,PLABEL
      IF(LAG1.EQ.0.AND.LAG2.EQ.0) GO TO 50
C
C     SET UP PLOT ARRAYS AND PLOT AUTOCORRELATION
C
      L1=LAG1 $ IF(L1.LT.0) L1=0 $ L2=LAG2
      IF(LAG2.GT.0) GO TO 109
      L1=-LAG2 $ L2=-LAG1
  109 I1=L1+1 $ I2=L2+1
  106 IF(I1.GE.I2) GO TO 50
      NPLT=I2-I1+1
      DO 107 I=1,NPLT
      II=I+I1-1
      XPLOT(I)=II-1
  107 RI(I)=SPECT(II)
      CALL ASCALE(RI,10.,NPLT,1,10.)
      IF(IFF.EQ.0) GO TO 50
      IF(NPLT-256) 201,201,202
  201 K=1 $ GO TO 103
  202 K=NPLT/256
  103 ENCODE(50,920,IDEN) TRACK( J)
  920 FORMAT(*AUTOCORRELATION*3X,A10,22X)
      PN=RI(NPLT+1)
      PX=RI(NPLT+2)*10.+PN
      CALL FANFOLD(RI,NPLT,K,1,NPLT,IDEN,1H*,1.,PX,PN,YLABEL,2,120,0,0.,
     11H )
   50 CONTINUE
      RETURN
      END
      SUBROUTINE NORMAL
      COMMON/BLK1/STARTT,ITFMT,NBLK,IROW2,NCH,NPRINT,IPLOTA,IPLOTC,OFFSC
     1AL,DELTAT,SN,NRSKIP,LAP,NCROSS,ICROSS(2,20),NCHP,YLABEL(2),IWINDOW
     1,F1,F2,ITYPESP
      COMMON/BLK2/ICH(14),CHSUM(14),NOFF(14),CHSUMSQ(14),SIGMA(14),RMS(1
     14),MEAN(14),SCALFAC(14),CHSUM1(14),TRACK(14),ICHAN(14)
      COMMON/BLK3/NPT,TMAX,NPTO2,NSPCT,DELF,N64,NPTO128
     1,INZERO,NREAD,NPTOT
      COMMON/BLK5/PCTC,NBINS,CMAX(14),CMIN(14),DBIN(14),BINS(100,14)
     1,CHISQC
      REAL MU,MEAN
      DIMENSION YL(3),IDEN(5)
```

APPENDIX H

```
      DATA IDEN/10HCOUNTS     ,4*10H           /
      NBINSM3=NBINS-3
      DO 1 I=1,NCHP
      J=ICHAN(I)
      MU=MEAN(J)
      ENCODE(30,902,YL) TRACK(J)
  902 FORMAT(3X*HISTOGRAM FOR *A10,3X)
      PN=PX=0.
      CALL FANFOLD(BINS(1,I),NBINS,1,1,100,IDEN,1H*,1.,PN,PX,YL,3,
     1100,0,XARRAY,XLABEL)
      SIG=SIGMA(J)
      FACTOR=1./(2.5066282/5*SIG)
      DEL=DBIN(I)
      DMAX=0.
      SUM=0.
      IF(DMIN(J).LT.-20.) GO TO 6
      PN=PFUN(-20.,DMIN(J),MU,SIG)*FACTOR
      GO TO 7
    6 PN=0.
    7 CONTINUE
      PRINT 905, (BINS(K,I),K=1,NBINS)
  905 FORMAT(/* BINS*/(10F10.0))
      CHISQ=0.
      DO 2 K=1,NBINS
      A=K*DEL+DMIN(J)
      SUM=     BINS(K,I)
      P=SUM/NPTOT
      PN=    PFUN(A-DEL,A,MU,SIG)*FACTOR
      IF(PN.EQ.0) GO TO 2
      CHISQ=NPTOT*(P-PN)**2/PN+CHISQ
    2 CONTINUE
      ALPHA=1.-PCTC/100.
      PRINT 900, NBINSM3,CHISQ,ALPHA,CHISQC
  900 FORMAT(///12X33H * * * GOODNESS OF FIT TEST * * *//* DEGREES OF FR
     1EEDOM =*I5,10X*CHI-SQUARE =*F10.3//* AT THE SIGNIFICANCE LEVEL OF*
     2F10.3,10X*THE CRITICAL VALUE OF CHI-SQUARE IS*F10.3)
    1 CONTINUE
      RETURN
      END
      FUNCTION PFUN(A,B,MU,SIG)
      REAL MU
      PFUN=0.
      DX=(B-A)/50.
      X=A-DX/2.
      DO 1 I=1,50
      X=X+DX
      PFUN=PFUN+DX*EXP(-(X-MU)**2/(2.*SIG*SIG))
    1 CONTINUE
      RETURN
      END
      SUBROUTINE PLOTP(YLABEL,FRAMEL,NF,BAND,PPLOT,NPP,PLABEL,NP,IFF)
```

APPENDIX H

```
      DIMENSION BAND(1),PPLOT(1),YLABEL(2),FRAMEL(1),PLABEL(5),FFID(5)
     1,IDEN(6),BCF(24)
      DATA BCF/50.,63.,80.,100.,125.,160.,200.,250.,315.,400.,500.,630.,
     1800.,1000.,1250.,1600.,2000.,2500.,3150.,4000.,5000.,8000.,10000.,
     220000./
C
C     PLOT POWER FOR BAND CENTER FREQUENCIES 50-20K HZ
C
 4001 DO 2001 I=1,NPP
 2001 BAND(I)=I+4
      PMAX=PPLOT(1)
      DO 2002 I=2,NPP
 2002 PMAX=AMAX1(PPLOT(I),PMAX)
      NMAX=IFIX(PMAX)
      IF(NMAX.LT.PMAX) NMAX=NMAX+1
      PMAX=NMAX
      PMIN=PMAX-5.
      I2=NPP
      DO 2004 I=1,I2
      IF(PPLOT(I).LT.PMIN) PPLOT(I)=PMIN
 2004 CONTINUE
      PPLOT(NPP+1)=PMIN $ PPLOT(NPP+2)=.5
      BAND(NPP+1)=1. $ BAND(NPP+2)=5.
      IF(IFF.EQ.0) RETURN
      FFID(1)=YLABEL(1) $ FFID(2)=YLABEL(2)
      NWORDS=(NF+9)/10
      DO 1 I=1,NWORDS
    1 FFID(I+2)=FRAMEL(I)
      ENCODE(54,905,IDEN) PLABEL
  905 FORMAT(*LOG *5A10)
      CALL FANFOLD(PPLOT,NPP,1,1,NPP,IDEN,1H*,1.,PMAX,PMIN,FFID,NWORDS+2
     1,120,1,BCF,1CHFREQUENCY )
      RETURN
      END
      SUBROUTINE BANDS(DELF,NSPCT,SPECT,PPLOT,BAND,NPP,IERR,IPOWPLT)
      COMPLEX SPECT(1)
      DIMENSION PPLOT(1),IND(46,2),FNC(46),FNL(47)
      DATA FNL/1.1220,1.4125,1.7783,2.2387,2.8184,3.5481,4.4668,5.6234,
     17.0795,8.9125,11.220,14.125,17.783,22.387,28.184,35.481,44.668,
     256.234,70.795,89.125,112.20,141.25,177.83,223.87,281.84,354.81,
     3446.68,562.34,707.95,891.25,1122.0,1412.5,1778.3,2238.7,2818.4,
     43548.1,4466.8,5623.4,7079.5,8912.5,11220.,14125.,17765.,22387.,
     528184.,35481.,44668./
      DATA FNC/1.2589,1.5849,1.9953,2.5119,3.1623,3.9811,5.0119,6.3096,
     17.9466,10.,12.589,15.849,19.953,25.119,31.623,39.811,50.119,63.096
     2,79.433,100.,125.89,158.49,199.53,251.19,316.23,398.11,501.19,
     3630.96,794.33,1000.,1258.9,1584.9,1995.3,2511.9,3162.3,3981.1,
     45011.9,6309.6,7943.3,10000.,12589.,15849.,19953.,25119.,31623.,
     539811./
      DIMENSION PSD(46),PSDPHZ(46),BAND(27)
      DO 3 I=1,27
```

APPENDIX H

```
  3 BAND(I)=I
    IERR=0
    DO 1 I=1,46
    BND=FNL(I+1)-FNL(I)
    I1=I
    IF(BND.GE.DELF) GO TO 2
  1 CONTINUE
    IERR=1
    RETURN
  2 DO 62 I=1,46
    PSDPHZ(I)=PSD(I)=0.
 62 IND(I,1)=IND(I,2)=0
    II=I1
    DO 63 I=2,NSPCT
    F=(I-1)*DELF
 68 IF(II-46) 64,64,65
 65 II=46
    GO TO 66
 64 IF(F-FNL(II+1)) 66,67,67
 66 IF(IND(II,2).EQ.0) IND(II,1)=I
    IND(II,2)=IND(II,2)+1
    GO TO 63
 67 II=II+1
    GO TO 68
 63 CONTINUE
    NPP=0
    DO 76 I=I1,46
    IF(IND(I,2)) 76,76,70
 70 ISTRT=IND(I,1)
    NPP=NPP+1
    CALL BNDSUM(SPECT(ISTRT),IND(I,2),PSD(I))
    PSD(I)=PSD(I)*DELF
    PSDPHZ(I)=PSD(I)/(FNL(I+1)-FNL(I))
 76 CONTINUE
    NPP=NPP-(17-I1)
    IF(NPP.LT.0) NPP=0
    IF(NPP.GT.27) NPP=27
    PRINT 80, (FNC(I),PSD(I),PSDPHZ(I),I=I1,46)
 80 FORMAT(//*    1/3 OCTAVE BAND*5X*POWER*5X*POWER SPECTRAL*/*  CENTER
   1 FREQUENCY*4X*SPECTRUM*7X*DENSITY*/(F12.0,E19.4,E15.4))
    IF(NPP.EQ.0) GO TO 3004
    GO TO (3001,3002),IPOWPLT
3001 DO 3003 I=1,NPP
    PPLOT(I)=-100. $ IF(PSD(I+16).GT.0.) PPLOT(I)=ALOG10(PSD(I+16))
3003 CONTINUE
    GO TO 3004
3002 DO 3005 I=1,NPP
    PPLOT(I)=-100.
    IF(PSDPHZ(I+16).GT.0.) PPLOT(I)=ALOG10(PSDPHZ(I+16))
3005 CONTINUE
3004 RETURN
```

```
      END
      SUBROUTINE BNDSUM(S,NUM,PSD)
      COMPLEX S(1)
      PSD=0.
      IF(NUM.GT.1) GO TO 1
      PSD=REAL(S(1))
      RETURN
    1 PSD=REAL(S(1))+REAL(S(NUM))
      IF(NUM.EQ.2) RETURN
      NUMM1=NUM-1
      DO 2 I=2,NUMM1
    2 PSD=PSD+REAL(S(I))
      RETURN
      END
      SUBROUTINE SPLINE (X,Y,KNT,KNTOUT,YOUT,DX)
      DIMENSION C(4,4),A(4),IPIV(4),X(1),Y(1),YOUT(1)
      XM = 2.*X(1) - X(2)
      YM = 2.*Y(1) - Y(2)
      XN = X(1)
      YN = Y(1)
      XO = X(2)
      YO = Y(2)
      XP = X(3)
      YP = Y(3)
      SLN = (YN-YM)/(XN-XM)
      SLO = (YO-YN)/(XO-XN)
      SLP = (YP-YO)/(XP-XO)
      RKOUT = KNTOUT
      IYNX=X(1)/DX
      IF(IYNX*DX.LT.X(1)) IYNX=IYNX+1
      VAR=IYNX*DX
      DO 1 I=1,IYNX
    1 YOUT(I)=Y(1)
      IYNX=IYNX+1
      LIM = KNT + 1
      DO   7000   N=3,LIM
      IF (SLN .NE. SLO .OR. SLO .NE. SLP)  GO TO 3000
C     LINEAR
      A(4) = 0.
      A(3) = 0.
      A(2) = SLO
      A(1) = YN - SLO*XN
      GO TO 6000
 3000 CONTINUE
      A(1) = (SLO - SLN)/(XO - XM)
      A(2) = (SLP - SLO)/(XP - XN)
      IF (A(1) .NE. A(2))  GO TO 5000
C     PARABOLIC
      C(1,1) = 1.
      C(2,1) = 1.
      C(3,1) = 1.
```

```
      C(1,2) = XN
      C(2,2) = XO
      C(3,2) = XP
      C(1,3) = XN*XN
      C(2,3) = XO*XO
      C(3,3) = XP*XP
      A(1) = YN
      A(2) = YO
      A(3) = YP
      CALL SIMEQ (C, 3, A, 1, DET, IPIV, 4, ISC)
      A(4) = 0.
      GO TO 6000
 5000 CONTINUE
C   CUBIC
      C(1,1) = 1.
      C(2,1) = 1.
      C(3,1) = 0.
      C(4,1) = 0.
      C(1,2) = XN
      C(2,2) = XO
      C(3,2) = 1.
      C(4,2) = 1.
      C(1,3) = XN*XN
      C(2,3) = XO*XO
      C(3,3) = 2.*XN
      C(4,3) = 2.*XO
      C(1,4) = XN*C(1,3)
      C(2,4) = XO*C(2,3)
      C(3,4) = 3.*C(1,3)
      C(4,4) = 3.*C(2,3)
      A(1) = YN
      A(2) = YO
      A(3) = TAN(.5*(ATAN(SLN) + ATAN(SLO)))
      A(4) = TAN(.5*(ATAN(SLO) + ATAN(SLP)))
      CALL SIMEQ (C, 4, A, 1, DET, IPIV, 4, ISC)
 6000 CONTINUE
      SUM = A(1)
      VARP = 1.
      DO   6100  K=2,4
      VARP = VARP*VAR
      SUM  = SUM + A(K)*VARP
 6100 CONTINUE
      YOUT(IYNX) = SUM
      VAR=FLOAT(IYNX)*DX
      IYNX = IYNX + 1
      IF (VAR .LE. X(N-1)) GO TO 6000
      XM = XN
      YM = YN
      XN = XO
      YN = YO
      XO = XP
```

```
      YO = YP
      IF (N .LT. KNT)  GO TO 6500
      IF (N .EQ. KNT+1)  GO TO 7000
      XP = 2.*X(KNT) - X(KNT-1)
      YP = 2.*Y(KNT) - Y(KNT-1)
      GO TO 6600
 6500 CONTINUE
      XP = X(N+1)
      YP = Y(N+1)
 6600 CONTINUE
      SLN = SLO
      SLO = SLP
      SLP = (YP - YO)/(XP - XO)
 7000 CONTINUE
      IF(IYNX.GT.KNTOLT) RETURN
      DO 2 I=IYNX,KNTOUT
    2 YOUT(I)=Y(KNT)
      RETURN
      END
      OVERLAY(PATS,4,0)
      PROGRAM CROSSSP
      COMMON CMAIN(1)
      COMMON/BLK1/STARTT,ITFMT,NBLK,IPOW2,NCH,NPRINT,IPLOTA,IPLOTC,OFFSC
     1AL,DELTAT,SN,NRSKIP,LAP,NCROSS,ICROSS(2,20),NCHP,YLABEL(2),IWINDOW
     1,F1,F2,ITYPESP,WCON,NFSKIP,IFF,LAG1,LAG2
      COMMON/BLK2/ICH(14),CHSUM(14),NOFF(14),CHSUMSQ(14),SIGMA(14),RMS(1
     14),MEAN(14),SCALFAC(14),CHSUM1(14),TRACK(14),ICHAN(14)
      COMMON/BLK3/NPT,TMAX,NPTO2,NSPCT,DELF,N64,NPTO128
     1,INZERO,NREAD
      COMMON/BLK6/ISAVE64,IRI,IXPLOT,IDATA,IZ,ISPECT
      COMMON/BLK8/IAUTOSP(14),IAUTOCO(14),ICRSP(20),ICRCOR(20),ITRA(20),
     1ICOH(20)
      COMMON/BLK9/NFILTP
      COMMON/BLK10/NRCRD7
      CALL CROSS(CMAIN(IZ),CMAIN(ISPECT),CMAIN(IRI),CMAIN(IXPLOT))
      RETURN
      END
      SUBROUTINE CROSS(Z,SPECT,RI,XPLOT)
      COMPLEX SPECT(1),Z(1)
      DIMENSION RI(1),XPLOT(1),FRAMEL(3),PLABEL(6)
      COMMON/BLK1/STARTT,ITFMT,NBLK,IPOW2,NCH,NPRINT,IPLOTA,IPLOTC,OFFSC
     1AL,DELTAT,SN,NRSKIP,LAP,NCROSS,ICROSS(2,20),NCHP,YLABEL(2),IWINDOW
     1,F1,F2,ITYPESP,WCON,NFSKIP,IFF,LAG1,LAG2
      COMMON/BLK2/ICH(14),CHSUM(14),NOFF(14),CHSUMSQ(14),SIGMA(14),RMS(1
     14),MEAN(14),SCALFAC(14),CHSUM1(14),TRACK(14),ICHAN(14),IFILTER(14)
      COMMON/BLK3/NPT,TMAX,NPTO2,NSPCT,DELF,N64,NPTO128
     1,INZERO
      COMMON/BLK8/IAUTOSP(14),IAUTOCO(14),ICRSP(20),ICRCOR(20),ITRA(20),
     1ICOH(20)
      COMMON/BLK9/NFILTP
      COMMON/BLK10/NRCRD7
```

APPENDIX H

```
      DIMENSION IDEN(5),BAND(27),PPLOT(27)
      DATA RAD/57.2957795/
C
C     START OF LOOP FOR COMPUTING CROSS FUNCTIONS
C
      DO 70 ICR=1,NCROSS
      K1=ICROSS(1,ICR) $ K2=ICROSS(2,ICR)
      IF(IAUTOSP(K1).EQ.0.OR.IAUTOSP(K2).EQ.0) GO TO 75
      DO 77 I=1,NCHP
      IF(K1.EQ.ICHAN(I)) J1=I
      IF(K2.EQ.ICHAN(I)) J2=I
   77 CONTINUE
      DO 76 I=1,NSPCT
   76 SPECT(I)=0.
C
C     AVERAGE CROSS SPECTRA FOR ONE PAIR
C
      DO 71 IBLK=1,NBLK
      IJ=J1+(IBLK-1)*NCHP
      CALL READMS(8,Z,NPT,IJ)
      IJ=J2+(IBLK-1)*NCHP
      CALL READMS(8,Z(NSPCT+1),NPT,IJ)
      DO 71 I=1,NSPCT
   71 SPECT(I)=SPECT(I)+Z(I)*CONJG(Z(I+NSPCT))
      IF(NFILTP.LE.0) GO TO 111
      IF(IFILTER(J1).EQ.0.AND.IFILTER(J2).EQ.0) GO TO 111
      CALL READMS(8,RI,NSPCT,NBLK*NCHP+1)
      DO 112 I=1,NSPCT
  112 SPECT(I)=SPECT(I)*RI(I)
  111 CONTINUE
      CALL WRITMS(9,SPECT,NPT,NCHP+1)
C
C     COMPUTE CORRECTED CROSS SPECTRUM
C
      CON=DELTAT*DELTAT/(6.28318308*WCON*NBLK)
      DO 115 I=1,NSPCT
  115 SPECT(I)=SPECT(I)*CON
      ENCODE(23,900,FRAMEL) TRACK(K1),TRACK(K2)
  900 FORMAT(A10,3H X ,A10)
      IF(ICRSP(ICR).LE.0) GO TO 105
      GO TO (60,61),ITYPESP
   60 CON=CON*DELF
      ENCODE(50,901,PLABEL) DELF
  901 FORMAT(*CROSS POWER SPECTRUM  (BANDWIDTH =*G9.2,*)*,6X)
      NPLABEL=44
      PRINT 898, (PLABEL(I),I=1,5),FRAMEL
  898 FORMAT(*1*5A10,5X,3A10)
      ENCODE(60,921,IDEN) TRACK(K1),TRACK(K2)
  921 FORMAT(*CROSS POWER SPECTRUM  *A10,1X,A10,7X)
      GO TO 62
   61 ENCODE(50,902,PLABEL)
```

```
  902 FORMAT(*CROSS POWER SPECTRAL DENSITY*22X)
      NPLABEL=28
      PRINT 899, (PLABEL(I),I=1,3),FRAMEL
  899 FORMAT(*1*3A10,5X,3A10)
      ENCODE(60,922,IDEN) TRACK(K1),TRACK(K2)
  922 FORMAT(*CROSS POWER SPECTRAL DENSITY *A10,1X,A10)
   62 IF(NPRINT.GT.0) PRINT 897
  897 FORMAT(//4X*I*4X*FREQUENCY*9X*REAL*10X*IMAG*7X*AMPLITUDE*7X*PHASE*
     1)
C
C     SET UP PLOT ARRAYS AND PRINT CROSS SPECTRUM
C
      DO 72 I=1,NSPCT
      F=(I-1)/TMAX
      XPLOT(I)=F
      AMP=CABS(SPECT(I))
      IF(AMP) 79,80,79
   80 ARG=0. $ GO TO 78
   79 ARG=ATAN2(AIMAG(SPECT(I)),REAL(SPECT(I)))*RAD
   78 CONTINUE
      IF(I-NPRINT)73,73,200
   73 PRINT 915, I,F,SPECT(I),AMP,ARG
  915 FORMAT(I5,5E14.5)
  200 IF(IPLOTC.EQ.0) GO TO 201
      GO TO (201,201,202,202,202,202),IPLOTC
  201 RI(I)=SPECT(I)
      RI(I+NSPCT+2)=AIMAG(SPECT(I))
      GO TO 72
  202 RI(I)=AMP
      RI(I+NSPCT+2)=ARG
   72 CONTINUE
      IF(IPLOTC-2) 301,301,302
  301 IDEN6=10H (REAL)
      IDEN7=10H (IMAG)
      GOTO 303
  302 IDEN6=10H AMPLITUDE
      IDEN7=10H  PHASE
  303 NRCRD7=NRCRD7+1
      NSPCTP2=NSPCT+2
      PRINT 1000, NRCRD7,IDEN,IDEN6
 1000 FORMAT(//21H * * * * * RECORD NO.,I5, * ON TAPE7 CONTAINS *6A10,10
     1H * * * * *)
      WRITE(7) IDEN,IDEN6,NSPCT,(XPLOT(I),RI(I),I=1,NSPCT)
      NRCRD7=NRCRD7+1
      PRINT 1000, NRCRD7,IDEN,IDEN7
      WRITE(7) IDEN,IDEN7,NSPCT,(XPLOT(I),RI(I+NSPCTP2),I=1,NSPCT)
C
C     PLOT FANFOLD PLOTS
C
      NW=(NPLABEL-1)/10+2
      NPLABEL=NW*10
```

APPENDIX H

```
      IF(IPLOTC.EQ.0) GO TO 105
      GO TO (107,107,108,109,109,109) IPLOTC
  107 ILOG=0 $ GO TO 106
  108 ILOG=2
  106 ENCODE(10,903,PLABEL(NW))
  903 FORMAT(*     (REAL)*)
      CALL PLOTNB(YLABEL,FRAMEL,23,XPLOT,RI,NSPCT,ILOG,F1,F2,PLABEL,NPLA
     1BEL,IFF,1)
      ENCODE(10,904,PLABEL(NW))
  904 FORMAT(*     (IMAG)*)
      CALL PLOTNB(YLABEL,FRAMEL,23,XPLOT,RI(NSPCT+3),NSPCT,ILOG,F1,F2,
     1PLABEL,NPLABEL,IFF,0)
      GO TO 105
  109 ILOG=IPLOTC-3
  104 ENCODE(10,905,PLABEL(NW))
  905 FORMAT(* MAGNITUDE*)
      CALL PLOTNB(YLABEL,FRAMEL,23,XPLOT,RI,NSPCT,ILOG,F1,F2,PLABEL,NPLA
     1BEL,IFF,1)
      ENCODE(10,906,PLABEL(NW))
  906 FORMAT(*      PHASE*)
      IF(ILOG.GT.1) ILOG=ILOG-2
      CALL PLOTNB(YLABEL,FRAMEL,23,XPLOT,RI(NSPCT+3),NSPCT,ILOG,F1,F2,
     1PLABEL,NPLABEL,IFF,0)
  105 CONTINUE
      IF(ICRCOR(ICR).LE.0) GO TO 411
C
C     COMPUTE CROSSCORRELATION
C
      DO 74 I=2,NSPCT
      Z(I)=SPECT(I)
   74 Z(NSPCT+I)=CONJG(SPECT(NSPCT-I+2))
      Z(1)=Z(NSPCT+1)=SPECT(1)
      IF(ITYPESP.GT.1) GO TO 113
      DO 114 I=1,NPT
  114 Z(I)=Z(I)/DELF
  113 CONTINUE
      CALL FOURT(Z,NPT,1,-1,1,SPECT)
      PRINT 916, TRACK(K1),TRACK(K2)
      CON=6.283185308/TMAX
      DO 110 I=1,NPT
  110 SPECT(I)=Z(I)*CON
C
C SET UP PLOT ARRAYS, PRINT AND PLOT CORRELATION
C
      IF(INZERO) 500,500,501
  500 DO 502 I=1,NSPCT
      IM1=I-1
      IPN=I+NSPCT
      RI(IPN)=SPECT(I)
      XPLOT(IPN)=IM1
      IMN=I-NSPCT-1
```

```
      RI(I)=SPECT(IPN)
  502 XPLOT(I)=IMN
      GO TO 503
  501 DO 90 I=1,NSPCT
      IM1=I-1
      IPN=I+NSPCT
      I2M1=2*IM1
      RI(IPN)=NPT*SPECT(I)/(NPT-I2M1)
      XPLOT(IPN)=IM1
      IMN=I-NSPCT-1
      I2MN=2*IMN $ IF(I.EQ.1) I2MN=I2MN+1
      RI(I)=NPT*SPECT(IPN)/(NPT+I2MN)
   90 XPLOT(I)=IMN
  503 CONTINUE
      PRINT 917, (XPLOT(I),RI(I),I=1,NPT)
  917 FORMAT(2X,8(F5.0,E11.3))
  916 FORMAT(//*1CROSSCORRELATION, *A10,* X *A10//8(7X*I*5X*RXY*))
      ENCODE(60,923,PLABEL) TRACK(K1),TRACK(K2)
  923 FORMAT(*CROSSCORRELATION    *A10,1X,A10,20X)
      WRITE(7) PLABEL,NPT,(XPLOT(I),RI(I),I=1,NPT)
      NRCRD7=NRCRD7+1
      PRINT 1000, NRCRD7,PLABEL
      IF(LAG1.EQ.0.AND.LAG2.EQ.0) GO TO 411
      I1=LAG1+NSPCT+1 $ I2=LAG2+NSPCT+1
  206 IF(I1.GE.I2) GO TO 411
      NPLT=I2-I1+1
      CALL ASCALE(RI(I1),10.,NPLT,1,10.)
      IF(IFF.EQ.0) GO TO 411
      IF(NPLT-256) 101,101,102
  101 K=1 $ GO TO 103
  102 K=NPLT/256
  103 ENCODE(50,920,IDEN) FRAME1
  920 FORMAT(*CROSS CORRELATION*3X,3A10)
      PN=RI(NPLT+I1)
      PX=RI(NPLT+I1+1)*10.+PN
      CALL FANFOLD(RI(I1),NPLT,K,1,NPLT,IDEN,1H*,1.,PX,PN,YLABEL,2,120,0
     1,0.,1H )
      GO TO 411
   75 PRINT 918, TRACK(K1),TRACK(K2)
  918 FORMAT(//* CROSS SPECTRA FOR *A10,* X *A10,* CANNOT BE COMPUTED*)
      GO TO 70
  411 IF(ICOH(ICR).LE.0) GO TO 401
C
C     COMPUTE COHERENCE
C
      CALL READMS(9,RI,NSPCT,J1)
      CALL READMS(9,RI(NSPCT+1),NSPCT,J2)
      CALL READMS(9,SPECT,NPT,NCHP+1)
      CON=DELTAT**2/(6.283185308*WCON*NBLK)
      DO 402 I=1,NSPCT
      XPLOT(I)=(I-1)/TMAX
```

```
      DENOM=.25*(RI(I)*RI(NSPCT+I))
      IF(DENOM) 412,412,413
  412 RI(I)=0. $ GOTO 402
  413 RI(I)=CABS(SPECT(I)/SQRT(DENOM))*CON
  402 CONTINUE
      PRINT 403, FRAMEL,(I,XPLOT(I),RI(I),I=1,NPRINT)
C
C     PRINT AND PLOT COHERENCE
C
  403 FORMAT(//*1COHERENCE *3A10//5X*I*3X*FREQUENCY*4X*COHERENCE*3(6X*I*
     13X*FREQUENCY*4X*COHERENCE*)/(I6,2E13.5,I6,2E13.5,I6,2E13.5,I6,2E13
     2.5))
      ENCODE(60,924,PLABEL) TRACK(K1),TRACK(K2)
  924 FORMAT(*COHERENCE  *A10,1X,A10,28X)
      NRCRD7=NRCRD7+1
      WRITE(7) PLABEL,NSPCT,(XPLOT(I),RI(I),I=1,NSPCT)
      PRINT 1000, NRCRD7,PLABEL
      ENCODE(50,404,PLABEL)
  404 FORMAT(*COHERENCE*41X)
      CALL PLOTNB(YLABEL,FRAMEL,23,XPLOT,RI,NSPCT,0,F1,F2,PLABEL,10,IFF,
     .11)
  401 IF(ITRA(ICR).EQ.0) GO TO 70
C
C     COMPUTE TRANSFER FUNCTION
C
      CALL READMS(9,SPECT,NPT,NCHP+1)
      IF(ITRA(ICR).LT.0) GO TO 405
      CALL READMS(9,RI,NSPCT,J1)
      ENCODE(60,406,PLABEL) TRACK(K1),TRACK(K2)
      GO TO 407
  405 CALL READMS(9,RI,NSPCT,J2)
      ENCODE(60,408,PLABEL) TRACK(K1),TRACK(K2)
  408 FORMAT(*TRANSFER FUNCTION, TRAYX FOR *A10,3H X ,A10,8X)
  406 FORMAT(*TRANSFER FUNCTION, TRAXY FOR *A10,3H X ,A10,8X)
  407 CON=DELTAT**2/(6.28318306*WCON*NBLK)
      DO 409 I=1,NSPCT
      XPLOT(I)=(I-1)/TMAX
      DENOM=.5*RI(I)
      IF(DENOM) 414,414,415
  414 RI(I)=0. $ GO TO 409
  415 RI(I)=CABS(SPECT(I)/DENOM)*CON
  409 CONTINUE
C
C     PRINT AND PLOT TRANSFER FUNCTION
C
      PRINT 410, PLABEL,(I,XPLOT(I),RI(I),I=1,NPRINT)
  410 FORMAT(//*1*6A10,        //5X*I*3X*FREQUENCY,*7X*TRA*3(9X*I*3X*FREQ
     1UENCY*7X*TRA*)/(I6,2E13.5,I6,2E13.5,I6,2E13.5,I6,2E13.5))
      NRCRD7=NRCRD7+1
      WRITE(7) PLABEL,NSPCT,(XPLOT(I),RI(I),I=1,NSPCT)
      PRINT 1000, NRCRD7,PLABEL
      CALL PLOTNB(YLABEL,FRAMEL,23,XPLOT,RI,NSPCT,0,F1,F2,PLABEL,24,IFF,
     .11)
   70 CONTINUE
      RETURN
      END
```

REFERENCES

1. Welch, Peter D.: The Use of Fast Fourier Transform for the Estimation of Power Spectra: A Method Based on Time Averaging Over Short, Modified Periodograms. IEEE Trans. Audio & Electroacoust., vol. AU-15, no. 2, June 1967, pp. 70-73.

2. Otnes, Robert K.; and Enochson, Loren: Digital Time Series Analysis. John Wiley & Sons, Inc., c.1972.

3. Blackman, R. B.; and Tukey, J. W.: The Measurement of Power Spectra. Dover Publ., Inc., 1959.

4. Greville, T. N. E.: Spline Functions, Interpolation, and Numerical Quadrature. Mathematical Methods for Digital Computers, Vol. II, Anthony Ralston and Herbert S. Wilf, eds., John Wiley & Sons, Inc., c.1967, pp. 156-168.

5. Bendat, Julius S.; and Piersol, Allan G.: Measurement and Analysis of Random Data. John Wiley & Sons, Inc., c.1966.

"The aeronautical and space activities of the United States shall be conducted so as to contribute . . . to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

—NATIONAL AERONAUTICS AND SPACE ACT OF 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

TECHNICAL REPORTS: Scientific and technical information considered important, complete, and a lasting contribution to existing knowledge.

TECHNICAL NOTES: Information less broad in scope but nevertheless of importance as a contribution to existing knowledge.

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons. Also includes conference proceedings with either limited or unlimited distribution.

CONTRACTOR REPORTS: Scientific and technical information generated under a NASA contract or grant and considered an important contribution to existing knowledge.

TECHNICAL TRANSLATIONS: Information published in a foreign language considered to merit NASA distribution in English.

SPECIAL PUBLICATIONS: Information derived from or of value to NASA activities. Publications include final reports of major projects, monographs, data compilations, handbooks, sourcebooks, and special bibliographies.

TECHNOLOGY UTILIZATION PUBLICATIONS: Information on technology used by NASA that may be of particular interest in commercial and other non-aerospace applications. Publications include Tech Briefs, Technology Utilization Reports and Technology Surveys.

Details on the availability of these publications may be obtained from:

SCIENTIFIC AND TECHNICAL INFORMATION OFFICE

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D.C. 20546