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
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.
# CONTENTS

<table>
<thead>
<tr>
<th>CONTENTS</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUMMARY</td>
<td>1</td>
</tr>
<tr>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>SYMBOLS</td>
<td>3</td>
</tr>
<tr>
<td>THEORY AND EQUATIONS</td>
<td>7</td>
</tr>
<tr>
<td>Discrete Fourier Transform</td>
<td>7</td>
</tr>
<tr>
<td>Spectral Representations</td>
<td>9</td>
</tr>
<tr>
<td>Data and Spectral Windows</td>
<td>11</td>
</tr>
<tr>
<td>Variation of Estimates</td>
<td>14</td>
</tr>
<tr>
<td>One-Third-Octave Power Spectra and General Power Spectra</td>
<td>16</td>
</tr>
<tr>
<td>Estimated Cross Power Spectra</td>
<td>17</td>
</tr>
<tr>
<td>Estimated Autocorrelation Functions</td>
<td>17</td>
</tr>
<tr>
<td>Spectral Filtering and Narrow-Band Correlation Functions</td>
<td>19</td>
</tr>
<tr>
<td>Estimates of Cross Correlation Functions</td>
<td>19</td>
</tr>
<tr>
<td>Transfer and Coherence Functions</td>
<td>20</td>
</tr>
<tr>
<td>Histograms</td>
<td>21</td>
</tr>
<tr>
<td>PROGRAM DESCRIPTION</td>
<td>22</td>
</tr>
<tr>
<td>Operating Environment</td>
<td>22</td>
</tr>
<tr>
<td>Program Specifications</td>
<td>23</td>
</tr>
<tr>
<td>OPERATING INSTRUCTIONS</td>
<td>23</td>
</tr>
<tr>
<td>Deck Setups for Langley Operating System</td>
<td>23</td>
</tr>
<tr>
<td>Card Input Data Description</td>
<td>25</td>
</tr>
<tr>
<td>Output Description</td>
<td>30</td>
</tr>
<tr>
<td>Restrictions and Limitations</td>
<td>32</td>
</tr>
<tr>
<td>Error Messages and Remedies</td>
<td>33</td>
</tr>
<tr>
<td>CONCLUDING REMARKS</td>
<td>34</td>
</tr>
<tr>
<td>APPENDIX A — FINITE FOURIER TRANSFORM OF A PERIODIC SIGNAL</td>
<td>35</td>
</tr>
<tr>
<td>APPENDIX B — SPECTRAL ESTIMATION THROUGH USE OF THE FINITE FOURIER TRANSFORM</td>
<td>39</td>
</tr>
<tr>
<td>APPENDIX C — AUTOCORRELATION ESTIMATION FROM ESTIMATED POWER SPECTRAL DENSITY</td>
<td>43</td>
</tr>
<tr>
<td>APPENDIX D — BINARY INPUT TAPE FORMATS</td>
<td>46</td>
</tr>
<tr>
<td>Tape Format 1</td>
<td>46</td>
</tr>
<tr>
<td>Tape Format 2</td>
<td>47</td>
</tr>
</tbody>
</table>
Tape Format 3 ................................................................. 50

APPENDIX E — FLOW CHART FOR PATS ................................. 51

APPENDIX F — PROGRAMS AND SUBPROGRAMS USED BY PATS ....... 54

APPENDIX G — LANGLEY LIBRARY SUBROUTINES ......................... 57
  Subroutine AScale ...................................................... 57
  Function GAMMF ....................................................... 58
  Subroutine ITR2 ........................................................ 60
  Subroutine OPENMS ............................................... 62
  Subroutine READMS .................................................. 63
  Subroutine REcin ..................................................... 64
  Subroutine RECOUT ................................................. 66
  Subroutine SIMEQ .................................................... 68
  Subroutine WRITMS .................................................. 69

APPENDIX H — SOURCE LISTING OF PATS ................................. 70

REFERENCES .............................................................. 114
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 gen-
eral 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.
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

\[ E(\cdot) \] expectation operator

\[ f(t) \] continuous function of time

\[ f_j \] frequency of data points in amplitude bin \( j \)

\[ F(\omega) \] continuous function of \( \omega \), Fourier transform of \( f(t) \)

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

\[ G(\omega) \] continuous function of \( \omega \), Fourier transform of a linear system response

\[ h(t) \] continuous function of time

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

\[ i = \sqrt{-1} \]

\[ \text{Im}(\cdot) \] 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 \(\chi^2\)

\(P_f(\omega)\) continuous function of \(\omega\), power spectrum of the time function \(f(t)\)

\(P_m\) power in the \(m\)th 1/3-octave band

\(R_f(\tau)\) continuous function of time lag \(\tau\), autocorrelation function of \(f(t)\)

\(R_x(\tau)\) autocorrelation function (eq. (B4))

\(R_{xy}(\tau)\) continuous function of time lag \(\tau\), cross correlation function of \(x(t)\) and \(y(t)\)

\(\operatorname{Re}(\text{ })\) real part of a complex number

\(S_f(\omega)\) continuous function of \(\omega\), power spectral density of \(f(t)\)

\(S_m\) power spectral density in the \(m\)th 1/3-octave band

\(S_x(\omega)\) continuous function of \(\omega\), power spectral density of \(x(t)\)

\(S_y(\omega)\) continuous function of \(\omega\), power spectral density of \(y(t)\)

\(S_{xy}(\omega)\) continuous function of \(\omega\), 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

\( \text{var}(\cdot) \)  
\( \) 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 \( \omega \), Fourier transform of \( x(t) \)

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

\( y(t) \)  
\( \) continuous function of time

\( Y(\omega) \)  
\( \) continuous function of \( \omega \), 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 \( z_k = \sum_{j=0}^{N-1} z_j W_j^k \)
\( \alpha \) significance level of a \( \chi^2 \) distribution

\( \beta \) factor dependent on window chosen

\( \gamma^2_{xy}(\omega) \) continuous function of \( \omega \), coherence function of \( x(t) \) and \( y(t) \)

\( \Gamma() \) incomplete gamma function

\( \delta() \) Dirac delta function

\( \Delta \) change

\( \mu \) mean value

\( \nu \) positive integer

\( \sigma^2 \) variance of the random process \( x \)

\( \phi_k \) discrete phase angle, deg

\( \chi^2 \) chi-square random variable

\( \chi^2_{c} \) critical value of \( \chi^2 \)

\( \chi^2_{e} \) expected value of \( \chi^2 \)

\( \omega \) frequency, rad/sec

\( \omega_k, \omega_n \) discrete frequencies, rad/sec

\( \omega_\nu, f_\nu \) Nyquist frequency, rad/sec and Hz, respectively

Mathematical notation:

\( \sim \) estimated quantities

\( ' \) new variable

\( * \) complex conjugate operator
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

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

(1)

\[ f(t) = \int_{-\infty}^{\infty} F(\omega) e^{i\omega t} \, d\omega \]  

(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 \([-T/2, T/2]\) continuously, then the finite Fourier transform of \( f(t) \) is given by

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

(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

\[
\omega_k = \frac{2\pi k}{N \Delta t} \quad (k = 0, 1, 2, \ldots, N/2)
\]

by the discrete Fourier transform (DFT) given by

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

where \( \Delta t \) 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 \( \omega_v = \pi/\Delta t \), 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 \( \omega_v \) 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

\[
z_k = \sum_{j=0}^{N-1} z_j W^{jk} \quad (5)
\]

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

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

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

\[
\hat{A}_k = \frac{2}{N} \text{Re}\left( z_{\nu k} \right) \quad (k = 0, 1, 2, \ldots, N/2\nu)
\]

\[
\hat{B}_k = -\frac{2}{N} \text{Im}\left( z_{\nu k} \right)
\]

where \( \hat{\cdot} \) indicates an estimate of the required quantity. An estimate of the phase \( \phi_k \) at frequency \( \omega_k \) 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 $\Delta t$ are available, a spectral estimate may be obtained by employing these values in equation (5). Then, by equation (4),

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

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 \( \Delta t \) and these are employed in the standard transform (eq. (5)), then the spectral estimate becomes

\[
\hat{S}_f(\omega_k) = \frac{\Delta t}{4\pi N} \left| z_k \right|^2
\]  \hspace{1cm} (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:

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

$$F_T(\omega) = \int_{-\infty}^{\infty} F(\omega') u_{T/2}(\omega - \omega') \, d\omega'$$  \hspace{1cm} (11)

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

$$u_{T/2}(\omega) = \frac{T}{2\pi} \frac{\sin(\omega T/2)}{\omega T/2}$$  \hspace{1cm} (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
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 $\omega'$. 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

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

the Hamming data window given by

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

and the Parzen data window given by

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

$$\tilde{S}_f(\omega_k) = \frac{(\Delta t)^2}{2\pi W_u} |z_k|^2$$  \hspace{1cm} (16)

where

$$W_u = \int_{-\infty}^{\infty} w_d^2(t) \, dt$$  \hspace{1cm} (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 \(\omega\), 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

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

where \(\text{E}\) and \(\text{var}\) indicate the ensemble expectation and variance of the random variable \(\tilde{S}_f(\omega)\), respectively, and \(\beta\) 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,

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

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)

$$\frac{k_{eq} \tilde{S}_f(\omega)}{2 \chi^2_{k_{eq}}(\frac{\alpha}{2})} < \text{E}[\tilde{S}_f(\omega)] \leq \frac{k_{eq} \tilde{S}_f(\omega)}{2 \chi^2_{k_{eq}}(1 - \frac{\alpha}{2})}$$

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

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

$$\int_{b}^{\infty} p\left(\frac{2}{\chi^2_{k_{eq}}}\right) \text{d}\chi^2_{k_{eq}} = \alpha$$

The function $p\left(\frac{2}{\chi^2_{k_{eq}}}\right)$ 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 $k_{eq} \approx \frac{36N_t}{11N}$, 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.
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 $\Delta t$ 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

$$\hat{P}_f(\omega_k) = \frac{\Delta t}{NW_d} |z_k|^2$$  \hspace{1cm} (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

\[
\hat{P}_m = \sum_{\Delta \omega_m} \hat{P}_f(\omega_k)
\]  

(20)

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

\[
\hat{S}_m = \frac{\hat{P}_m}{2\pi \Delta \omega_m}
\]  

(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:

\[
\hat{S}_{xy}(\omega_k) = \frac{(\Delta t)^2}{2\pi W_u} z_k z_k',
\]  

(22)

where

\[
\begin{align*}
z_k &= \sum_{j=0}^{N-1} x(j \Delta t) W_{jk} \\
z_k' &= \sum_{j=0}^{N-1} y(j \Delta t) W_{jk}
\end{align*}
\]  

(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:
\[ \tilde{R}_f(\tau) = \int_{-\infty}^{\infty} e^{i\omega \tau} \tilde{S}_f(\omega) \, d\omega. \] (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 \( \tau \). Thus, it is necessary to introduce a new estimate

\[ \tilde{R}_x(\tau) = W_R \int_{-\infty}^{\infty} \tilde{S}_x(\omega) e^{i\omega \tau} \, d\omega \] (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 \( \tau \) 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](https://example.comINLINE_IMAGE)  
**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.](image)

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

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

$$\gamma_{xy}^2(\omega) = \frac{|\tilde{S}_{xy}(\omega)|^2}{\tilde{S}_x(\omega) \tilde{S}_y(\omega)}$$

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

$$|\tilde{S}_{xy}(\omega)|^2 \leq \tilde{S}_x(\omega) \tilde{S}_y(\omega)$$

then \(\gamma_{xy}^2(\omega) \leq 1\).

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

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

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 \(\gamma_{xy}^2\) as possible.
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.](image)

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

\[
\hat{\mu} = \frac{1}{N_t} \sum_{j=1}^{N_t} x_j
\]

and the sample variance \( \hat{\sigma}^2 \) as

\[
\hat{\sigma}^2 = \frac{1}{N_t - 1} \sum_{j=1}^{N_t} (x_j - \hat{\mu})^2.
\]

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 \( \chi^2 \) distribution with \( k_{eq} = (N_b - 3) \) degrees of freedom,
where \( N_b \) is the number of bins selected. The effective value of \( \chi^2 \), or \( \chi_e^2 \), is computed by

\[
\chi_e^2 = \sum_{j=1}^{N_b} \frac{(f_j - N_t p_j)^2}{N_t p_j}
\]  

(30)

The probability density functions of the \( \chi^2 \) random variable are given by

\[
p(\chi^2) = 2^{-n/2} \Gamma^{-1}_{\frac{k_{eq}}{2}}\left(\frac{\chi^2}{2}\right)^{\frac{k_{eq}-1}{2}} e^{-\chi^2/2}
\]

(31)

where \( n \) is the number of degrees of freedom. The critical value of \( \chi^2 \), or \( \chi_c^2 \), for the \( \alpha \) 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 \( \chi_e^2 \) is quite sensitive to \( N_b \). A detailed description of hypothesis testing using the \( \chi^2 \) 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 60000g for compiling and executing the source version presented here and 55000g 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.

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

1See Langley Computer Programing Manual for format.
RFL,10000.
DROPFIL(TAPE1)
REWIND(TAPE7)
1REQUEST,TAPE99,MY. SAVTP,RIS,your 3 initials, identification
COPYBF(TAPE7,TAPE99)
DROPFIL(TAPE99)
EXIT.
RFL,10000.
DROPFIL(TAPE1)
REWIND(TAPE7)
1REQUEST,TAPE99,MY. SAVTP,RIS,your 3 initials, identification
COPYBF(TAPE7,TAPE99)
DROPFIL(TAPE99)
27/8/9
(Data card deck inserted here)
36/7/8/9

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

JOB card
USER card
FETCH(A4119, SOURCE)
RUN(S, ,SCFILE)
LINECNT,10000.
1REQUEST,TAPE1,MY. tape no.,ROL,
REWIND(TAPE1)
SETINDF.
1.GO.

1See Langley Computer Programming Manual for format.
2End-of-record card.
3End-of-file card.
RFL,10000.
DROPFIL(TAPE1)
REWIND(TAPE7)
^REQUEST,TAPE99,HY. SAVTP,RIS,your 3 initials, identification
COPYBF(TAPE7,TAPE99)
DROPFIL(TAPE99)
EXIT.
RFL,10000.
DROPFIL(TAPE1)
REWIND(TAPE7)
^REQUEST,TAPE99,HY. SAVTP,RIS,your 3 initials, identification
COPYBF(TAPE7,TAPE99)
DROPFIL(TAPE99)

27/8/9
Mod card deck if any (may be a blank record)

27/8/9
(Data card deck inserted here)

36/7/8/9

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.

---

^See Langley Computer Programming Manual for format.

^End-of-record card.

^End-of-file card.

---
NAMELIST input format:

<table>
<thead>
<tr>
<th>FORTRAN name</th>
<th>Default value</th>
<th>Parameter type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$INPUT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ITFMT</td>
<td>2</td>
<td>I</td>
<td>Code for input tape format:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 tape format 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 tape format 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3 tape format 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(see appendix D for format descriptions)</td>
</tr>
<tr>
<td>NFSKIP</td>
<td>0</td>
<td>I</td>
<td>Number of logical binary files on input tape to be skipped before starting execution of this case</td>
</tr>
<tr>
<td>NRSKIP</td>
<td>0</td>
<td>I</td>
<td>Number of logical binary records on input tape to be skipped before starting execution of this case</td>
</tr>
<tr>
<td>SN</td>
<td>R</td>
<td></td>
<td>Serial number of input data</td>
</tr>
<tr>
<td>DELTAT</td>
<td>R</td>
<td></td>
<td>1/Sampling rate of input data</td>
</tr>
<tr>
<td>STARTT</td>
<td>0.0</td>
<td>R</td>
<td>Starting time in seconds at which program is to start processing data from input tape</td>
</tr>
<tr>
<td>OFFSCAL</td>
<td>$10^6$</td>
<td>R</td>
<td>Offscale value for all channels</td>
</tr>
<tr>
<td>NCH</td>
<td>I</td>
<td></td>
<td>Number of data channels on input tape (maximum value = 14)</td>
</tr>
<tr>
<td>SCALFAC</td>
<td>1.</td>
<td>R</td>
<td>Array of NCH values of scale factors, one for each channel of input data; every point for channel i is multiplied by SCALFAC(i).</td>
</tr>
<tr>
<td>NPTOT</td>
<td>I</td>
<td></td>
<td>Total number of data points to be read for each channel</td>
</tr>
<tr>
<td>NREAD</td>
<td>I</td>
<td></td>
<td>Number of data points per block to be read for each channel (maximum value is 1024 for INZER0=0, 512 for INZER0=1)</td>
</tr>
<tr>
<td>IAUTOSP</td>
<td>0</td>
<td>I</td>
<td>Array of NCH codes for computing auto spectra:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 compute auto spectrum for channel i</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 do not compute auto spectrum for channel i</td>
</tr>
<tr>
<td>FORTRAN name</td>
<td>Default value</td>
<td>Parameter type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------</td>
<td>----------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| IAUTOICO     | 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,  
|              |               |                | \( \tilde{H}(\omega) = \tilde{S}_{xy}(\omega)/\tilde{S}_y(\omega) \)  
<p>|              |               |                | 0 do not compute transfer function for pair i |</p>
<table>
<thead>
<tr>
<th>FORTRAN name</th>
<th>Default value</th>
<th>Parameter type</th>
<th>Description</th>
</tr>
</thead>
</table>
| 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 |
| IPLOTTC      | 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) |
<table>
<thead>
<tr>
<th>FORTRAN name</th>
<th>Default value</th>
<th>Parameter type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPLOTC</td>
<td>0</td>
<td>I</td>
<td>3 plot and save magnitude and phase (linear) against frequency (linear)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5 plot magnitude (log scale) and phase against frequency (linear) and save magnitude and phase values</td>
</tr>
<tr>
<td>F1</td>
<td>0.0</td>
<td>R</td>
<td>Lower limit of frequency to be plotted on narrow-band spectra plots</td>
</tr>
<tr>
<td>F2</td>
<td>20000.</td>
<td>R</td>
<td>Upper limit of frequency to be plotted on narrow-band spectra plots</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(If F1 and F2 are both zero, no narrow-band plots will be made)</td>
</tr>
<tr>
<td>LAG1</td>
<td>0</td>
<td>I</td>
<td>Lower limit of the number of time lags to be plotted on correlation plots</td>
</tr>
<tr>
<td>LAG2</td>
<td>0</td>
<td>I</td>
<td>Upper limit of the number of time lags to be plotted on correlation plots</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(If LAG1 and LAG2 are both zero, no correlation plots will be made)</td>
</tr>
<tr>
<td>PCTC</td>
<td>90.</td>
<td>R</td>
<td>Percent band to be used for calculation of confidence band and level of significance in chi-square calculation</td>
</tr>
<tr>
<td>NBINS</td>
<td>0</td>
<td>I</td>
<td>Number of bins to be used in histograms:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 no histograms</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(maximum value=100)</td>
</tr>
<tr>
<td>DMAX</td>
<td>0.</td>
<td>R</td>
<td>Array of values of maximum readings for each channel</td>
</tr>
<tr>
<td>DMIN</td>
<td>0.</td>
<td>R</td>
<td>Array of values of minimum readings for each channel</td>
</tr>
<tr>
<td>INZERO</td>
<td>0</td>
<td>I</td>
<td>Code for zero insertion option:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 insert NREAD zeros at end of input data block (block size is 2 x NREAD)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(this option should be used for runs requesting cross correlations)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 no zero insertion</td>
</tr>
<tr>
<td>FORTRAN name</td>
<td>Default value</td>
<td>Parameter type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------</td>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>NFILTP</td>
<td>0</td>
<td>I</td>
<td>Number of points in input spectral filter: 0 no spectral filter (maximum value=50)</td>
</tr>
<tr>
<td>FREQF</td>
<td>0.0</td>
<td>R</td>
<td>Array of values of frequency for spectral filter</td>
</tr>
<tr>
<td>WGHTF</td>
<td>0.0</td>
<td>R</td>
<td>Array of values of weights for spectral filter (user should be careful to specify points close together where derivative of filter function changes)</td>
</tr>
</tbody>
</table>

Input cards after NAMELIST input:

<table>
<thead>
<tr>
<th>Card no.</th>
<th>FORTRAN name</th>
<th>Format</th>
<th>Parameter type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>YLABEL</td>
<td>2A10</td>
<td>A</td>
<td>Array of two words (20 characters) to be written on each plot frame for case identification</td>
</tr>
<tr>
<td>2</td>
<td>TRACK</td>
<td>8A10</td>
<td>A</td>
<td>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</td>
</tr>
</tbody>
</table>

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 $R_{XY}(\gamma)$, 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 Fl 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:

<table>
<thead>
<tr>
<th>Word</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 to 6</td>
<td>A</td>
<td>Label describing function and channels</td>
</tr>
<tr>
<td>7</td>
<td>I</td>
<td>Number of points in output function, NP</td>
</tr>
</tbody>
</table>
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 SSOOOG for the absolute binary version and GOOOOG 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.
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.
Suppose that $f(t)$ is a signal with period $p$. Then, it can be represented by the Fourier series

$$f(t) = \frac{A_0}{2} + \sum_{n=1}^{\infty} (A_n \cos \omega_n t + B_n \sin \omega_n t)$$  \hspace{1cm} (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 $\Delta t$ are available for a total record length of $T = N \Delta t$.

The finite Fourier transform of this signal is given by

$$X_k = \sum_{j=0}^{N-1} f(j \Delta t) e^{-12\pi j k/N} \quad (k = 0, 1, 2, \ldots, N/2)$$

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 $\nu$ is a positive integer. In this case, the $m$th harmonic will be equal to the $k$th 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.
APPENDIX A

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

\[
\frac{X_{\nu m}}{N} = \frac{A_m - iB_m}{2} + \frac{1}{2} \sum_{l=1}^{\infty} \left( \frac{A_{mN}}{\nu + m} + \frac{A_{mN}}{\nu - m} \right) - \frac{1}{2} \sum_{l=1}^{\infty} \left( \frac{B_{mN}}{\nu + m} - \frac{B_{mN}}{\nu - m} \right)
\] (A2)

The summation terms in equation (A2) involve aliasing of power from higher frequencies. Note that the aliasing depends upon \( N/\nu \), 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

\[
\left| X_{\nu m} \right| = \sqrt{\frac{A_m^2 + B_m^2}{2}}
\]

and

\[
\tan \phi_m = \frac{B_m}{A_m} = \frac{\text{Im}(X_{\nu m})}{\text{Re}(X_{\nu m})}
\] (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

\[
A_n = \frac{2}{p} \int_0^p f(t) \cos \omega nt \, dt = 0
\]

and

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

Thus, the square wave admits the Fourier series representation

\[
f(t) = \frac{4M}{\pi} \sum_{n=1}^{\infty} \frac{1}{2n - 1} \sin \frac{2\pi(2n - 1)t}{p}
\] (A4)
APPENDIX A

The magnitude and phase of this representation are given by

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

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

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

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

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

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

\[
= \begin{cases} 
0 & \text{(k even)} \\
-2M \cot \frac{\pi k}{N} & \text{(k odd)}
\end{cases}
\]
APPENDIX A

Thus,

\[
\frac{X_k}{N} = \begin{cases} 
0 & \text{(k even)} \\
-\frac{12M}{N} \cot \frac{\pi k}{N} & \text{(k odd)} 
\end{cases}
\]

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

\[A_n = 0 \quad \text{and} \quad \phi_n = \frac{\pi}{2}\]

Further, when \( k \) is odd,

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

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

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

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

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

Taking the ensemble expectation of this quantity yields

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

where

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

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

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

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

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

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

\[ E[\hat{S}_X(\omega)] = \frac{\pi}{T} \int_{-\infty}^{\infty} \hat{S}_X(\omega') \, U_d(\omega - \omega') \, U_d(\omega' - \omega) \, d\omega' = \frac{\pi}{T} \int_{-\infty}^{\infty} \hat{S}_X(\omega') \, U_d^2(\omega' - \omega) \, d\omega' \quad (B7) \]

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

\[ \int_{-\infty}^{\infty} E[\hat{S}_X(\omega)] \, d\omega = \frac{\pi}{T} \frac{1}{4\pi^2} \int_{-\infty}^{\infty} dt_1 \int_{-\infty}^{\infty} u_d(t_1) u_d(t_2) \, R_X(t_1 - t_2) \, dt_2 \int_{-\infty}^{\infty} e^{i\omega(t_1 - t_2)} \, d\omega \quad (B8) \]

and since

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

equation (B8) becomes

\[ \int_{-\infty}^{\infty} E[\hat{S}_X(\omega)] \, d\omega = \frac{\pi}{T} \frac{1}{2\pi} \int_{-\infty}^{\infty} dt_1 \int_{-\infty}^{\infty} u_d(t_1) u_d(t_2) \, R_X(t_1 - t_2) \, \delta(t_1 - t_2) \, dt_2 \]

\[ = \frac{\pi}{T} \frac{1}{2\pi} \int_{-\infty}^{\infty} u_d^2(t) \, R_X(0) \, dt_1 \]

\[ = \frac{R_X(0)}{2T} \int_{-\infty}^{\infty} u_d^2(t_1) \, dt_1 \quad (B10) \]
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:

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

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

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

where

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

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

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

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

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

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

For the boxcar window,

$$W_u = T$$  \hfill (B15)

For the Hann window,

$$W_u = \frac{3T}{8}$$  \hfill (B16)
APPENDIX B

For the Hamming window,

\[ W_u = T \left( 0.3974 + \frac{0.9936}{\pi} \right) \]  \hspace{1cm} (B17)

And for the Parzen window,

\[ W_u = T \frac{151}{560} \]  \hspace{1cm} (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,

$$\hat{R}_X(\tau) = \int_{-\infty}^{\infty} \tilde{S}_X(\omega)e^{i\omega\tau} \, d\omega$$  \hspace{1cm} (C1)

Thus,

$$E[\hat{R}_X(\tau)] = \int_{-\infty}^{\infty} E[\tilde{S}_X(\omega)]e^{i\omega\tau} \, d\omega$$ \hspace{1cm} (C2)

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

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

Thus, equation (C2) becomes

$$E[\hat{R}_X(\tau)] = \frac{2\pi}{W_u} \int_{-\infty}^{\infty} d\omega' e^{i\omega'\tau} S_X(\omega') \int_{-\infty}^{\infty} d\omega \, U_d(\omega' - \omega)e^{-i(\omega' - \omega)\tau}$$ \hspace{1cm} (C3)

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

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

$$= \frac{2\pi}{W_u} R_X(\tau) \int_{-\infty}^{\infty} d\omega_0 U_d(\omega_0)e^{-i\omega_0\tau}$$ \hspace{1cm} (C4)

Recall the definition (from eq. (B6))

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

43
APPENDIX C

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

\[
\int_{-\infty}^{\infty} d\omega \ U_d^2(\omega) e^{-i\omega_0 \tau} = \frac{1}{2\pi} \int_{-\infty}^{\infty} dt \int_{-\infty}^{\infty} dt' \ u_d(t) u_d(t') \frac{1}{2\pi} \int_{-\infty}^{\infty} d\omega e^{-i\omega_0(t-t'+\tau)}
\]

(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

\[
E[\hat{R}_x(\tau)] = \frac{\int_{-\infty}^{\infty} u_d(t) u_d(t + \tau) dt}{\int_{-\infty}^{\infty} u_d^2(t) dt}
\]

(C6)

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

\[
\hat{R}_x(\tau) = w_R \int_{-\infty}^{\infty} \tilde{S}_x(\omega) e^{i\omega \tau} d\omega
\]
APPENDIX C

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:

<table>
<thead>
<tr>
<th>Word</th>
<th>Type</th>
<th>Contents and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Floating</td>
<td>The number of channels of data in this frame; less than 40 for continuous data and less than 100 for commutated data</td>
</tr>
<tr>
<td>2</td>
<td>Serial number; the input card format for serial number should be 6 digits wide</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>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</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Words 5 and 6 are additional engineering identification</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Words 7 and 8 are Greenwich Mean Time and are used only for telemetry data; for ground facilities, word 8 may be ground facilities</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Frame count which starts at 1 for each new serial number</td>
<td></td>
</tr>
</tbody>
</table>
APPENDIX D

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

The relationship between frames and records is shown below.

<table>
<thead>
<tr>
<th>Number of channels, N</th>
<th>Words per frame</th>
<th>Frames per record</th>
<th>Words per record</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 ≤ N ≤ 10</td>
<td>20</td>
<td>25</td>
<td>500</td>
</tr>
<tr>
<td>10 &lt; N ≤ 20</td>
<td>30</td>
<td>17</td>
<td>510</td>
</tr>
<tr>
<td>20 &lt; N ≤ 30</td>
<td>40</td>
<td>12</td>
<td>480</td>
</tr>
<tr>
<td>30 &lt; N ≤ 40</td>
<td>50</td>
<td>10</td>
<td>500</td>
</tr>
<tr>
<td>*30 &lt; N ≤ 100</td>
<td>110</td>
<td>4</td>
<td>440</td>
</tr>
</tbody>
</table>

*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
APPENDIX D

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

Record 1 ID Record

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

*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

<table>
<thead>
<tr>
<th>Word</th>
<th>Name</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>KEY</td>
<td>A</td>
<td>NAMES</td>
</tr>
<tr>
<td>2</td>
<td>NN</td>
<td>I</td>
<td>Number of unblocked parameters plus number of blocked parameters</td>
</tr>
<tr>
<td>3 NN + 2</td>
<td>NAMES</td>
<td>A</td>
<td>Names for unblocked data parameters followed by names for blocked data</td>
</tr>
</tbody>
</table>

Each parameter including time will have a name.

Record 3 UNITS Record

<table>
<thead>
<tr>
<th>Word</th>
<th>Name</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>KEY</td>
<td>A</td>
<td>UNITS</td>
</tr>
<tr>
<td>2</td>
<td>NN</td>
<td>I</td>
<td>Number of unblocked parameters plus number of blocked parameters</td>
</tr>
<tr>
<td>3 NN + 2</td>
<td>UNITS</td>
<td>A</td>
<td>Units for unblocked data followed by UNITS for blocked data</td>
</tr>
</tbody>
</table>

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

Record 4 through EOF, DATA Records

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

The data records are optimally packed to approach, but not exceed, 512 words per record.

The record size is determined as follows:

\[
\text{SIZE} = \text{NFR} \times \text{KCH} + \text{IWD} + 2
\]

where

\[
\text{NFR} = (510 - \text{IWD})/\text{KCH}
\]
APPENDIX D

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

<table>
<thead>
<tr>
<th>Word</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Serial number</td>
</tr>
<tr>
<td>2</td>
<td>Time</td>
</tr>
<tr>
<td>3</td>
<td>Data channel 1</td>
</tr>
<tr>
<td>4</td>
<td>Data channel 2</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>NCH + 2</td>
<td>Data channel NCH</td>
</tr>
</tbody>
</table>

All words are in the floating-point mode.
APPENDIX E - FLOWCHART FOR PATS

Read card input data

Read one block of tape input for each required channel

Calculate Fourier transform for each channel and store on random-access file

All input blocks read?

Yes

Are histograms desired?

Yes

Compute histograms and print fanfold plots

No

Go to B

B

Compute auto spectrum selected by ITYPESP

Print auto spectrum in tabular form and write on binary output file

Are both F1 and F2 nonzero?

No

Print fanfold plot as indicated by IPLOTA

Yes

C

Is autocorrelation desired for this channel?

No

Calculate inverse transform from auto PSD

Yes

Calculate autocorrelation

Go to C

A

Read transform for one channel from random-access file

Apply spectral filter if desired

Compute auto PSD

Is auto spectral output desired for this channel?

No

Go to C

Yes

Go to B
APPENDIX E

Print autocorrelation in tabular form and write on binary output file

Are both LAG1 and LAG2 nonzero?

Yes: Print fanfold plot

No: Are all desired auto functions complete?

Yes: End of case

No: Are any cross functions requested?

Yes: Read transform for each channel of a pair from random-access file

Apply spectral filter if desired

Compute cross PSD

Go to E

Is cross spectral output desired for this pair?

Yes: Compute cross spectrum selected by ITYPESP

Print cross spectrum in tabular form and write on binary output file

No: Are both LAG1 and LAG2 nonzero?

Yes: Print fanfold plot as indicated by IPLOTC

No: End of case

Go to G

Is cross correlation for this pair desired?

Yes: Calculate cross correlation from cross PSD

Print cross correlation in tabular form and write on binary output file

Go to F

Go to A
APPENDIX E

F
Are both LAG1 and LAG2 nonzero?
Yes
Print fanfold plot

G
Is coherence for this pair desired?
Yes
Compute coherence from auto and cross PSD's
Print coherence in tabular form and write on binary output file

Are both F1 and F2 nonzero?
No
Go to I
Yes
Print fanfold plot

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
Go to D
Yes
Print fanfold plot

Are all cross functions complete?
No
Go to I
Yes
End of case

Go to H

Go to I
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
APPENDIX F

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

CROSSSP Sets up storage arrays for CROSS
References CROSS
APPENDIX F

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 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.
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: \( Y = \text{GAMMF}(A, X) \), where \( \text{GAMMF}(A, X) \) is defined as the integral from \( X \) to \( \infty \) of \( \exp(-\mu) \) times \( \mu \) to the \( (A-1) \)th power \( d\mu \).

Restrictions: \( X \geq 0 \); when \( X = 0 \), \( A \) is not a nonpositive integer. The following subprograms are called by \( \text{GAMMF} \): 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 \( A = 0 \),

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

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

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

(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 \( A \neq 0, X \geq \sqrt{|A+1|} \),

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


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
- \( DELTX \) \( \Delta X \), 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 \( A < B \), \( \Delta X \) 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 \( \Delta X \) thereafter, up to and including a specified end point \( b \). A change of sign of the function across a \( \Delta X \) 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 \( \Delta X \) be given so that no more than one root is present within a \( \Delta X \) 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 \[ |X| > \epsilon_1, \]
\[ \left| \frac{X - X_{i-1}}{X_i} \right| \leq \epsilon_1 \]

If \[ X \leq \epsilon_1, \]
\[ |X - X_{i-1}| \leq \epsilon_2 \]

Storage: 260 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:** 1038 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

XXX XXXX 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: 1318 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 2001g.

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

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: 3018 locations.

Other coding information: Day file diagnostics and their meaning:

1. UNASSIGNED FILE MEDIUM FILE TAPE\textsubscript{nn} – 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 $512 \times 10^3$ CM words) to be written. RECIN is programmed 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 2001g. 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 51110 CM words, then RECOUT offers no advantage and should not be used.

If the programmer wishes to write a file containing multfile records 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 programmer 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 (51210 CM words) in a central memory buffer before transmitting it to the actual disk or tape file.

Storage: 3208 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 2001g.

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 $A$; $1 \leq N \leq NMAX$.
- **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:
  \[ \det(A) = (10^{100})^{ISCALE}(DETERM) \]
- **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.


Storage: 4328 locations.

Subroutine date: August 1, 1968.
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: 1024 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,C)
PROGRAM MAIN(INPUT,OUTPUT,TAPE1,TAPE5=INPUT,TAPE6=OUTPUT,TAPE7,TAPE8,TAPE9)
COMMON CMAIN(4102)

C C CMAIN MUST BE DIMENSIONED 4*NMAX+6
C
C KNDEX MUST BE DIMENSIONED NBCMAX+2
C
DIMENSION INDEX(129),KNDEX(802)
COMMON/BLK1/START,ITFMT,NBLK,IP0W2,NCH,NPRINT,IFF,IFFLF,IPLOT,OFFS
1AL,DELTAT,SN,NRSKIP,LAP,NRCROSS,ICPS(2),20,NCHP,YLABEL(2),IWINDOW
1,F1,F2,ITYPESP,WCON,NFSKIP,IFF,LAG1,LAG2
COMMON/BLK2/ICH(14),CHSUM(14),NFF(14),CHSUMSQ(14),SIGMA(14),RMS(1
14),MEAN(14),SCALEFA(14),CHSUM1(14),TRACK(14),ICHA(14),IFILTER(14)
COMMON/BLK3/NPT,TMAX,NPT02,NPRINT,DELF,N64,NPT0128
1,INZERO,INREAD,NPTUT
COMMON/BLK4/NMAX,NCHMAX,NCHC
COMMON/BLK5/INDEX,NMAX(14),DVMIN(14),DVMIN(14),BINN(100,14)
,CHSOC
COMMON/BLK6/ISAVE64,IRI,IXPLOT,1DATA,IZ,ISPECT
COMMON/BLK7/IWORDD(11)
COMMON/BLK8/IAUTOSP(14),IAUTOCL(14),ICRSP(20),ICRCOR(20),ITRA(20)
1,CHSOC
COMMON/BLK9/NXILIP,FREQ(50),WGMF(5C)
COMMON/BLK10/NRCDT
DATA NRCDT/70/

NMAX=1024
NCHMAX=14
NBCMAX=80
,IZ=IRI=1
,IXPLOT=IRI+NMAX+4
,ISPECT=ISAVE64=IXPLOT+NMAX+2
,1DATA=ISAVE64+64*NCHMAX
PATS=4HPATS
CALL OPENMS(9,INDEX,129,0)
CALL OPENMS(6,KNDEX,NBCMAX+2,0)
PRINT 1000
1000 FORMAT(1H1///////23X1H=43(12H *1/23,X1H*B5X1H*/23X1H*8A*F PR OGR
1 A M FOR ANALYSIS OF TIME SERIES #4X1H* 2/23X1H*85X1H*/23X1H*85X1H*/23X1H*43(14
14))

C C C READ NAMELIST AND CARD INPUT
C
C 100 CALL OVERLAY(PATS,1,0)
C
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

COMPUTE FOURIER TRANSFORM FOR NCH CHANNELS AND STORE ON RANDOM ACCESS FILE.
REPEAT UNTIL ALL DATA IS READ AND PROCESSED:

CALL OVERLAY(PATS,2,0)

AVERAGE NBLK POWER SPECTRA FOR EACH CHANNEL FOR AUTO SPECTRA AND
APPLY SPECTRAL FILTER. COMPUTE AUTOCORRELATION, PRINT FANFOLD PLOTS

CALL OVERLAY(PATS,3,0)

TO COMPUTE CROSS SPECTRA
GET NBLK TRANSFORMS FOR BOTH OF THE CHANNELS IN EACH SELECTED PAIR
AVERAGE THE PRODUCTS FOR NARROW BAND SPECTRA, COMPUTE CROSSCORRELATIONS,
COHERENCE, AND TRANSFER FUNCTIONS. PRINT FANFOLD PLOTS

IF(NCROSS.GT.0) CALL OVERLAY(PATS,4,0)

GO TO 100
END

SUBROUTINE PLOTNB(YLABEL,FRAMEL,NF,XPLOT,RI,NSPCT,ILOG,F1,F2,PLABEL,IL,EFF,ISEARCH)
DIMENSION YLABEL(2),FRAMEL(1),XPLOT(1),RI(1),PLABEL(5),FF10(5),
I1OEN(6)
ILOG - COVE FOR TYPE OF SCALE FOR AXES
= 0 BOTH SCALES LINEAR
= 1 HORIZ. SCALE LOG, VERT. SCALE LINEAR
= 2 HORIZ. SCALE LINEAR, VERT. SCALE LOG
= 3 BOTH SCALES LOG

OPTIONS 1 AND 3 ARE NOT USED BY PATS

ILOG=ILOG+1
IF(ISEARCH.EQ.0) GO TO 6
J1=1 & J2=NSPCT
DO 1 I=1,NSPCT
IG=I
IF(XPLOT(IG).GE.F1) GO TO 2
CONTINUE
1 CONTINUE
2 J1=J & IF(ILOG.EQ.1.OR.ILOG.EQ.3) J1=MAXC(J1,2)
DO 3 I=1,NSPCT
J=NSPCT-I+J1
IF(XPLOT(J).LE.F2) GO TO 4
3 CONTINUE
4 J2=J
NF=NF/10 & IF(NF*10.LT.NF) NF=NF+1
NFP=NF/10 & IF(NFP*10.LT.NFP) NFP=NFP+1
IF(J1.LT.J2) GO TO 5
PRINT 900, (PLABEL(I),I=1,NFP),(FRAMEL(I),I=1,NFP)
900 FORMAT(*HANDWIDTH FOR PlOTS TOO NARROW */ NU PLUT GENERATED FOR */ II0A10)
APPENDIX H

RETURN
5 NPLT=J2-J1+1
   IF(NPLT.EQ.NSPCT) GO TO 100
C MOVE PLOTTING REGION TO BEGINNING OF ARRAY
DO 4002 I=J1,J2
  J=I-J1+1
4002 XPLOT(J)=XPLUT(I)
6 CONTINUE
DO 4003 I=J1,J2
  J=I-J1+1
  RI(J)=RI(I)
4003 CONTINUE
100 CONTINUE
103 GO TO (105,105,106,106,107,107,108,108,110)
C LINEAR VERTICAL SCALE
105 CALL ASCAL(RI,10.,NPLT,1,10.)
   YMIN=RI(NPLOT+1) $ YMAX=YMIN+10.*RI(NPLOT+2)
   GO TO 107
C LOG VERTICAL SCALE
106 SMAX=-10.
   DO 108 I=1,NPLT $ IF(RI(I)) 112,113
112 R(I)=10. $ GC TO 108
113 R(I)=ALOG(RI(I)) $ SMAX=AMAX1(RI(I),SMAX)
108 CONTINUE
   IMAX=SMAX
   IF(IMAX.LT.SMAX) IMAX=IMAX+5
   YMIN=RI(NPLOT+1)=IMIN $ YMAX=IMAX $ RI(NPLOT+2)=5
   DO 114 I=1,NPLT $ IF(RI(I) .LT.YMIN) RI(I)=YMIN
114 CONTINUE
107 CONTINUE
   IF(NF.EQ.0) GO TO 24
   IF(NPLT.GT.256) NPLT=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) ENCOD(54,902,IEN,PLABEL)
   IF(ILOG.LT.1) ENCOD(60,902,IEN,PLABEL)
902 FORMAT(5X,5A10,5X)
   CALL FANFOLD(RI,NPLT,1,NPLT,1,YMAX,YMIN,FFID,NF+2,
913,0,XPLUT,YFREQUENCY)
901 FORMAT(*LOG *A10)
24 RETURN
END
SUBROUTINE FANFOLD(PLOT,NP,K,NF,NMAX,IDEN,CHAR,PNORM,PMAX,PMIN,
1ZLABEL,NYL,LINE,WRITEX,XARRAY,XLABEL)
DIMENSION PLOT(NMAX),CHAR(N),IDEN(N),PNORM(N),YLABEL(N)
1,PMAX(1),PMIN(1),PSCALE(1),PLINE(124),XARRAY(1)
IF(LINE=126) 7,7,6
7 NCHAR=104 $ GO TO 9
8 NCHAR=124
APPENDIX H

9 CONTINUE
IF(NF.GT.IO) MF=1C
PRINT 900, (YLABE1(I), I=1, NYL)
900 FORMAT(*1&48X$A10)
IF(1WRITEX, EQ.O) GO TO 10
PRINT 907, (XLABEL(J), J=1,6), (J, XARAY(J), J=1, NPT, K)
9C7 FORMAT(*45X$A10,$ CODE SCALE IS AS FOLLOWS*75X$ CODE *A1C,4($
10 CONTINUE
NSKIP=0
IF(NPT/K-15) 16, 16, 17
16 NSKIP=NSKIP+1 GO TO 13
17 IF(NPT/K-25) 18, 16, 13
18 NSKIP=1
13 CONTINUE
DO 1 N=1, NF
IF(PMIN(1).NE.PMAX(1)) GO TO 1
PMIN(1)=PMAX(1)=PLUT(1,1)
DO 2 J=1, NPT, K
PMIN(1)=PAMUM(1, PMAX(1), pmum(I))
PAMUM(1)=PAMUM(1, PMAX(1), pmum(I))
2 CONTINUE
1 PSOE=( PMAX(1)-PAMUM(1))*PNUM(I)/NCHAR-4
PRINT 901, (I, CHAP(I), (IDEM(J), J=1, 5), PNUM(I), PMAX(I), PAMUM(I),
1 PSOE(I, I=1, NF)
901 FORMAT(*53X*PLT DESCRIPTION*4X FUNCINON*5X$ SCALE*4X$ NO.*4X$ C
1 CHARACTER*5X$ IDENTIFICATION*37X$ FACTOR*9X$ MAXIMUM*8X$ MINIMUM*8X$ RES
2CUTION*15,9X, A1, 5X, 5A1C, 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)
13 4 4 5 5 6 6 7 7 8 8 9 9 1
20 10 11 11 12/10X$0...0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....0....073
905 FORMAT(//20X$1 1 2 2 3)
13 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....0....5....0....5....0....5....0....5....0....5....0....5....0....5....0....5....0....5....0....5....0....5....0....0)
30 0....0....5....0....5....0....5....0....5....0....5....0....5....0....5....0....5....0....5....0....5....0....5....0....5....0....5....0....0)
IF(NSKIP.EQ.1) GO TO 20
DO 21 I=1, NSKIP
21 PRINT 903
905 FORMAT(1H )
20 CONTINUE
DO 3 J=1, NPT, K
30 5 L=4, NCHAR
5 PLIN(I-1)=1H
DO 4 J=1, NF
P=(PLUT(J,1)-PMIN(1))/PSOE(I)
4 P=P+3.5
73
APPENDIX H

IF(IP.LE.2) GO TO 4
IF(PLINE(IP).NE.1) 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(1A,124A)
IF(NSKIP.EQ.0) GO TO 3
V 14 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.,6*0.,7*0.,8*0.,9*0.,10*0.,11*0.,12*)
906 FORMAT(10X*0,5*0.,6*0.,7*0.,8*0.,9*0.,10*0.,11*0.,12*)
RETURN
END
SUBROUTINE FOURT(DATA,NN,NDIM,ISIGN,IFORM,WK)
THE COOLEY-TUKEY FAST FOURIER TRANSFORM IN USASI BASIC
TRANSFORM(K1,K2,...) = SUM(DATA(J1,J2,...)*EXP(ISIGN*2*PI*SQ
*((J1-1)*K1-1)/NN(1)+*(J2-1)*K2-1)/NN(2)*...)), SUMMED FOR
J1, K1 BETWEEN 1 AND NN(1); J2, K2 BETWEEN 1 AND NN(2); ETC.
THERE IS NO LIMIT TO THE NUMBER OF SUBSCRIPTS. DATA IS A
MULTIDIMENSIONAL COMPLEX ARRAY WHOSE REAL AND IMAGINARY
PARTS ARE ADJACENT IN STORAGE, SUCH AS FORTRAN IV PLACES THE
IF ALL IMAGINARY PARTS ARE ZERO (DATA ARE DISGUISED REAL), S
ELSE IFORM TO ZERO TO CUT THE RUNNING TIME BY UP TO FORTY PERCENT
OTHERWISE, IFORM = +1. THE LENGTHS OF ALL DIMENSIONS ARE
STORRED IN ARRAY NN, OF LENGTH NDIM. THEY MAY BE ANY POSITIVE
INTEGERS, THO THE PROGRAM RUNS FASTER ON COMPOSITE INTEGERS,
ESPECIALLY FAST ON NUMBERS RICH IN FACTORS OF TWO. ISIGN IS
OR -1. IF A -1 TRANSFORM IS FOLLOWED BY A +1 ONE OR A +1
BY A -1 THE ORIGINAL DATA REAPPEAR, MULTIPLIED BY NTOI (=NN
NN(2)*...). THE TRANSFORM VALUES ARE ALWAYS COMPLEX, AND ARE KE
IN ARRAY DATA, REPLACING THE INPUT. IN ADDITION, IF ALL
DIMENSIONS ARE NOT POWERS OF TWO, ARRAY WORK MUST BE SUPPLIE
COMPLEX OF LENGTH EQUAL TO THE LARGEST NON 2**K DIMENSION.
OTHERWISE, REPLACE WORK BY ZERO IN THE CALLING SEQUENCE.
NORMAL FORTRAN DATA ORDERING IS EXPECTED, FIRST SUBSCRIPT VA
FASTEST. ALL SUBSCRIPTS BEGIN AT ONE.
Running time is much shorter than the naive $NT^2$, being given by the following formula. Decompose $NT$ into $2^{*K2} * 3^{*K3} * 5^{*K5} * \ldots$. Let $\sum2 = 2^{*K2}$, $\sumF = 3^{*K3} + \ldots$ and $NF = K3 + K5 + \ldots$. The time taken by a multi-
dimensional transform on these $NT$ data is $T = TO + NTCT^2 (T_2 * \sum2 + T_3 * \sumF + T_4 * \sum4)$. On the CUC 33Cu (floating point add
of six microsecond), $T = 3000 + NTCT(1500 + 4.9 \sum2 + 68 \sumF + 320 \sum4)$ microseconds on complex data. In addition, the accuracy is greatly improved, as the rms relative error is bounded by $3*2^\left(-B\right)*\text{factor}(i**1.5)$, where $B$ is the number of bits in the floating point fraction and factor$(i)$ are the prime factors of $NT$.

The discrete Fourier transform places three restrictions upon
data.
1. The number of input data and the number of transform values
must be the same.
2. Both the input data and the transform values must be equally
spaced points in their respective domains of time and
frequency. Calling these spacings $\delta$T and $\delta$F, it must be
true that $\delta$F = $\frac{2\pi}{NN(1) \delta$T}. Of course, $\delta$T need
be the same for every dimension.
3. Conceptually at least, the input data and the transform
represent single cycles of periodic functions.

Example 1. Three-dimensional forward Fourier transform of a
complex array dimensioned 32 by 25 by 13 in FORTRAN IV.

```fortran
DIMENSION DATA(32,25,13),WORK(56),NN(3)
COMPLEX DATA
DATA NN/32,25,13/
DO 1 I=1,32
DO 1 J=1,25
DO 1 K=1,13
1 DATA(I,J,K)=COMPLEX VALUE
CALL FOURT(DATA,NN,3,-1,1,WORK)
```

Example 2. One-dimensional forward transform of a real array
length 64 in FORTRAN II.

```fortran
DIMENSION DATA(2,64)
DO 2 I=1,64
2 DATA(I,1)=REAL PART
2 DATA(I,1)=0.
CALL FOURT(DATA,64,1,-1,0,0)
```

```fortran
DIMENSION DATA(2),NN(1),IFACT(32),WORK(1)
WI=1.00
WR=1.00
WSTPF=1.00
WSPI=1.00
TWOP=5.293185307
IF(NDIM-1)920,1,1
```
APPENDIX H

1 NTOT=2
2 DO 1 IDIM=1,NDIM
3 IF(N(N-IDIM))920,920,2
4 NTOT=NTOT*N(IDIM)
5 C MAIN LOOP FOR EACH DIMENSION
6 C
7 NP1=2
8 DO 910 IDIM=1,NDIM
9 N=N*IDIM
10 NP2=NP1*N
11 IF(N-1)*920,920,5
12 C FACTOR N
13 C
14 M=N
15 NTWO=NP1
16 IF=1
17 IDIV=2
18 IQUOT=M/IDIV
19 IREM=M-IDIV*IQUOT
20 IF(IQUOT-10IV)5C,11,11
21 IF(IREM)=29,12,2C
22 NTWO=NTWO+NTWO
23 M=IQUOT
24 GO TO 1C
25 1C IDIV=3
26 IQUOT=M/IDIV
27 IREM=M-IDIV*IQUOT
28 IF(IQUOT-10IV)5C,31,31
29 IF(IREM)=32,4C
30 IFACT(IF)=IDIV
31 IF=IF+1
32 M=IQUOT
33 GO TO 3C
34 3C IDIV=IDIV+2
35 GO TO 30
36 30 IF(IREM)=51,6C
37 NTWO=NTWO+NTWO
38 GO TO 70
39 70 IFACT(IF)=M
40 C SEPARATE FOUR CASES--
41 C
42 C 1. COMPLEX TRANSFORM OR REAL TRANSFORM FOR THE 4TH, 5TH, E
43 C DIMENSIONS.
44 C 2. REAL TRANSFORM FOR THE 2ND OR 3RD DIMENSION. METHOD--
45 C TRANSFORM HALF THE DATA, SUPPLYING THE OTHER HALF BY C
46 C JUGATE SYMMETRY.
47 C 3. REAL TRANSFORM FOR THE 1ST DIMENSION, N ODD. METHOD--
48 C TRANSFORM HALF THE DATA AT EACH STAGE, SUPPLYING THE D
49 C HALF BY CONJUGATE SYMMETRY.
50 C
APPENDIX H

4. REAL TRANSFORM FOR THE 1ST DIMENSION, E EVEN. METHOD-
TRANSFORM A COMPLEX ARRAY OF LENGTH N/2 WHOSE REAL PART
ARE THE EVEN NUMBERED REAL VALUES AND WHOSE IMAGINARY
PART ARE THE ODD NUMBERED REAL VALUES. SEPARATE AND SUPPLY
THE SECOND HALF BY CONJUGATE SYMMETRY.

70 NUN2=NP1*(NP2/NTW0)
ICASE=1
IF(IDIM=1)71,90,90
IF,IDIM=2)72,72,90
ICASE=2
IF(IDIM=1)73,73,90
ICASE=3
IF(NTw0-NP1)9C,9C,74
ICASE=4
NTWO=NTW0/2
N=N/2
NP2=NP2/2
NTOT=NTOT/2
T=3
DO 80 J=2,NTOT
DATA(J)=DATA(I)
80 I=I+2
TIPNG=NP1
IF(ICASE=2)10C,65,10C
IF(ICASE=2)NP0*(1+NPPEV/2)
C
C SHUFFLE THE FACTORS OF TWO IN N. AS THE SHUFFLING
C CAN BE DONE BY SIMPLE INTERCHANGE, NO WORKING ARRAY IS NEEDED
C
100 IF(NTw0-NP1)65C,65C,110
110 NP2HF=NP2/2
J=1
DO 150 I2=1,NP2,WNUN2
IF(J=I2)120,130,130
120 I1MAX=I2+WNUN2-2
DO 125 I1=I2,I1MAX,2
DO 125 I3=I1,NTCT,NP2
J3=J+I3-12
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=WNUN2)150,14G,140
150 J=J+M
APPENDIX H

MAIN LOOP FOR FACTORS OF TWO. PERFORM FOURIER TRANSFORMS OF LENGTH FOUR, WITH ONE OF LENGTH TWO IF NEEDED. THE TWIDDLE FACTORS OF TWN=SIGN*SQRT(-1)*M/(4*MAX)). CHECK FOR SIGN=SQW AND REPEAT FOR W=ISIGN*SQRT(-1)*CONJUGATE(W).

NON2T=NON2+NON2
IPAR=N2M0/NP1
31C IF(IPAR-21350,33C,320
32C IPAR=IPAR/4
33C GO TO 310
34C DO 350 I=1,N1RNG,2
350 DO 340 J3=I1,N1CT,NON2T
K2=KI+NON2
TEMP=DATA(K2)
TEMP1=DATA(K2+1)
DATA(K2+1)=DATA(K1)-TEMP
DATA(K2+1)=DATA(K1+1)-TEMP1
DATA(K1)=DATA(K1)+TEMP
DATA(K1+1)=DATA(K1)+TEMP1
360 MMAX=NON2
370 IF(MMAX=NP2HF37C,600,606
380 LMAX=MAX0(NON2T,MMAX/2)
390 IF(MMAX=NON21405,405,380
400 THETA=-TWOPI*FLAT(NON2)/FLAT(4*MMAX)
410 IF(ISIGN)400,390,390
420 WR=COS(THETA)
430 HI=SIN(THETA)
440 DO 570 L=NON2,LMAX,NON2T
52C KL=KMIN,NTOT
530 K2=KMIN,NTOT
540 IF(MMAX=NON2142C,420,410
550 w2R=wR*WR-WI*WI
560 w2I=2.*WR*WI
570 w3R=W2R*WR-W2I*WI
580 w3I=W2R*WI+W2I*WR
590 DO 600 I=1,N1RNG,2
600 DO 620 J3=I1,N1CT,NON2T
KMIN=J3*IPAR
610 IF(MMAX=NON2143C,430,440
620 KMIN=J3
630 KDIF=IPAR*MAX
640 KSTEP=4*KDIF
650 DO 52C K1,KM1T,ATOT,KSTEP
660 IF(MMAX=NON2146C,46C,480
APPENDIX H

460 U1R=DATA(K1)+DATA(K2)
U1I=DATA(K1+1)+DATA(K2+1)
U2R=DATA(K3)+DATA(K4)
U2I=DATA(K3+1)+DATA(K4+1)
U3R=DATA(K1)-DATA(K2)
U3I=DATA(K1+1)-DATA(K2+1)
IF(ISIGN)470=75,475

470 U4R=DATA(K3+1)-DATA(K4+1)
U4I=DATA(K4)-DATA(K3)
GO TO 510

475 U4R=DATA(K4+1)-DATA(K3+1)
U4I=DATA(K3)-DATA(K4)
GO TO 510

480 T2R=W2R*DATA(K2)-W2I*DATA(K2+1)
T2I=W2R*DATA(K2+1)+W2I*DATA(K2)
T3R=WR*DATA(K3)-WI*DATA(K3+1)
T3I=WR*DATA(K3+1)+WI*DATA(K3)
T4R=W3R*DATA(K4)-W3I*DATA(K4+1)
T4I=W3R*DATA(K4+1)+W3I*DATA(K4)
U1R=DATA(K1)+T2R
U1I=DATA(K1+1)+T2I
U2R=T3R+T4R
U2I=T3I+T4I
U3R=DATA(K1)-T2R
U3I=DATA(K1+1)-T2I
IF(ISIGN)490=450,550,556

490 U4R=T3I-T4I
U4I=T4R-T3R
GO TO 510

500 U4R=WR-T3I-T4I
U4I=T3R-T4R

510 DATA(K1)=U1R+U2R
DATA(K1+1)=U1I+U2I
DATA(K2)=U3R+U4R
DATA(K2+1)=U3I+U4I
DATA(K3)=U1R-U2R
DATA(K3+1)=U1I-U2I
DATA(K4)=U3R-U4R

520 DATA(K4+1)=U3I-U4I
K4I=N=(KMIN-J3)+J3
KDIF=KSTEP
IF(KDIF=KMAX+2),530,530

530 CONTINUE
M=MMAX-M
IF(ISIGN)540=550,556,690

540 TEMPR=WR
WR=WI
WI=TEMPR
GO TO 560

550 TEMPR=WR
WR=WI

APPENDIX H

WI=TEMPPP

560 IF(M-LMAX)565,565,410
565 TEMPR=WR
605 IFP1=IFP1/IFACT(IF)
J1RNG=NP2
IF(ICASE=3)612,611,612
611 J1RNG=(NP2+IFP1)/2
J2STP=NP2/IFACT(IF)
J1RG2=(J2STP+IFP2)/2
612 J2MIN=1+IFP2
IF(IFP1-NP2)615,610,610
610 IFP2=IFP2/IFACT(IF)
J1RNG=NP2
IF(ICASE=3)612,611,612
611 J1RNG=(NP2+IFP1)/2
J2STP=NP2/IFACT(IF)
J1RG2=(J2STP+IFP2)/2
612 J2MIN=1+IFP2
615 DO 633 J2=J2MIN,IFP1,IFP2
THETA=-TWOP1*FLCAT(J2-1)/FLOAT(NP2)
IF(ISIGN)625,620,620
620 THETA=-THETA
625 SINTH=SIN(THETA/2.)
WSTPR=-2*SINTH*SINTH
WSTPI=SIN(THETA)
WR=WSTPR+1
WI=WSTPI
J1MIN=J2+IFP1
60 0 35 J1=J1MIN,J1RNG,IFP1
J1MAX=J1+1JRNG-2
60 0 630 J3=J3,NCTC,NP2
J3MAX=J3+IFP2-NP1
60 0 630 J3=J3,J3MAX,NP1
TEMPR=DATA(J3)
DATA(J3)=DATA(J3)+WR-DATA(J3+1)*WI
630 DATA(J3+1)=TEMPR*WI+DATA(J3+1)*WR
635 THETA=-TWOP1*FLOAT(IFACT(IF))
IF(ISIGN)645,645,645
640 THETA=-THETA
645
APPENDIX H

650 SINTH = SIN(THETA/2.)
651 WSTPR = -2.*SINTH*SINTH
652 WSTPI = SIN(THETA)
653 KSTEP = 2*N/IFACT(IF)
654 KRANG = KSTEP*(IFACT(IF)+1)
655 DO 698 I1 = 1, IIIRNG, 2
656 DO 699 I3 = 11, NTCT, NP2
657 DO 690 KTMIN = 1, KRANG, KSTEP
658 J1MAX = I3 + JIRNG - IFP1
659 J2MAX = J1 + IFP2 - NP1
660 J3MAX = J3 + IFP1 - IFP2
661 K = KTMIN+(J3-J1+(J1-I3)/IFACT(IF))/NP1IF
662 IF(KMIN-I)655,655,665
663 SUMR = 0.
664 SUMI = 0.
665 DO 660 J2 = J3, J2MAX, IFP2
666 SUMR = SUMR + DATA(J2)
667 SUMI = SUMI + DATA(J2+1)
668 WORK(K) = SUMR
669 WORK(K+1) = SUMI
670 GO TO 680
671 KCONJ = K + 2*(N - KTMIN+1)
672 J2 = J2MAX
673 SUMR = DATA(J2)
674 SUMI = DATA(J2+1)
675 OLDSR = 0.
676 OLDSI = 0.
677 J2 = J2-IFP2
678 TEMPR = SUMR
679 TEMPI = SUMI
680 SUMR = TWOWR * SUMR - OLDSR + DATA(J2)
681 SUMI = TWOWR * SUMI - OLDSI + DATA(J2+1)
682 OLDSR = TEMPR
683 OLDSI = TEMPI
684 J2 = J2-IFP2
685 IF(J2-J3)675,675,670
686 TEMPR = WR * SUMF - OLDSR + DATA(J2)
687 TEMPI = WI * SUMI
688 WORK(K) = TEMPR - TEMPI
689 WORK(KCONJ) = TEMPF - TEMPI
690 TEMPR = WR * SUMF - OLDSI + DATA(J2+1)
691 TEMPI = WI * SUMI
692 WORK(K+1) = TEMPR - TEMPI
693 WORK(KCONJ+1) = TEMPF - TEMPI
694 CONTINUE
695 IF(KMIN-I)685,685,686
696 WR = WSTPR + 1.
697 WI = WSTPI
698 GO TO 690
APPENDIX H

686 TEMP = WR
687 WR = WP * WSTPR - w1 * WSTPI + WR
688 w1 = TEMP * WSTPI + w1 * WSTPR + w1
690 TMWR = WP + WR
691 IF (ICASE = 3) $s2, 651, 692$
692 IF (IFP1 = NP1) 165, 642, 692
693 K = 1
694 IF (IFP1 = NP2) 69, 692
695 K = K + 2
696 GO TO 698

C
C COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N ODD, BY CG
C JUGATE SYMMETRIES AT EACH STAGE.
C
697 J3MAX = I3 + IFP2 - NP1
698 DO 697 J3 = I3, J3MAX, NP1
699 J2MAX = J3 + NP2 - J2STP
700 DO 697 J2 = J3, J2MAX, J2STP
701 J1MAX = J2 + J1RG - IFP2
702 J1CNJ = J3 + J2MAX + J2STP - J2
703 DO 697 J1 = J2, J1MAX, IFP2
704 K = 1 + J1 - I3
705 DATA (J1) = WORK (K)
706 DATA (J1 + 1) = WORK (K + 1)
707 IF (J1 - J2) 697, 696
708 DATA (J1CNJ) = WORK (K)
709 DATA (J1CNJ + 1) = - WORK (K + 1)
710 J1CNJ = J1CNJ - IFP2
711 CONTINUE
712 IF = IF + 1
713 IFP1 = IFP2
714 IF (IFP1 = NP1) 710, 712, 702

C
C COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N EVEN, BY C
C JUGATE SYMMETRIES.
C
715 GO TO (90C, 8GC, 8GC, 70C), ICASE
716 NHALF = N
717 N = N + N
718 THETA = - TWOPI / FLCAT (N)
719 IF (ISIGN) 703, 702
720 THETA = - THETA
721 STH = SIN (THETA / 2.)
722 WSTPR = - 2. * STH * SINH
723 WSTPI = SIN (THETA)
724 WR = WSTPP + 1.
725 WI = WSTPI
726 IMIN = 3
727
APPENDIX H

JMIN=2*NHALF-1
GO TO 725
710 J=JMIN
DO 720 I=IMIN,NTCT,NP2
SUMR=(DATA(I)+DATA(J))/2
SUMI=(DATA(I)+DATA(J))/2
DIFR=(DATA(I)-DATA(J))/2
DIFI=(DATA(I)-DATA(J))/2
TEMPR=WR*SUMI+WI*DIFR
TEMPI=WI*SUMI-WF*DIFR
DATA(I)=SUMR*TEMPR
DATA(J)=SUMR-TEMPP
DATA(I+1)=DIFI*TEMPI
DATA(J)=SUMR-TEMPP
DATA(J+1)=-DIFI*TEMPI
720 J=J+NP2
IMIN=IMIN+2
JMIN=JMIN-2
TEMPR=WP
WP=WR*WSTPR-WI*WSTPI+WK
WI=TEMPR*WSTPI+HI*WSTPP+WI
725 IF(1 SIGN)731,74C,750
730 IF(1 SIGN)731,74C,750
731 DO 735 I=IMIN,NTCT,NP2
735 DATA(I+1)=-DATA(I+1)
740 NP2=NP2+NP2
NTOT=NTOT+NTOT
J=NTOT+1
745 IMIN=IMAX-2*NHALF
I=IMIN
GO TO 755
750 DATA(J)=DATA(I)
DATA(J+1)=-DATA(I+1)
755 I=I+2
J=J-2
IF(I=IMAX)750,76C,760
760 DATA(J)=DATA(I+1)-DATA(I)
DATA(J+1)=C
IF(I-J)770,78C,780
765 DATA(J)=DATA(I)
DATA(J+1)=DATA(I+1)
770 I=I-2
J=J-2
IF(I=IMIN)775,77C,765
775 DATA(J)=DATA(I+1)-DATA(I+1)
DATA(J+1)=0
IMAX=IMIN
GO TO 745
780 DATA(I)=DATA(I)+DATA(2)
DATA(2)=0
GO TO 900
APPENDIX H

C COMPLETE A REAL TRANSFORM FOR THE 2ND OR 3RD DIMENSION BY
C CONJUGATE SYMMETRIES.

800 IF(I1P+NP1) 8C5, 900, 900
805 DO 860 I3=1,NTCT,NP1
I2MAX=I3+NP2-NP1
DO 860 I2=I3,I2MAX+NP1
I3=1,NTCT,NP1
810 JMAX=2*I3+NP1-1
IF(I2-I3) 320, 820, 810
820 IF(I1DIM-2) 850, 850, 830
830 J=JMAX+NP1
DO 840 I=IMIN,IMAX+2
DATA(I)=DATA(J)
840 J=J-2
850 IF,IDIM,2) 850, 850, 830
DATA(I)=DATA(J)
860 J=JMAX
C END OF LOOP ON EACH DIMENSION
900 NPO=NPI
910 NPREV=N
920 RETURN
END
OVERLAY (PATS, 1, C)

PROGRAM PPAIM
.COMMON/BLK1/STAPTT,ITFMT,NBLK,IPOW2,NCH,NPRINT,IPLUTA,IPLUTC,OFFSC
AL,DELTAT,SN,NRSKIP,LAP,NCROSS,ICROSS,200,NCHP,YLABEL(2),IWINDOW
1,F1,F2,ITYPESP,MO,NSKIP,IFF,LAG1,LAG2
.COMMON/BLK2/ICHI(14),CHSUM(14),NOFF(14),CHSUMSQ(14),SIGMA(14),RMS(14)
. ,MEAN(14),SCALFAC(14),CHSUM(14),TRACK(14),ICCHAN(14),IFILTER(14)
.COMMON/BLK3/NPT,TMAX,NPTO2,SNPCT,DELT,N64,NPTO12
1,INZERO,NREAD,NPTOT
.COMMON/BLK4/NMAX,NCHMAX,N8CMA
.COMMON/BLK5/PCTC,NBINS,DMAX(14),DMIN(14),OBIN(14),BINS(1),
1,CHISQC
.COMMON/BLK8/IAUTOSP(14),IAUTOCQ(14),ICRSP(20),ICRCUR(20),ITRA(20)
1,ICOH(20)
.COMMON/BLK9/NFILTF,FREQF(50),WQHTF(50)
.NAMELIST/INPUT/ITFMT,NSKIP,SN,DELTAT,STARTT,OFFSCAL,NCH,
1,SCALFAC,NPTOT,NREAD,IAUTOSP,IAUTOCQ,NCROSS,ICROSS,ICRSP,ICRCQK,
2ITRA,ICOH,LAP,WINDOW,ITYPESP,NPRINT,IPLUTA,IPULTC,FI,F2,LAG1,LAG2
3 ,PCTC,NBINS,DMAX,DMIN,INZERO,NFILTF,FREQF,WQHTF,IFILTER
APPENDIX H

DATA NFFILTF,PPEQF,WGHTF,IFILTER/0,100*G,14*0/
DATA IAOUDP,S,IALTOC,IORSP,ICRUCN,ITKA,ICOH/108*0/
DATA IF,LAG1,LAG2/1,0,0/
DATA NFKIP/0/
DATA ITFMT,STARTT,ICH,NRSKIP,SCALFAC,OFFSCAL,LAP,NCROSS,ICROSS,
INPRINT,PLOTAT,PLOTCT,WINDOW,FL,FZ,ITYPESF/
500 CIFMT,STARTT,ICH,NRSKIP,SCALFAC,OFFSCAL,LAP,NCROSS,ICROSS,
INPRINT,PLOTAT,PLOTCT,WINDOW,FL,FZ,ITYPESF/
500 1 CIFMT,STARTT,ICH,NRSKIP,SCALFAC,OFFSCAL,LAP,NCROSS,ICROSS,
INPRINT,PLOTAT,PLOTCT,WINDOW,FL,FZ,ITYPESF/
22,0,15*G,14*1,1*E+6,42*0,100,1,1,0,20000,2/
DATA I0,CHAN/2HID,4HID/2HID/2HID/2HID/2HID/
DATA I0,ZPOS,NOINS,OMAX,OAM/2*0,28*0/.
READ AND PRINT NAMELIST INPUT

C C READ INPUT
IF(EDF,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(8AI)
PRINT 5001, YLABEL
PRINT 5001, (I,TRACK(I),I=1,NCH)
5003 FORMAT(/ CASE ID*5X1H*2A10, 1H*/)
5003 FORMAT(/ CASE ID*5X1H*2A10, 1H*/)
PRINT 5004
5004 FORMAT(*1*)
C C CHECK INPUT DATA, PRINT ERROR MESSAGES
C NPT=NREAD
NBLK=NPTOT/NPT
IF(NCH.LE.NCHMAX) G10 TO 101
PRINT 103, NCHMAX
STOP 103
103 FORMAT(/** NCH GT*13,*, PROGRAM WILL NOT READ TAPE CORRECTLY. JOB
1 TERMINATED*)
101 CONTINUE
IF(NFSKIP) 310,310,311
311 DD 312 IFSKIP=1,NFSKIP
313 READ(1) SKIPREC
IF(EDF,1) 312,313
312 CONTINUE & PRINT 904, NFSKIP
904 FORMAT(/6H * *,15,20H FILES SKIPPED) * (*
310 IF(ITYPESF.GT.3) GO TO 9C
NCROSS=0
DO 91 I=1,NCH
IF(IAUTOCT(I).GT.0) IAUTOCT(I)=0
91 CONTINUE
APPENDIX H

PRINT 92
92 FORMAT(/ AMPLITUDE SPECTRUM OPTION CHosen/* ONLY AUTO SPECTRUM
1WILL BE CALCULATED*)
90 CONTINUE
ICORR=0
DO 116 I=1,NCH
   IF(AUTOCO(I).EQ.0) GO TO 116
   ICORR=1
   IF(AUTOSP(I).EQ.0) IAUTOSP(I)=-1
116 CONTINUE
IF(NCROSS) 118,116,108.
108 DO 110 I=1,NCROSS
   J1=ICROSS(I,1) $ J2=ICROSS(I,2)
   IF(AUTOSP(J1).EQ.0) IAUTOSP(J1)=-1
   IF(AUTOSP(J2).EQ.0) IAUTOSP(J2)=-1
   IF(ICRCOR(I).NE.0) GO TO 117
   IF(UTCRI(I).NE.0) GO TO 211
   IF(UTCRI(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(ICRSP(EQ.1.AND.INZERO.EQ.0) PRINT 907
907 FORMAT( // * * * YOU MAY HAVE CIRCULAR ERROR IN YOUR CORRELATION
INS 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*17)
109 DO 35 IK=1,NCH
   NOFF(IK)=0
   CHSUM(IK)=CHSUMSQ(IK)=0.
   CHSUML(IK)=0.
35 CONTINUE
35 CONTINUE
   IPOW2=0 $ NTEMP=NPT
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 203
   PRINT 206 $ STOP 207
201 PRINT 204
204 FORMAT( // * BLOCK SIZE NOT A POWER OF TWO. SLOW FOURIER TRANSFORM W
1LL BE USED*)
   IF(NPT.LE.NMAX) GO TO 205
   PRINT 206
206 FORMAT( // * BLOCK SIZE TOO LARGE FOR DIMENSIONS PROVIDED. JOB TERMI
INATED*)
APPENDIX H

STOP 20b
205 CONTINUE
NPTO2=NPT/2.
NSPCT=NPLOT=NPTC2 & IF(NPRINT.GT.NPT) NPRINT=NPTO2
TMAX=NPT*DELTAT
DELFL=T4AX
NPTO128=NPT/128
IF(INZERG.EQ.0).OR.(LAP.EQ.C) GO TO 111
PRINT 902
LAP=0
932 FORMAT(// "%15.50 PERCENT OVERLAP ON ZERO INSERTION RUNS*/" INPUT
1DATA ALTERED, LAP=0 *)
111 CONTINUE
IF(LAP.EQ.0) NBLK=2*BBLK-1
NCHP=0
DO 80 I=1,NCH
IF(IAUTOSP(I)) GO TO 81
NCHP=NCHP+1
ICHAN(NCH)=1
80 CONTINUE
IF(NCHP) 32,32,33
82 PRINT .84
84 FORMAT(//* INPUT INQUIRES NO CHANNELS TO BE PROCESSED, CASE ENDED
1 DATA ALTERED, LAP=0*)
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) =", /* NO. OF YL
10CKS (NBLK) =", /* NCHP * NBLK GREATER THAN NBCMAX=, 15/" 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) NOCF=2*NBLK & IF(LAP.EQ.0) NDUF=1.6364*(NBLK-1)
CALL CSQ((10G.*PCTC)/?.,NDUF,BU,BL,ICODE)
BL=100.*/BL & BU=100.*/BU
PRINT 903, PCTC,.B.,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
3RCEENT AND*F5.0,* PERCENT OF THE TRUE SMOOTHED SPECTRUM")
C
C COMPUTE CRITICAL VALUE OF CHISQURE FOR NORMALITY TEST
C
IF(INBINS.GT.100) NRBINS=100
IF(NRBINS.F0.0) GO TO 111
DO 112 I=1,NCHP

APPENDIX H

\[ K = ICHAN(I) \]
\[ UBI(R,J) = (OMAX(K) - OMIN(K)) / NBINS \]
\[ UO 112 J = 1, NBINS \]
112 RINS(I,J) = 0.
\[ NBINS3 = NBINS - 3 \]
CALL CSQ(PCTC, NBINS3, BL, BU, ICODE)
CHISQC = NBINS3 / BL
113 CONTINUE
IF(NFBTP(EQ, 0)) GO TO 114
IF(NFILTP(EQ, 5C)) GO TO 115
PRINT 905
905 FORMAT(/55H ** ** INPUT ERROR, NFILTP GT 50, EXECUTION ENDED **
905 1 *)
STOP 96
115 PRINT 906
906 FORMAT(/126H ** ** SPECTRAL FILTERING OPTION SELECTED. CORRELATION
906 1 INS WILL BE CALCULATED FROM FILTERED SPECTRA FOR CHANNELS SPECIFIED
906 2. ** **)
114 CONTINUE
C
C COMPUTE DATA WINDOW CORRECTION FACTOR FOR SPECTRA
C
\[ CAPT = 1/RAD*DELTAT - [IWINO1 = IWINDOW + 1 \]
GO TO (300, 301, 302, 303), IWINO1
300 CONTINUE
WCON = CAPT $ GO TO 304
301 CONTINUE
WCON = CAPT * .37 $ GO TO 304
302 CONTINUE
WCON = CAPT * 713672702 $ GO TO 304
303 CONTINUE
WCON = CAPT * 269649571
304 CONTINUE
C
C CORRECT PLOT LIMITS
C
IF(F1 LT 0.) F1 = 0.
FMAX = NPT02 * DFLF
IF(F2 GT FMAX) F2 = FMAX
IF(LAG1 LT NSPCT) LAG1 = NSPCT
IF(LAG2 GE NSPCT) LAG2 = NSPCT - 1
IF(LAG2 GT LAG1) GO TO 305
L1 = LAG2; LAG2 = LAG1 + LAG1 = L1
305 CONTINUE
RETURN
END
SUBROUTINE CSQ(P, N, BL, BU, ICODE)
EXTERNAL FUNC
COMMON/PLOF/A, C
A = N/2.
Q = P/100.
C
FIND UPPER BOUND
CALL IT21(CHISQ, C01, LOQC, L0, FUNC, 1.E-6, 1.E-6, 100, ICODE)
IF(ICODE = 3) 1, 1, 2
2 AQ = 1.E+12
APPENDIX H

GO TO 3
1 BU=N/CHISO
3 Q=1.-Q
C FIND LOWER BOUND
CALL ITP2(CHISO,0.01,1000.,1.,FUNCTION,E-6,E-8,11C,ICODE)
IF(ICODE=3) 4,4,5
5 BL=0.
GO TO 6
4 BL=N/CHISO
6 RETURN
END
FUNCTION FUNCTION(CHISQ)
COMMON/PROBF/A,C
DIMENSION H60( 15)
DATA H6C/-.01 1.8 ,-.0067, -. 0033 ,-.0010, .GCC i, 2 *.GGC6, .00*2 ,-.OC03,
1-. 0006, -. OOC5, .COG2, .0017, .0043, .0062/ .
IFU-15) 1,1,2 . .. . • .
1 X=CHISO/2.
FUN=Q-GAMMF(A,X)/GAMMF(A,0.)
GO TO 3
2 X=(CHISQ/ (2.*A))**(1./3. )-(-1.**X(A))/SQRT(1./(8.*A))
IX=X/.5 $ IF((IX*5.GT.X) I X = I X-l . i I X = I X + 8.3. IF(IX) 0,0,7
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)*(H6C(IX)*XREM*(H60(IX)-H60(IX))/.5)
GO TO 11
6 HNU=0.
11 CONTINUE
XREM=X-(IX-8)*.5
IF(XREM) 4,5,5
4 FUNC=Q-(1.+PMINUS1)
GO TO 3
5 FUNC=Q+PMINUS1
3 RETURN
END
OVERLAY(PATS,2»G)
PROGRAM BLOCKS
COMMON/CMAIN( 1, -
COMMON/BLK1/STARTT,ITFMT,NBLK,IPCH2,NC4,NPRINT,IPLOTA,IPLOTG,UFFSC
N3L,DELTAT,SN,RSKIP,LAP,NCROSS,SCROSS(2,20),NCMP,PLABEL(2),IXINOW
1,FL,FZ,ITYPEPES,NNN,Fских,IFF,LAG1,LAG2
COMMON/BLK2/ICH(14),CHSUM(L14),NFF(14),CHSUMSQ(14),SIGMA(14),RMS(L
14),MEAN(14),SCALFAC(14),CHSUML(14),TRACK(14),ICHAN(14)
APPENDIX H

COMMON/BLK3/NPT,TMAX,NPT02,NSPCT,DELF,N64,NPT0120
1,INZERO,NREAD
COMMON/BLK5/PCTC,NAINS,O#AX(14),OMIN(14),DBIN(14),BIN(100,14)
1,CHISQC
COMMON/BLK6/ISAVE64,IRI,IPLOT,IOATA,IZ,ISPECT
COMMON/BLK7/I(14)
COMMON/BLK8/IAUTOSP(14),IAUTOCL(14),ICRSP(26),ICRCOR(26),ITRA(20)
1,ICOH(20)
DO 1 NH=1,NBLK
CALL READPE(NB,CMAIN(I0ATA),CMAIN(ISAVE64))
CALL TRAN(NB,CMAIN(IZ),CMAIN(ISAVE64),CMAIN(ISPECT))
1 CONTINUE
RETURN
END

SUBROUTINE READPE(NB,DATA,SAVE64)
DIMENSION DATA(U,SAVE64(II
COMMON/BLK1/STARTT,ITFMT,NBLK,FOW2,NPRINT,IPHTA,PLU,C,OFFSC
1AL,DELTAT,SN,NRSKIP,LAP,NXROSS,ICROSS(2,20),NCHP,YLABEL(2),IWINDOW
1,F1,F2),ITYPESP,WCON,NFSKIP,IFF,LAG1,LAG2
COMMON/BLK2/ICH(14),CHSUM(14),NFF(N14),CHSUMSQ(14),SIGMA(14),RMS(1
14),MEAN(N14),SCALFAC(14),CHSUM(14),TRACK(14),ICHAN(14)
COMMON/BLK3/NPT,TMAX,NPT02,NSPCT,DELF,N64,NPT0120
1,INZERO,NREAD
COMMON/BLK7/NN,IW0,KCH,NFR,II,KK,N1,JI,KEC,NREC,LOC
IF(ITFMT.EQ.3) GO TO 4000
IF(NB-1) 12,12,11
12 GO TO (3003,301C),ITFMT
C ATRAN FORMAT PARAMETERS SET UP
3001 IF(NRSKIP) 105,105,106
106 DD 107 I=1,NRSKIP
107 READ(1)
105 CONTINUE
IF(NCH.GT.10) GO TO 3002
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
APPENDIX H

GO TO 3003
3007 PRINT 3008
3008 FORMAT(/** NCH GT 100 NCT 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(NSKIP) 3009,3009,108
108 DO 109 I=1,NSKIP
109 READ(1)
C C FIND STARTING TIME ON TAPE
C
3009 CONTINUE
GO TO (3013,3014), IFMT
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 3016
3014 READ(1) KEY,NN,(DATA(I),I=1,NN)
IF(EOF,1) 3,3015
3015 DO 1 J=1,NFR
I J=(J-1)*KCH+IWD+1
JJ=J
IF(DATA(I)-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 NI=JJ+1 $ KK=0
IF(NI,GTE,NFR) GO TO 15
BACKSPACE 1
GO TO 16
15 NIH (!=IWD+1

91
APPENDIX H

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=II-1
   IREC=IREC1-1
   GO TO 14
13 IREC=0
14 ILOC=0
10 CONTINUE
   DO 5 J=N1, NFR
      KK=KK+1
   6 ILOC=ILOC+1
      IJ=(ILOC-1)*NCHP
      DO 30 IKK=1, NCHP
         IK=ICHAN(IKK)
         X=DATA(IJ+IKK)*SCALFAC(IK)
         IF (ABS(X)-OFFSCAL) 33, 34, 34
   33 CHSUM(IKK)=CHSUM(IKK)+X
      SAVE64(IJ+IKK)=X
      GO TO 30
   34 SAVE64(IJ+IKK)=CHSUM(IKK)/(KK+(NB-1)*NREAD)
      CHSUM(IKK)=CHSUM(IKK)*((KK*(NB-1)*NREAD)+(KK*(NB-1)*NREAD)+1.)
      NOFF(IKK)=NOFF(IKK)+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
   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, EXECUTION ENDED/)
3064 CONTINUE
   NL=1
   I=IWD+1
   GO TO 10
7 CONTINUE
C ONE BLOCK READ, READY TO BE PROCESSED BY FFT
C
APPENDIX H

IF(LOC.EQ.0) RETURN
IREC=IREC+1
CALL WRITMS(9,SAVE64,LOC*NCHP,IREC)
RETURN

C
C RECIN FORMAT
C

4000 NREC=NREAD/64 $ NCHP2=NCHP $ 11=1
I=0
IF(NH-1) 4001,4001,3029
4001 IF(NSKIP.EQ.0) GO TO 3050
DO 3051 IRSKIP=1,NSKIP
CALL RECIN(1,2,NN,DATA,1,NCHP,1)
PRINT 3052,NSKIP
3052 FORMAT(/110,* RECORDS SKIPPED*)
3050 CONTINUE
I=I+1
CALL RECIN(1,2,NN,DATA,1,NCHP,1)
IF(EOF,1) 3,3027
3027 IF(DATA(2)-START) 3050,3022,3022
3022 IF(LDATA(1).EQ.SN) GO TO 3035
PRINT 3016
STOP 05
3035 I1=2
PRINT 901, I
901 FORMAT(/* STARTT FOUND AT RECORD*I*)
KK=1
DO 3023 IKK=1,NCHP
IJ=1CHAN(IKK)
XX=DATA(IJ)+SCALFAC(IJ)
IF(ABS(XX)-OFFSCL) 3030,3031,3031
3030 CHSUM(IJ)=CHSUM(IJ)+XX
3031 XX=0. $ NUFF(IJ)=NOFF(IJ)+1
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
WRITE=C
IREC=NPT02/64
NREC=NREAD/64
ILOC=(NPT02-IREC*64)/NCHP
IREC=IREC+1 $ NWORDS=N04 $ NLEFT=N04-ILOC
4018 IF(IREC-NREC) 4016,4016,4017
4017 NWORDS=(NREAD-NREC*64)/NCHP
NLEFT=WORDS-ILOC
4016 CALL READMS(9,SAVE64,NWORDS,IREC)
IREC=IREC+1
KK=KK+NLEFT/NCHP
IF(LOC.EQ.0) GO TO 4014
APPENDIX H

DU 4006 I=1,NLEFT
4006 SAVE64(I)=SAVE64(I+ILOC)

IF(KK-NPT02) 4007,4012,4013
4007 IF(IREC-NREC) 4CCV,4009,4010
4010 NWORDS=(NREAD-NREC*64)*NCHP
IF(NWORDS.LT.ILCC) ILOC=NWORDS
4009 CALL READMS(9,SAVE64(NLEFT+1),ILOC,IREC)
KK=KK+ILOC/NCHP
4014 IF(KK-NPT02) 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 IREC=IWRITE+1
4013 I1=NLEFT/NCHP-(KK-NPT02)+1
GO TO 3060
4019 I1=NLEFT/NCHP+1
GO TO 4015
4020 IF(ILOC-NLEFT-I+1) GO TO 4022
I1=(ILOC-NLEFT)/NCHP+1
GO TO 4015
4021 I1=(NLEFT+ILOC)/NCHP-(KK-NPT02)+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+1
CALL RECIN(1,2,NN,DATA,1,NCHP,1)
IF(EOF.1) 3063,3034
3034 KK=KK+1
I1=(ILOC-I)*NCHP
DO 3026 IKK=1,NCHP
IK=ICHAN(IKK)
XX=DATA(IK+2)*SCALFAC(IK)
IF(ABS(XX)-OFFSCAL) 3032,3033,3035
3033 XX=CHSUM(IXK)/(KK-I+NB-L1)*NREAD)
NOFF(IK)=NOFF(IK)+1
3032 CHSUM(IK)=CHSUM(IK)+XX
3026 SAVE64(I+IKK)=XX
3025 CONTINUE
I1=1
3024 CALL WRITMS(9,SAVE64,N64,IREC)
IF(NREC*64.EQ.NREAD) RETURN
3046 NLEFT=NREAD-64*NRLC
NREC=NREC+1
DO 3042 ILOC=I1,NLEFT
APPENDIX H

CALL RECIN(1,2,NN,DATA,1,NCHP2,1)
IF(EOF,1) 3045,3041
3041 KK=KK+1
J=ICHAN(JJ)
DO 3042 IKK=1,NCHP
IK=ICHAN(IKK)
XX=DATA(IK+2)
IF(XX.NE.0) 3043,3044
3044 XX=CHSUM(IK)/(KK-1+N8-1)*NREAD
NOFF(IK)=N0FF(IK)+1
3043 CHSUM(IK)=CHSUM(IK+XX)
3042 SAVE64(IJ+IKK)=XX
CALL WRTMS(9,SAVE64,NLEFT*NCHP,NREC)
RETURN
END

SUBROUTINE TRAN(NH,Z,SAVE64 SPEC)
DIMENSION Z(1),SAVE64(1),SPEC(1)
COMPLEX Z
COMMON/BLK1/START,ITFMT,NBLK,IPOW,2,NC,HP,DATA,1,NN,KPRINT,IPLOTA,PLUTC,OFFSC
IA,DELTA,T,SN,NSKIP,LP,N,CR,12,12,NCHP,YLABEL(2),WINDOW
,sf,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
COMMON/BLK2/ICH(14),CHSUM(14),NGFF(14),CHSUMW(14),SIGMA(14),RMS(14)
,MEAN(14),SCALEFACT(14),CHSUM(14),TRACK(14),ICHAN(14)
COMMON/BLK3/NPT,TPx,TP02,NSPCT,DELFA,NE64,NPT0128
,1,NZERO,NREAD
COMMON/BLK2/PCTC,NBINS,DMAX(14),DMIN(14),DBIN(14),R0INS(100,14)
DO 19 JJ=1,NCHP
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 I=1,64
IK=I-64+1
IL=(IJ-1)*NCHP+JJ
20 Z(IIK)=SAVE64(IJ)
IF(NREC.EQ.64) 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+IKK)=SAVE64(IJ)
10 CONTINUE
C COMPUTE BLOCK MEAN AND COUNTS FOR HISTOGRAM
**APPENDIX H**

```fortran
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
DMIN = DMIN(J) & DEL = IN(J)
DO 41 I = 1, NREAD
IBIN = (Z(I) - DMIN) / DEL + 1
IF(IBIN.LT.1) IBIN = 1
IF(IBIN.GT.NBINS) IBIN = NBINS
41 BINS(IBIN,J) = BINS(IBIN,J) + 1
40 CONTINUE
DO 23 I = 1, NREAD
23 Z(I) = Z(I) - BLKMEAN

C APPLY DATA WINDOW
C IF(IWINDOW.EQ.0) GO TO 33
30 CALL HANNING(Z, NREAD)
GO TO 33
31 CALL HANNING(Z, NREAD)
GO TO 33
32 CALL PAPZL(Z, NREAD)
33 CONTINUE

C INSERT ZEROS
C IF(INZERO.EQ.0) GO TO 25
35 Z(I*NPT02) = 0.
35 CONTINUE

C COMPUTE FOURIER TRANSFORM AND STORE ON RANDOM ACCESS FILE
C 24 CALL FOUTH(Z, NPT, 1, 1, 0, SPECT)
I = (NB-1)*NCHP*JJ
CALL WRITMS(I, Z, NPT, IJ)
19 CONTINUE
RETURN
END
SUBROUTINE HANNING(Z, NPT)
COMPLEX Z(I)
DATA PI/3.1415926535898/
DO 1 I = 1, NPT
D = 5*(1. - COS(Z(I)*PI*(I-1.)/NPT))
1 Z(I) = Z(I) * D
RETURN
END
```
APPENDIX H

SUBROUTINE HAMMING(Z, NPT)
COMPLEX Z(I)
PARAMETER (PI = 3.14159265358979)
PION = PI / NPT
N2 = NPT + 2
DO 1 I = 1, NPT
T = (I + 1 - N2) * PION
D = 0.54 + 0.46 * COS(T)
1 Z(I) = Z(I) * D
RETURN
END

SUBROUTINE PARZEMZ(NPT)
COMPLEX Z(I)
TM = NPT / 2
TMM1 = TM - 1
TM02 = TM / 2.
DO 1 I = 1, NPT
J = I - TMM1
IJ = I * ABS(J)
IF (IJ - TM02) LT 2,2,3
2 U = 1. - 6. * J * J * (1. - T(J))
GO TO 1
3 0 = 2. * (1. - J) ** 3
1 Z(I) = Z(I) * 0
RETURN
END

OVERLAY(PATS, 3, C)
PROGRAM AUTOSP
COMMON CMAIN(1)
COMMON/BLK1/STARTT, ITFTM, NBLK, IPON2, NC4, NPRINT, IPLOTA, IPLOTB, ONSC
I4, DELTAT, SN, NRSKIP, LAP, NCRASS, ICROSS(2, 20), NCHP, YLABEL(2), IWINDOW
1, E1, F2, ITYPESP, WCCN, NRSKIP, IFF, LAG1, LAG2
COMMON/BLK2/ICH(14), CHSUM(14), NFF(14), CHSUMW(14), SIGMA(14), RMS(14)
MEAN(14), SCALFAC(14), CHSUM1(14), TRACK(14), ICHAN(14)
COMMON/BLK3/NPT, TMAX, NPT02, NISPCT, DELF, No4, NPT0128
1, INZERO, NREAD
COMMON/BLK5/PCTC, NBI, DMAX(14), UMIN(14), DMIN(14), BINS(10), 14
COMMON/BLK6/ISAVE64, 1, IXPLOT, 1 DATA, IZ, ISHUNT
COMMON/BLK8/IALTSPI(14), IAUTO(14), ICROSS(20), ICACOR(20), ITRA(20),
ICROM(20)
COMMON/BLK9/NFILT, FREQF(50), WGFNT(50)

C
COMPUTE AND PRINT SPECTRAL FILTER

IF (NFILT, LE, 0) GO TO 1
CALL SPLINE(FREQF, WGFNT, NFILT, NISPCT, CMAIN(I1), DELF)
NREC = NBLK * NCHP + 1
CALL WRITEMS(8, CMAIN(I1), NISPCT, NREC)
DO 2 I = 1, NISPCT
2 CMAIN(IXPLOP + I - 1) = (I - 1) * DELF
I1 = IXPLOT - 1 $ I2 = IRI - 1
PRINT 900, (I, CMAIN(I1 + I), CMAIN(I2 + I), I = 1, NISPCT)
900 FORMAT(6/1ISPETRAL FILTER WLOGING FNCATION*/ 5X*I*3X*FREQUENCY* 10X*WEIGHT*3(7X*I*3X*FREQUENCY*6X*WEIGHT*)/(10, 2E13, 5, 10, 2E13, 5, 10, 13)
APPENDIX H

22E13.5,16,2E13.5))
1 CONTINUE
CALL AUTO(CMAIN(I2),CMAIN(ISPECT),CMAIN(IIR),CMAIN(IXPLOT))
C CALL AUTO FUNCTION ROUTINE WITH COMPUTED BLOCK ADDRESSES
C IF(NININS.GT.0) CALL NORMAL
RETURN
END
SUBROUTINE AUTO (I,SPECT,R1,XPLOT)
DIMENSION SPECT(1),XPLOT(1),R(1),Z(1)
COMPLEX Z,SPECT
COMMON/BLK1/STARTT,ITFMT,NBLK,IPON2,NC4,NPRINT,IPLUTA,IPLOT,C,OFFSC
1AL,DELTAT,SN,NRSKIP,LAP,NCROSS,ICROSS(2,20),NCHP,YLABEL(2),IWINOW
1,F1,F2,ITYPESP,NCUN,NFSKIP,IFF,LAG1,LAG2
COMMON/BLK2/ICH(14),CHSUM(14),NUFF(14),CHSUMSQ(14),SIGMA(14),RMS(1
14),MEAN(14),SCALE(14),CHSUM1(14),TRACK(14),ICHAN(14),IFILTE(14)
COMMON/BLK3/NPT,TMAX,NPT,N28
COMMON/BLK4/NPT0128,
CALL ELmal,SN,NRSKIP,LAP,ACROSS,ICROSS(2,20),MCHP,YLABEL(2),IWNUQW
i,Fl,F2,ITYPESP,NCUN,NFSKIP,IFF,LAG1,LAG2
COMMON/BLK6/IATOSP(14),IATOCO(14),ICRSP(20),ICRCOR(20),ITKA(20),
ICOH(20)
COMMON/BLK9/NFLTP
COMMON/BLK10/NRCRO/)
DIMENSION PLABEL(6),PPLUT(27),BAND(27),IUSDIC
REAL MEAN
PRINT 914, (TRACK(J),NOFF(J),J=1,NCH)
912 FORMAT(914 NO. OF OFF-SCALE VALUES FOR EACH CHANNEL*/(1X,A10,'10))
C COMPUTE MEAN AND VARIANCE OF EACH CHANNEL
C
00 36 IKK=1,NCHP
36 MEAN(IK)=CHSUM(IK)/(NBLK*NREAD)
SECMM=CHSUMSQ(IK)/(NBLK*NREAD)
SIGMAM=(SECMM-MEAN(IK)**2 )
IF(SIGMASO> 30,37,37
38 SIGMAUK ) = SQRT(SIGMASO)
36 CONTINUE
C START OF LEOP FOR COMPUTING AUTO SPECTRA AND CORRELATIONS
C
00 50 JJ=1,NCHP
50 J=ICHAN(JJ)
DO 53 I=1,NSPCT
53 SPECT(I)=0.
C AVERAGE NBLK SPECTRA FOR ONE CHANNEL

98
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
81 DO 82 I=1,NSPCT
   ZL=Z(I)
   AMP=ABS(ZL)**2*ARG=0.
   IF (AMP.0) ARG=ATAN2(REAL(ZL),IMAG(ZL))
82 SPEC(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)
50 DO 85 I=1,NSPCT
   SPECT(I)=SPECT(I)*CON
   AMP=REAL(SPECT(I))*CON
   ARG=ATAN2(IMAG(SPECT(I)),REAL(SPECT(I)))*57.2957795/NBLK
   SPECT(I)=CMPLX(AMP,ARG)
85 CONTINUE
RMS(J)=0.
GO TO 86
83 CON=DELTAT*DELTAT/(6.283185308*KCON*NBLK)
DO 59 I=1,NSPCT
   SPECT(I)=SPECT(I)*CON
   SUM=SUM-SPECT(I)
59 CONTINUE
RMS(J)=SQRT(SUM*CON*12.5563762)
86 CONTINUE
C
C APPLY SPECTRAL FILTER
IF(INFILTP.EQ.3) CALL FILTER(J)*CON GO TO 58
87 CONTINUE
112 SPECT(I)=SPECT(I)*R(1)
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(3X*CHANNELS=13,3X*MEAN=E12.5,3X*SIGMA=*
       E12.5,3X*POWERS)
1002 FORMAT(8X*I*3X*FREQUENCY*6X*POWER*)
IF(ITYPESP.EQ.3) GO TO 87
DO 60 I=1,NSPCT
   XPLT(I)=Y(1-I)*DELT
   AMP=SPECT(I)**2.*
   SPECT(I)=R(I)*AMP
60 CONTINUE
911 FORMAT(4(16,ZE13.5))
```
APPENDIX H

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=*9.2, *)*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*5X, A10, 20X)
NPLABEL=27
PRINT 1006, (PLABEL(I), I=1,3)

1005 FORMAT(//20X7N *** ,5A10,6H ***)
1006 FORMAT(//20X7H *** ,3A10,6H ***)
GO TO 68

68 DO 88 I=1, NSPCT
XPLOT(I)=(I-1)*DELF
88 RI(I)=SPECT(I)

87 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 ***)
NLABEL=18
PRINT 1007, NBLK

1007 FORMAT(/** AVERAGE OF*15, * TRANSFORMS/ //5X*I*3X*FREQUENCY*4X*AMPLITUDE*4X*PHASE*)
PRINT 1008, (I, XPLOT(I), SPECT(I), I=1, NPRINT)

1008 FORMAT(16, 2E13.5*, F8.2, I16, 2E13.5*, F8.2)
GO TO 89

69 CONTINUE
PRINT 1002, NBLK
PRINT 911, (I, XPLOT(I), RI(I), I=1, NPRINT)

89 CONTINUE
NPRCD7=NPRCD7+1
WHITE(7), PLABEL, NSPCT, (XPLOT(I), RI(I), I=1, NSPCT)
PRINT 1010, NPRCD7, PLABEL

1010 FORMAT(/**2IH *** * RECORD NO.*, I5, * ON TAPE7 CONTAINS *6A10,10*
L***)

C PLOT FANFULD PLCTS AND COMPUTE 1/3 G.B. SPECTRUM
GO TO (7C, 70, 62, 63, 62, 63), IPLUTA

62 ILOG=0 $ GO TO 61
63 ILOG=2

61 CALL PLOTNB(YLABEL, TRACK(J), L0, XPLOT, RI, NSPCT, ILOG, F1, F2, PLABEL, INPLABEL, IFF, I)
IF (ITYPESP .LT. 3) GO TO 70
ILOG=6
DO 90 I=1, NSPCT
90 RI(I)=AIMAG(SPECT(I)) ENCODE(50, 904, PLABEL)

100
APPENDIX H

904 FORMAT(*PHASE ANGLE, DEGREES*30X)
NPLABEL=20
CALL PLOTN1(YLABEL,TRACK(J),10,XPLUT,R1,NSPCT,1LOG,F1,F2,PLABEL,NPLABEL,0,0)
GO TO 100
70 CONTINUE
IF(IYPESP.EQ.3) GO TO 100

C

 COMPUTE 1/3 OCTAVE BAND SPECTRA
C

GO TO (100,65,1CC,190,65,65),!PLUTA
CALL HANDS(DELFP,NSPCT,SPECT,PPLUT,BAND,NPP,IERR,IYPESP)
IF(NPP.LE.0) GO TO 100
CALL PLOTN1(YLABEL,TRACK(J),10,BAND,PPLUT,NPP,PLABEL,NPLABEL,IFF)
64 CONTINUE
GO TO (72,71),IYPESP
71 ENCODE(60,73,PLABEL) TRACK(J)
73 FORMAT(*1/3 OR AUTO POWER SPECTRAL DENSITY*6X,A10,1DX)
GO TO 74
72 ENCODE(60,75,PLABEL) TRACK(J)
75 FORMAT(*1/3 OR AUTO POWER SPECTRUM*4X,A10,2DX)
74 CONTINUE
NCRD7=NCRRD7*1
WRITE(7) PLABEL,NPP,(BAND(I),PPLUT(I),I=1,NPP)
PRINT 1010,NCRD7,PLABEL
100 IF(IAUTOCUI(J).LE.0) GO TO 50
C

 COMPUTE AUTOCORRELATION
C

DO 101 I=2,NSPCT
Z(I)=SPECT(I)
101 Z(I)+NSPCT)=SPECT(NSPCT-I+2)
Z(I)=Z(NSPCT+I)=SPECT(1)
IF(IYPESP.GT.1) GO TO 113
DO 114 I=1,NPT
114 Z(I)=Z(I)/DELFP
113 CONTINUE
CALL FOURT1(Z,NPT,1,-1,0,SPECT)
CON=3.14159265358979/TMAX
IF(INZER0) 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-1+2)
1012 CONTINUE
C

 PRINT AUTOCORRELATION
C

DO 102 I=1,NPT
IM1=I-1

APPENDIX H

RI(I) = SPECT(I)

102 XPLOT(I) = I + 1

PRINT 1003, TRACK(J), (XPLOT(I), RI(I), I = 1, NSPCT)

1003 FORMAT(//"#AUTO CORRELATION", "/(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, 10C4, LABEL) TRACK(J)

1004 FORMAT(//"#AUTO CORRELATION", *A10, 32X)

NRCRD7 = NRCRD7 + 1

WRITE(17) PLABF.L, (XPLGT(I), RI(I), I = 1, NSPCT)

PRINT 1010, NPCRL7, PLABEL

IF(LAG1.EQ.0.AND.LAG2.EQ.0) GO TO 50

C
C SET UP PLOT ARRAYS AND PLOT AUTOCORRELATION

C

L1 = LAG1, L2 = LAG2

IF(LAG2.GT.0) GO TO 109

L1 = LAG2, L2 = -LAG2

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(I)

CALL ASCALE(RI, 10..NPLT, 1, 10.)

IF(NPLT-25) = 201, 201, 202

201 K = 1, GO TO 103

202 K = NPLT/256

103 ENCODE(50, 920, IOEN) TRACK(J)

920 FORMAT("#AUTO CORRELATION", *A10, 32X)

PN = RI(NPLT)

PX = RI(NPLT - 2) * 10. * PN

CALL FANFOLD(PI, NPLT, K, 1, NPLT, I2EN, 1H*, L*, PX, PN, YLABEL, 2, I2O, 0, 0, 

50 CONTINUE

RETURN

END

SUBROUTINE NORMAL

COMMON/BLK1/STARTT, IFMT, NBLK, IFUW2, NCH, NPRINT, IPLOTA, IPLUTC, OFFSC

1AL, DELTAT, SN, NRSKIP, LAP, NCROSS, ICROSS(2, 20), NCHP, YLABEL(2), IWINDOW

1, F1, F2, FYPESP

COMMON/BLK2/ICH(14), CHSUM(14), NUFF(14), CHSUMSQ(14), SIGMA(14), RMS(1

14), MEAN(14), SCALFA(14), CHSUM(14), TRACK(14), ICHAN(14)

COMMON/BLK3/NPT, TMAX, NPTU2, NSPCT, UELF, N64, NPT0128

1, INZERO, NREAD, NPTOT

COMMON/BLK5/PCTC, NBINS, CMAX(14), UMINS(14), OBOIN(14), BINS(10, 14)

1, CHISOC

REAL MU, MEAN

DIMENSION YL(3), ICEN(5)
APPENDIX H

DATA IDEN/10HCOUNTS ,4*10H /
NBINSM3=NBINS -3
DO 1=1,NCHP
J=ICHAN(I)
MU=MEAN(J)
ENCOD(30,902,YL) TRACK(J)
102 FORMAT(3X*HISTOGRAM FOR *A10,3X)
PN=PX=0.
CALL FANFOLD(HINS1,1,NBINS,L,1,LOG,IDJN,1H*,1.,PN,PX,YL,3,
1LOG,0.,XAPP,YL,XLABEL)
SIG=SIGMA(J)
FACTOR=1./ (2.5066*K2/*SIG)
DEL=OBIN(!)
OMAX=0.
SUM=0.
SFMIN(J).LT.-20.) GO TO 6
FN=PFUN(-20.,OMIN(J),MU,SIG)*FACTOR
GO TO 7
6 PN=0.
7 CONTINUE
PRINT 909, (B!NS(K,T ),K=1,NBINS)
905 FORMAT(*BINS(*/(1OF13.0))
CHISO=0.
DO 2 K=1,NBINS
A=K*DEL+OMIN(J)
SUM= B!NS(K,1)
P=SUM/NPTOT
PN= PFUN(A-DEL,4,MU,SIG)*FACTOR
IF(P<=0.0) GO TO 2
CHISO=CHISO*(P-PN)**2/PN+CHISO
2 CONTINUE
ALPHA=1.-?CTC/1CC.
PRINT 909, NBINSM3,CHISO,ALPHA,CHISQ
90C FORMAT(*//12X13F ** GOODNESS OF FIT TEST **//* DEGREES OF FR
1EDOM =*15,10X*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)
KCAL MU
PFUN=0.
DX=(B-A)/50.2
X=A-UX/2.
DO 1=1,50
X=X+DX
PFUN=PFUN+DX*EXP(-(X-MU)**2/(2.*SIG*SIG))
1 CONTINUE
RETURN
END
SUBROUTINE PLOTP(YLAREL,FRAMESL,NF,BAND,PLOT,NPP,PLABEL,MP,IFF)
APPENDIX H

```
DIMENSION BAND(1), PLOT(NPP), YLABEL(2), FRAME(1), PLABEL(5), FFLD(5)
1, IDEN(6), BCF(24)
DATA KCF/50., 63., 66., 100., 125., 160., 200., 250., 315., 400., 500., 630.,
1000., 1250., 1600., 2000., 2500., 3150., 4000., 5000., 6300., 10000.,
22000., 22000./

C PLOT POWER FOR BAND CENTER FREQUENCIES 50-20000 Hz

4001 DO 2001 I=1,NPP
2001 BAND(I)=I+4
PMAX=PLOT(I)
DO 2002 I=2,NPP
2002 PMAK=AMAK(PLOT(I),PMAK)
NMAK=ITAX(PMAX)
IF(NMAX.LT.PMAX) NMAX=NMAK+1
PMAX=NMAX
PMIN=PMAX-5.
I2=NPP
DO 2004 I=1,I2
IF(PLOT(I).LT.FMIN) PPLQT(I)=PMIN
2004 CONTINUE
PPLQT(NPP+1)=PMIN
PPLQT(NPP+2)=PMAX
CONTINUE
BAND(NPP+2)=I+5
IF(IFF.EQ.0) RETURN
FFID(I)=YLABEL(I)
FFID(2)=YLABEL(2)
NWFPOS=(NF9)/10
DO 5 I=1,NWFPOS
IFID(I*2)=FRAMEL(I)
ENCODE54,905, I DEN) PLABEL
905 FORM AT(*LOG *5A10)
CALL FANFOLD(PPLQT, NPP, 1, I, NPP, IDEN, IFID, PMAX, PMIN, FFLD, NWORDS+2,
1, 12G, 1, BCF, IFH FREQ UN E)
RETURN
SUBROUTINE BAND(S, N, N, I , N, BAND, N, I, 1, IFIELD, I)
COMPLEX SPECT
DIMENSION FPLT(1), IND(46,2), FNC(46), FNL(46)
DATA FNL/1. 1220,1.4125, 1. 7783,2.3872, 2. 8184,3. 5481,4. 4668,5. 0234,
17. 0795, 19. 129,1. 1220,1. 4125,1. 7783,2. 3872, 2. 8184,3. 5481,4. 4668,
25.0234, 29.755, 39.125, 49.2387, 4. 8184,3. 5481,4. 4668,5. 0234,
34.681, 56.234, 76.795, 97.699, 119.25, 141.25,1. 17763,2.2387, 2.8184,
35.466, 43.5481, 51.3456, 59.2345, 68.1234, 77.854, 86.854, 95.7654,
508143, 55841, 44668.
DATA FNC/1. 2969,1. 5049,1. 9955,2. 5119,3. 1623, 3. 9811,5. 0119,6. 3096,
17. 9466,1. 11,1. 58519, 1. 8499, 1. 9953,2. 5119,3. 1623, 3. 9811,5. 0119,6. 3096,
2.79. 433, 100., 125., 150., 184., 199., 30, 35, 39, 43, 47, 51, 55, 59, 63, 67,
71, 75, 79, 83, 87, 91, 95, 99, 103, 107, 111, 115, 119, 123, 127, 131,
3610.96, 394. 33, 1000.1250.9, 1500.9, 1849.9, 1995.3, 2511.9, 3062.3, 35981.1,
45011.9, 630. 9, 7543.3, 10000.0, 12569.9, 15049.9, 19953.3, 25119.9, 31623.7,
53981.1.
DIMENSION PSD(46), PSUPHLZ(46), BAND(27)
DO 3 I=1,27
```

APPENDIX H

3 BAND(I)=1
1 IERR=0
DO 1 I=1,46
AND=FNL(I+1)-FNL(I)
1 I=I
IF(BNU*GE.*DELF) GC TO 2
1 CONTINUE
IERR=1
RETURN
2 DO 62 I=1,90
PSDPHZ(I)=PSO(I)=G.
62 IND(I+1)=IND(I,2)=0
II=I
63 I=2,NSPCT
F=1/1-1)*DELF
68 IF(I1-40) 64,64,65
69 II=46
GO TO 66
64 IF(F-FNL(I1)) 66,67,67
66 IF(IND(I1,2).T.5.0C) IND(I1,1)=1
IND(I1,2)=IND(I1,2)+1
GO TO 63
67 II=II+1
GO TO 68
63 CONTINUE
NPP=G.
66 I=1,46
IF(IND(I,2)) 76,76,75
70 ISTR=IND(I,1)
NPP=NPP+1
CALL PNDSUM(SPECT(ISTR),IND(I,2),PSU(I))
PSO(I)=PSD(I)*DELF
PSDPHZ(I)=PSU(I)/(FNL(I+1)-FNL(I))
76 CONTINUE
NPP=NPP-(17-11)
IF(NPP. LT. 0) NPP=0
IF(NPP. GT. 27) NPP=27
PRINT 80, (PSU(I),PSO(I),PSDPHZ(I),I=1,46)
80 FORMAT(*1/3 OCTAVE BAND*5X*POWER SPECTRAL 5X*CENT.
1 FREQUENCY*4X*SPECTRUM*7X*ENERGY*/(F12.4E19.4,E15.4)
IF(NPP. EQ. 0) GO TO 3004
GO TO (3001,3002),IPWPLT
3001 DO 3003 I=1,NPP
PPLTOT(I)=-10G.
$ IF(PSO(I+16)GT.9.) PPLTOT(I)=ALOGL2(PSU(I+16))
3003 CONTINUE
GO TO 3004
3002 DO 3005 I=1,NPP
PPLTOT(I)=-10G.
IF(PSUPHZ(I+16)GT.9.) PPLTOT(I)=ALOGL2(PSUPHZ(I+16))
3005 CONTINUE
3004 RETURN
APPENDIX H

SUBROUTINE BNL(SL,NM,PSD)
  COMPLEX S(1)
  PSD=0.
  IF(NUMGT1) GO TO 1
  PSD=REAL(S(1))
  RETURN
1  PSD=REAL(S(1))+REAL(S(NUM))
  IF(NUM.EQ.2) RETURN
  NUM=NUM-1
  DO 2 I=2,NUM
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)
  XU = X(2)
  YU = Y(2)
  XP = X(3)
  YP = Y(3)
  SLN = (YN-YM)/(XN-XM)
  SLO = (YO-YN)/(XN-XM)
  SLP = (YP-YO)/(XP-XN)
  RKOUT = KNTOUT
  IYNX=X(1)/DX
  IF(IYNX.DT.X(1)) IYNX=IYNX-1
  IYNX=IYNX+1
  DO 1 I=1,IYNX
1  YOUT(I)=Y(I)
  IYNX=IYNX+1
  LIM = KNT+1
  DO 7000 N=3,LIM
    IF(SLN .NE. SLO .OR. SLO .NE. SLP) GO TO 3300
    LINEAR
    A(4) = 0.
    A(3) = 0.
    A(2) = SLO
    A(1) = YN - SLO*XN
    GO TO 6000
  CONTINUE
    A(1) = (SLO - SLN)/(XN - XM)
    A(2) = (SLP - SLO)/(XP - XN)
    IF(A(1) .NE. A(2)) GO TO 5000
    PARABOLIC
    C(1,1) = 1.
    C(2,1) = 1.
    C(3,1) = 1.

106
APPENDIX H

\[
\begin{align*}
C(1,2) &= XN \\
C(2,2) &= XP \\
C(3,2) &= XN \times XN \\
C(4,2) &= XN \\
C(1,3) &= YN \\
C(2,3) &= YO \\
C(3,3) &= YP \\
C(4,3) &= YP \\
A(1) &= YN \\
A(2) &= YO \\
A(3) &= YP \\
C(1,1) &= 1. \\
C(2,1) &= 1. \\
C(3,1) &= 0. \\
C(4,1) &= 0. \\
C(1,2) &= XN \\
C(2,2) &= XN \\
C(3,2) &= XN \\
C(4,2) &= XN \\
C(1,3) &= XN \times XN \\
C(2,3) &= XN \times YO \\
C(3,3) &= 2 \times XN \\
C(4,3) &= 2 \times XN \\
C(1,4) &= XN \times C(1,3) \\
C(2,4) &= XN \times C(2,3) \\
C(3,4) &= 3 \times C(1,3) \\
C(4,4) &= 3 \times C(2,3) \\
A(1) &= YN \\
A(2) &= YO \\
A(3) &= \tan(0.5 \times (\arctan(SL1) + \arctan(SL3))) \\
A(4) &= \tan(0.5 \times (\arctan(SL0) + \arctan(SLP))) \\
\end{align*}
\]

CALL SIMEO (C, 3, A, 1, DET, IPIV, 4, ISC)

5000 CONTINUE

C CUBIC

\[
\begin{align*}
C(1,1) &= 1. \\
C(2,1) &= 1. \\
C(3,1) &= 0. \\
C(4,1) &= 0. \\
C(1,2) &= XN \\
C(2,2) &= XN \\
C(3,2) &= XN \\
C(4,2) &= XN \\
C(1,3) &= XN \times XN \\
C(2,3) &= XN \times YO \\
C(3,3) &= 2 \times XN \\
C(4,3) &= 2 \times XN \\
C(1,4) &= XN \times C(1,3) \\
C(2,4) &= XN \times C(2,3) \\
C(3,4) &= 3 \times C(1,3) \\
C(4,4) &= 3 \times C(2,3) \\
A(1) &= YN \\
A(2) &= YO \\
A(3) &= \tan(0.5 \times (\arctan(SL1) + \arctan(SL3))) \\
A(4) &= \tan(0.5 \times (\arctan(SL0) + \arctan(SLP))) \\
\end{align*}
\]

CALL SIMEO (C, 3, A, 1, DET, IPIV, 4, ISC)

6000 CONTINUE

SUM = A(1)

VARP = 1.

GO TO 6100 

VARP = VARP * VAR

SUM = SUM + A(X) * VARP

6100 CONTINUE

YOUT(1YNX) = SUM

VAR = FLOAT(1YNX) * UX

1YNX = 1YNX + 1

IF (VAR \times LE. X(N-1)) GO TO 6000

XM = XN

YM = YN

XN = XO

YN = YO

XO = XP
APPENDIX H

YU = 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 - Y(N))/XP

7000 CONTINUE
IF(IYNX.GT.KNTOLT) RETURN
DO 2 I = IYNX,KNTCUT
2 YOUT(I) = Y(KNT)
RETURN
END

OVERLAY(PATS,4,4)
PROGRAM CFOSSSP
COMMON CMAIN
COMMON/BLK1/STARTT,ITFMT,NBLK,IPOW2,NCH,NPRINT,IPLOTA,IPLUTC,OFFSC
1AL,DELTAT,SN,NRSKIP,LAP,NCRROSS,ICROSS(2,20),NCHP,YLABEL(2),IWINDOW
1,F1,F2,ITYPESP,FCON,NFSKIP,IFF,LAG1,LAG2
COMMON/BLK2/IICH(14),CHSUM(14),NFF(14),CHSUMW(14),SIGMA(14),RMS(1
14),MEAN(14),SCALFAC(14),CHSUM(14),TRACK(14),ICHAN(14)
COMMON/BLK3/NPT,TMAX,NPT02,NSPC,DEL,NE4,NPT0128
1,INZEKO,NREAD
COMMON/BLK6/ISAVE(4),IJI,IXPLOT,IAUTA,I2,ISPECT
COMMON/BLK8/IAUTOSP(14),IAUTOCG(14),ICKF(20),ICRCONS(20),ITRA(20),
1ICH(20)
COMMON/BLK9/NFILTER
COMMON/BLK10/NRCROT
CALL CROSS(CMAIN(17),CMAIN(ISPECT),CMAIN(IJI),CMAIN(IXPLOT))
RETURN
END
SUBROUTINE CROSS(I,SPECT,F1,XPLOT)

COMPLEX SPECT(1),F1
DIMENSION R(1),X(1),PLABEL(6)
COMMON/BLK1/STARTT,ITFMT,NBLK,IPOW2,NCH,NPRINT,IPLOTA,IPLUTC,OFFSC
1AL,DELTAT,SN,NRSKIP,LAP,NCRROSS,ICROSS(2,20),NCHP,YLABEL(2),IWINDOW
1,F1,F2,ITYPESP,FCON,NFSKIP,IFF,LAG1,LAG2
COMMON/BLK2/IICH(14),CHSUM(14),NFF(14),CHSUMW(14),SIGMA(14),RMS(1
14),MEAN(14),SCALFAC(14),CHSUM(14),TRACK(14),ICHAN(14),IFILTER(14)
COMMON/BLK3/NPT,TMAX,NPT02,NSPC,DEL,NE4,NPT0128
1,INZEPN
COMMON/BLK6/ISAVE(4),IJI,IXPLOT,IAUTA,I2,ISPECT
COMMON/BLK8/IAUTOSP(14),IAUTOCG(14),ICKF(20),ICRCONS(20),ITRA(20),
1ICH(20)
COMMON/BLK9/NFILTER
COMMON/BLK10/NRCROT

108
APPENDIX H

DIMENSION IDEN(5),BAND(27),PPLLOT(27)
DATA RAD/57.2957795/
C
C
C
C START OF LCOP FCR COMPUTING CROSS FUNCTIONS
C
C
DO 70 1CR=1,NCRSS
K1=ICRSS(I1,ICR)  K2=ICRSS(I2,ICR)
IF(IAUTOSP(K1).EQ.0.AND.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(I,J,NPT,1J)
IJ=J2+(IBLK-1)*NCHP
CALL READMS(I,J(NSPCT+1),NPT,1J)
DO 71 I=1,NSPCT
71 SPECT(I)=SPECT(I)*Z(I)*CONJG(Z(I)*NSPCT))
IF(NFILTER(L).EQ.0.AND.IFILTER(L).EQ.0) GO TO 111
CALL READMS(I,R,L,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/(6.23185308*CON*NBLK)
DO 115 I=1,NSPCT
115 SPECT(I)=SPECT(I)*CON
ENCOD(23,400,FrlAM) TRACK(K1),TRACK(K2)
900 FORMAT(A10,3H X ,A10)
IF(ICRSSP(ICR).EQ.0) GO TO 105
GO TO (60,61) , ITYPESP
60 CON=CON*DELF
ENCOD(50,901,PLABEL) DELF
931 FORMAT(*CROSS POWER SPECTRUM (BANDWIDTH =G9.2,*)*,6X)
NPLABEL=44
PRINT 898, (PLABEL(I),I=1,5),FrlAM
898 FORMAT(*15A10,5X,3A10)
ENCOD(60,921,IDEN) TRACK(K1),TRACK(K2)
921 FORMAT(*CROSS POWER SPECTRUM *A10,1X,A10,7X)
GO TO 62
61 ENCOD(50,902,PLABEL)
APPENDIX H

902 FORMAT(*CROSS POWER SPECTRAL DENSITY*22X)
    NPLABEL=28
    PRINT 899, (PLABEL(I), I=1,3), FRAMEL
899 FORMAT(*I*3A10,5X,3A10)
    ENCODE(60,922,IDENT) TRACK(K1), TRACK(K2)
922 FORMAT(*CROSS POWER SPECTRAL DENSITY *A10,1X,ALG)
62 IF(NPRINT.GT.0) PRINT 897
897 FORMAT(/4X*I*4X*FREQUENCY*9X*REAL*1CX*IMAG*7X*AMPLITUDE*7X*PHASE*1)

C SET UP PLOT ARRAYS AND PRINT CROS S SPECTRUM
C
DO 72 I=1,NSPCT
  F=(I-1)/TMAX
  XPLOT(I)=F
  AMP=EXP(SPECT(I))
  IF(AMP) 79,80,78
78 CONTINUE
  IF(I-NPRINT)73,73,200
73 PRINT 915, I,F,SPECT(I),AMP,ARG
915 FORMAT(15.5E14.5)
200 IF(IPLOTC.EQ.0) GO TO 201
GO TO (201,201,201,201,201,201,201), IPLOTC
201 R(1)=SPECT(1)
    R(1+NSPCT+2)=AIMAG(SPECT(1))
    GO TO 72
202 R(I)=AMP
    R(I+NSPCT+2)=ARG.
72 CONTINUE
201 IF(IPLOTC.IE.2) 301,301,302
201 GO TO (301,301,302,302,302,302,302), IPLOTC
301 IDENT=10H (REAL)
    IDEN7=10H (IMAG)
    GOTO 303
302 IDENT=10H AMPLITUDE
    IDEN7=10H PHASE
303 NRCRD7=NRCRD7+1
    NSPCTP2=NSPCT+2
    PRINT 1000, NRCRD7, IDENT, IDEN6
1000 FORMAT(/21H ** ** ** RECORD NO., I5, * ON TAPE7 CONTAINS NO., I5, * PHASE **
    WRITE(7) IDENT, IDEN6, NSPCT, (XPLOT(I), R(I), I=1,NSPCT)
    NRCRD7=NRCRD7+1
    PRINT 1003, NRCRD7, IDENT, IDEN7
    WRITE(7) IDENT, IDEN7, NSPCT, (XPLOT(I), R(I+NSPCTP2), I=1,NSPCT)
C PLOT +FANFOLD PLOTS
C
NW=(NPLABEL-11)/10+2
NPLABEL=NW*10

110
APPENDIX H

IF(IPLTOC.EQ.0) GO TO 105
GO TO (107,108,109,109,109,109) IPLTOC
107 IPLG=2 GO TO 106
108 IPLG=2
109 ENCOD(10,904,PLABEL(NW))
903 FORMAT(* (REAL*))
   CALL PLOTNY(ylabel,framtn,23,xplot,ri,nspect,ilog,f1,f2,plabel,nplab
   rel,iff,i)
   ENCOD(10,905,plabel(NW))
904 FORMAT(* (IMAG*))
   CALL PLOTNY(ylabel,framtn,23,xplot,ri,nspect,ilog,f1,f2,plabel,nplab
   rel,iff,i)
905 CONTINUE
IF(CRCCOR(IR).LE.0) GO TO 411
C COMPUTE CROSSCORRELATION
C DO 74 I=2,NSPECT
   Z(I)=SPECT(I)
74 Z(NSPECT+I)=CONJG(SPECT(NSPECT-I+2))
   Z(I)=Z(NSPCT+I)=SPECT(I)
114 CONTINUE
IF(MPT.GT.1) GO TO 113
   Z(I)=Z(I)/MPT
113 CONTINUE
   CALL FOURCT(Z,NPT,-1,-1,SPECT)
   PRINT 916, TRACK(K1), TRACK(K2)
   CON=6.2*31kHz308/TMAX
   DO 110 I=1,NPT
110 SPECT(I)=Z(I)*CCN
C SET UP PLOT ARRAYS, PRINT AND PLOT CORRELATION
C IF(INZERO) 5CC,5OC,5C1
500 DO 502 I=1,NSPECT
   IM=I-1
   IPN=I+NSPECT
   R(I,IPN)=SPECT(I)
   XPLT(502)=IM
   MN=I-NSPECT
111
APPENDIX H

R1(I)=SPECT(IPN)

502 XPLOT(I)=IMN
GO TO 503

501 DO 90 I=1,NSPCT
IM1=I-1
IPN=I+NSPCT
IM2=2*IM1
RI(IPN)=NPT*SPECT(I)/(NPT-IM2)
XPLOT(IPN)=IM1
IMN=I-NSPCT-1
IM2=2*IMN $ IF(I.LE.1) IM2=IM2+1
RI(I)=NPT*SPECT(IPN)/(NPT+IM2)
90 XPLOT(I)=IMN
CONTINUE

503 CONTINUE
PRINT 917, (XPLOT(I),RI(I),I=1,NPT)
917 FORMAT(2X,8(F5.3,E11.3))
916 FORMAT(*CROSS CORRELATION, 8(A10,*X*AI0//8(7X*I*5X*RXY*))
ENCODE(60,523,PLABEL) TRACK(K1),TRACK(K2)
923 FORMAT(*CROSS CORRELATION *A10,IXA10)2CX)
WRITE(7) PLABEL,NPT,(XPLOT(I),RI(I),I=1,NPT)
NRCRD7=NCRCD7+1
PRINT 1000, NRCRD7,PLABEL
IF(LAG1.EQ.0.AND.LAG2.EQ.0) GO TO 411
IM1=LAG1+NSPCT+1 $ IM2=LAG2+NSPCT+1
926 IF(I1.GE.I2) GO TO 411
NPLT=I2-I1+1
CALL ASCALE(RI(H),100,NPLT,1,10.)
IF(NPLT.EQ.0) GO TO 411
IF(NPLT-256) 101,101,102
101 K=1 $ GO TO 103
102 K=NPLT/256
103 ENCODE(59,920,IDEN) FRAMEL
920 FORMAT(*CROSS CORRELATION*3X,5A10)
PN=RI(NPLT+11)
PX=RI(NPLT+I1+1)*10.0*PN
CALL FANFLOU(RI(II),NPLT,K,1,NPLT,1DEN,1H*,1*,PX,PN,YLABEL,2,126,0
1,0,1H*)
GO TO 411
75 PRINT 918, TRACK(K1),TRACK(K2)
918 FORMAT(*CROSS SPECTRUM FOR *A10,*X*AI0,*CANNOT *BE COMPUTED*)
GO TO 70
411 IF(ICON(NK),1.E0) GO TO 401

C

COMPUTE COHERENCE
C

CALL READMS(9,R1,NSPCT,J1)
CALL READMS(9,R1+NSPCT+1),NSPCT,J2)
CALL READMS(9,SPECT,NPT,NSPCT+1)
CON=DELTAT**2/(6.283185308*WCON*NSPCT)
DO 402 I=1,NSPCT
XPLOT(I)=(I-I1)/TMAX

112
APPENDIX H

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, XPLRT(I), RI(I), I=1, NPRINT)
C
C PRINT AND PLOT COHERENCE
C
403 FORMAT(//1XCOHERENCE, *3A10//5X*3X*FREQUENCY*4X*COHERENCE*3(6X*1e
13X*FREQUENCY*4X*COHERENCE*)/(10, 2E13.5, 10, 2E13.5, 10, 2E13.5, 10, 2E13.5)
2.5))
ENC0DE(00, 924, PLABEL) TRACK(K1), TRACK(K2)
924 FORMAT(*COHERENCE, *4A10, *4X*A10, 2X)
NRCRD7=NCRD7+1
WRITE(7) PLABEL, NSPCT, (XPLRT(I), RI(I), I=1, NSPCT)
PRINT 1000, NCRD7, PLABEL
ENC0DE(99, 406, PLABEL)
404 FORMAT(*COHERENCE*4X)
CALL PLOTNR(Y1, AEL, LAM, KL, XPLUT, RI-1, NSPCT, e
1, H2, PLABEL, 10, IFF, 11)
401 IF(TRA(I).EQ.O) GO TO 70
C
C COMPUTE TRANSFER FUNCTION
C
CALL READMS(9, SPECT, NPT, NCHP+1)
IF(TRA(I).LT.O) GO TO 405
CALL READMS(9, RI, NSPCT, 11)
ENC0DE(60, 406, PLABEL) TRACK(K1), TRACK(K2)
GO TO 407
405 CALL READMS(9, RI, NSPCT, J2)
ENC0DE(60, 408, PLABEL) TRACK(K1), TRACK(K2)
408 FORMAT(*TRANSFER FUNCTION, TRAY FOR *A10,3H X, A10, 2X)
C
C 406 FORMAT(*TRANSFER FUNCTION, TRAY FOR *A10,3H X, A10, 2X)
C
407 CON=DELTA*T/2(16.26185306*NC0N*NBLK)
GO 409 I=1, NSPCT
XPLRT(I)=(I-1)/TMAX
DENOM=10*RI(I)
IF(DENOM) 414, 414, 415
414 RI(I)=0. $ GOTO 409
415 RI(I)=CABS(SPECT(I)/DENOM)*CON
409 CONTINUE
C
C PRINT AND PLOT TRANSFER FUNCTION
C
PRINT 410, PLABEL, (I, XPLRT(I), RI(I), I=1, NPRINT)
410 FORMAT(//1X#10, //5X*3X*FREQUENCY, *7X*TRA*3(9X*1*3X*FREQ
UENCY*7X*TRA*)/(10, 2E13.5, 10, 2E13.5, 16, 2E13.5, 16, 2E13.5)
NRCRD7=NRCRD7+1
WRITE(7) PLABEL, NSPCT, (XPLRT(I), RI(I), I=1, NSPCT)
PRINT 1000, NCRD7, PLABEL
CALL PLOTNR(YLABEL, FRAMEL, 23, XPLRT, RI, NSPCT, 0, F1, F2, PLABEL, 24, IFF, 11)
70 CONTINUE
RETURN
END
REFERENCES


"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