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.
<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>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Tape Format 3</td>
<td>50</td>
</tr>
<tr>
<td>APPENDIX E – FLOW CHART FOR PATS</td>
<td>51</td>
</tr>
<tr>
<td>APPENDIX F – PROGRAMS AND SUBPROGRAMS USED BY PATS</td>
<td>54</td>
</tr>
<tr>
<td>APPENDIX G – LANGLEY LIBRARY SUBROUTINES</td>
<td>57</td>
</tr>
<tr>
<td>Subroutine ASCALE</td>
<td>57</td>
</tr>
<tr>
<td>Function GAMMF</td>
<td>58</td>
</tr>
<tr>
<td>Subroutine ITR2</td>
<td>60</td>
</tr>
<tr>
<td>Subroutine OPENMS</td>
<td>62</td>
</tr>
<tr>
<td>Subroutine READMS</td>
<td>63</td>
</tr>
<tr>
<td>Subroutine RECID</td>
<td>64</td>
</tr>
<tr>
<td>Subroutine RECOUT</td>
<td>66</td>
</tr>
<tr>
<td>Subroutine SIMEQ</td>
<td>68</td>
</tr>
<tr>
<td>Subroutine WRITMS</td>
<td>69</td>
</tr>
<tr>
<td>APPENDIX H – SOURCE LISTING OF PATS</td>
<td>70</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>114</td>
</tr>
</tbody>
</table>
PROGRAM FOR THE ANALYSIS OF TIME SERIES

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

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

SUMMARY

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

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

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

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

INTRODUCTION

Although there have been many computer programs written for the purpose of time-series analysis, each program depends upon the type of data and the objectives of the research. Standard Fourier series routines are useful in describing deterministic periodic functions. Through a slight generalization, a similar routine can be employed in the analysis of transient deterministic functions. However, when the signal is considered to be random, another type of analysis, based upon the concept of power spectra, must be
utilized. Most power-spectral techniques compute the average lagged product or correlation function of the input signal, which is quite expensive in terms of time and storage. Further, many of these programs are essentially research tools and are inefficient for the analysis of large quantities of data.

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

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

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

(1) Auto and/or cross power spectra
(2) Autocorrelations and/or cross correlations
(3) Fourier coefficients
(4) Coherence functions
(5) Transfer functions
(6) Histograms

Power spectra may be filtered in the frequency domain prior to output or further processing with a filter of the user's description. Data may be output on a line printer in tabular form and/or plotted on the fanfold form, as specified by the user. In addition, all output is saved on binary files, which may be processed further or displayed by means of user-designed graphics.
The program was specifically designed to run economically and efficiently in a batch-processing environment from remote terminal equipment. This requirement places a constraint on the size of the program, which in turn limits the maximum resolution obtainable in the frequency domain. In some cases it was prudent to use existing subroutines which are written in CDC COMPASS, an assembler level language. Functional descriptions of these subroutines are provided in an appendix, together with other information necessary to allow a user to adapt PATS to a non-CDC system with random-access capability.

SYMBOLS

\[ A_k, B_k \] Fourier coefficients

\[ B_{2j} \] Bernoulli number

\[ e \] base value for natural logarithms, 2.7182818284

\[ E( ) \] expectation operator

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

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

\[ F(\omega) \] continuous function of \( \omega \), 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 \( y(t) \)

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

\[ \text{Im}( ) \] 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

$p(x^2)$ probability density function, a function of the variable $x^2$

$P_f(\omega)$ 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 $y(t)$

$\text{Re}(\ )$ 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

var()  variance operator

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

$W_R$  data-window correction factor for correlation estimator

$W_u$  data-window correction factor for spectral estimator

x(t)  continuous function of time  t

X(\omega)  continuous function of  \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 $z_k = \sum_{j=0}^{N-1} z_j W^{jk}$
\( \alpha \) 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

\( \omega_v, f_v \) Nyquist frequency, rad/sec and Hz, respectively

Mathematical notation:

\( \sim^* \) estimated quantities

\( ^\prime \) new variable

\( * \) complex conjugate operator

6
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 \(-T/2 \) to \( T/2 \) continuously, then the finite Fourier transform of \( f(t) \) is given by

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

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

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 \( \omega_N = \pi/\Delta t \), 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 \( \omega_N \) 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

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

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

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

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

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

(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:

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

(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

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

(11)

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

$$u_{T/2}(\omega) = \frac{T}{2\pi} \frac{\sin(\omega T/2)}{\omega T/2}$$

(12)

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

![Figure 2.- The average smoothed spectral estimate.](image-url)
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

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

(13)

the Hamming data window given by

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

(14)

and the Parzen data window given by

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

(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

\[
\tilde{S}_f(\omega_k) = \frac{(\Delta t)^2}{2\pi w_u} |z_k|^2
\]

(16)

where

\[
w_u = \int_{-\infty}^{\infty} u_d^2(t) \, dt
\]

(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

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

where \( E \) 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 $\tilde{S}_1(\omega)$ is a chi-square random variable. Then,

$$\frac{\operatorname{var}[\tilde{S}_1(\omega)]}{\left\{E[\tilde{S}_1(\omega)]\right\}^2} = \frac{2}{k_{eq}}$$

where $k_{eq}$ is the equivalent number of degrees of freedom of the random variable $\tilde{S}_1(\omega)$, and bounds on the variation may be obtained from the relation (see ref. 5)

$$\frac{k_{eq} \tilde{S}_1(\omega)}{2 \chi^2_{k_{eq}}(\alpha/2)} \leq E[\tilde{S}_1(\omega)] \leq \frac{k_{eq} \tilde{S}_1(\omega)}{2 \chi^2_{k_{eq}}(1 - \alpha/2)}$$

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

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

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

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. Illustration of circular correlation error.](image-url)
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,
\[ R_{xy}(\tau) = W_R \int_{-\infty}^{\infty} \tilde{S}_{xy}(\omega) e^{i\omega t} \, d\omega \]  

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

\[ \gamma_{xy}^2(\omega) = \frac{\left| \tilde{S}_{xy}(\omega) \right|^2}{\tilde{S}_x(\omega) \tilde{S}_y(\omega)} \]  

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

\[ \left| \tilde{S}_{xy}(\omega) \right|^2 \leq \tilde{S}_x(\omega) \tilde{S}_y(\omega) \]  

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.
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.

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 $j$th 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 \( \chi^2_e \), is computed by

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

(30)

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

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

(31)

where \( n \) is the number of degrees of freedom. The critical value of \( \chi^2 \), or \( \chi^2_c \), for the \( \alpha \) significance level may be found from equation (31). The normality hypothesis is rejected when \( \chi^2_e > \chi^2_c \). The decision to reject the hypothesis, however, is left to the user, as the value of \( \chi^2_e \) 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 60000\( g \) for compiling and executing the source version presented here and 55000\( g \) 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 Programming Manual for format.
RE WIND (TAPE 7)
1 REQUEST, TAPE 99, HY.
COPYBF (TAPE 7, TAPE 99)
DROPFIL (TAPE 99)
EXIT.

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.
1 REQUEST, TAPE 1, HY.
REWIND (TAPE 1)
SETINDF.
I.GO.

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

---

1See Langley Computer Programming Manual for format.
2End-of-record card.
3End-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></td>
<td>I</td>
<td>Total number of data points to be read for each channel</td>
</tr>
<tr>
<td>NREAD</td>
<td></td>
<td>I</td>
<td>Number of data points per block to be read for each channel (maximum value is 1024 for INZERO=0, 512 for INZERO=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>
| IAUTOCO      | 0             | I              | Array of NCH codes for computing autocorrelation:  
|              |               |                | 1 compute autocorrelation for channel i  
|              |               |                | 0 do not compute autocorrelation for channel i |
| IFILTER      | 0             | I              | Array of NCH codes for spectral filtering:  
|              |               |                | 1 filter auto and cross spectra for channel i  
|              |               |                | 0 do not filter spectra for channel i  
|              |               |                | (autocorrelation and cross correlation will be computed from filtered spectra) |
| NCROSS       | 0             | I              | Number of pairs of channels to perform cross functions on (maximum value = 20) |
| ICROSS       | 0             | I              | Array of channel numbers for cross functions:  
|              |               |                | ICROSS(1,i) first channel no. for pair i  
|              |               |                | ICROSS(2,i) second channel no. for pair i |
| ICRSP        | 0             | I              | Array of NCROSS codes for computing cross spectra for each pair of channels:  
|              |               |                | 1 compute cross spectrum for pair i  
|              |               |                | 0 do not compute cross spectrum for pair i |
| ICRCOR       | 0             | I              | Array of NCROSS codes for computing cross correlations:  
|              |               |                | 1 compute cross correlation for pair i  
|              |               |                | 0 do not compute cross correlation for pair i |
| ITRA         | 0             | I              | Array of NCROSS codes for computing transfer functions:  
|              |               |                | 1 compute transfer function,  
|              |               |                | $\tilde{H}(\omega) = \tilde{S}_{xy}(\omega)/\tilde{S}_x(\omega)$  
|              |               |                | 0 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 |
| IPLOTC       | 0             | I              | Code for cross spectral fanfold plots and/or binary tape output:  
0 no output  
1 plot and save real and imaginary (linear scale) against frequency (linear scale) |
<table>
<thead>
<tr>
<th>FORTRAN</th>
<th>Default</th>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>value</td>
<td>type</td>
<td></td>
</tr>
<tr>
<td>IPlOTC</td>
<td>0</td>
<td>I</td>
<td>3 plot and save magnitude and phase</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(linear) against frequency (linear)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5 plot magnitude (log scale) and phase</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>against frequency (linear) and save</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>magnitude and phase values</td>
</tr>
<tr>
<td>Fl</td>
<td>0.0</td>
<td>R</td>
<td>Lower limit of frequency to be plotted</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>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</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>on narrow-band spectra plots</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(If Fl 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</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>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</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>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</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>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</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>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</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>for each channel</td>
</tr>
<tr>
<td>DMIN</td>
<td>0.</td>
<td>R</td>
<td>Array of values of minimum readings</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>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</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>data block (block size is 2 \times NREAD)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(this option should be used for runs</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>requesting cross correlations)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 no zero insertion</td>
</tr>
</tbody>
</table>
FORTRAN name | Default value | Parameter type | Description
---|---|---|---
NFILTP | 0 | I | Number of points in input spectral filter:
0 no spectral filter
(maximum value=50)
FREQF | 0.0 | R | Array of values of frequency for spectral filter
WGHTF | 0.0 | R | Array of values of weights for spectral filter
(user should be careful to specify points close together where derivative of filter function changes)

Input cards after NAMELIST input:

<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 off-scale values read for each channel

(6) For each channel processed,

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

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

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

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

(8) For each pair of channels processed,

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

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

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

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

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

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

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

Restrictions and Limitations

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

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

(2) The version of the program being presented has a maximum block size of 1024. The program storage requirements are 55000g for the absolute binary version and 60000g 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 WGHFTF 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 ITPES3=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.
APPENDIX A

FINITE FOURIER TRANSFORM OF A PERIODIC SIGNAL

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

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

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

The finite Fourier transform of this signal is given by

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

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 \( T = vp \), 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 \( t = vp \), 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,

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

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

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

and

\[
\tan \phi_m = \frac{B_m}{A_m} = -\frac{\text{Im}(X_{\nu m})}{\text{Re}(X_{\nu m})} \quad \text{(A3)}
\]

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

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

and

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

Thus, the square wave admits the Fourier series representation

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

The magnitude and phase of this representation are given by

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

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

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

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

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

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

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

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

(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

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

Taking the ensemble expectation of this quantity yields

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

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

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

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

39
APPENDIX B

Now, define

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

(B6)

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

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

(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

\[
\int_{-\infty}^{\infty} E[\hat{S}_x(\omega)] \, d\omega = \frac{\pi}{T} \frac{1}{4\pi^2} \int_{-\infty}^{\infty} dt_1 \int_{-\infty}^{\infty} u_d(t_1) \, u_d(t_2) \, \mathcal{R}_\chi(t_1 - t_2) \, dt_2 \int_{-\infty}^{\infty} e^{i\omega(t_1 - t_2)} \, d\omega 
\]  

(B8)

and since

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

(B9)

equation (B8) becomes

\[
\int_{-\infty}^{\infty} E[\hat{S}_x(\omega)] \, d\omega = \frac{\pi}{T} \frac{1}{2\pi} \int_{-\infty}^{\infty} dt_1 \int_{-\infty}^{\infty} u_d(t_1) \, u_d(t_2) \, \mathcal{R}_\chi(t_1 - t_2) \, \delta(t_1 - t_2) \, dt_2 \\
= \frac{\pi}{T} \frac{1}{2\pi} \int_{-\infty}^{\infty} u_d^2(t) \, \mathcal{R}_\chi(0) \, dt_1 \\
= \frac{\mathcal{R}_\chi(0)}{2T} \int_{-\infty}^{\infty} u_d^2(t_1) \, dt_1 
\]  

(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:

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

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

\[
\tilde{S}_x(\omega) = \frac{2T}{W_u} \hat{S}_x(\omega) \quad (B12)
\]

where

\[
W_u = \int_{-\infty}^{\infty} u_d^2(t_1) \, dt_1 \quad (B13)
\]

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

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

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

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

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

For the boxcar window,

\[
W_u = T \quad (B15)
\]

For the Hann window,

\[
W_u = \frac{3T}{8} \quad (B16)
\]
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,

\[
\hat{R}_X(\tau) = \int_{-\infty}^{\infty} \hat{S}_X(\omega) e^{i\omega \tau} d\omega \tag{C1}
\]

Thus,

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

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

\[
E\left[ \hat{S}_X(\omega) \right] = \frac{2\pi}{W_u} \int_{-\infty}^{\infty} \hat{S}_X(\omega') U_d^2(\omega' - \omega) d\omega' \]

Thus, equation (C2) becomes

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

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

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

\[
-\frac{2\pi}{W_u} R_X(\tau) \int_{-\infty}^{\infty} d\omega_0 U_d^2(\omega_0)e^{-i\omega_0\tau} \tag{C4}
\]

Recall the definition (from eq. (B6))

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

\[
U_d^2(\omega) = U_d(\omega) U_d^*(\omega) = \frac{1}{4\pi} \int_{-\infty}^{\infty} dt \int_{-\infty}^{\infty} dt' u_d(t) u_d(t') e^{-i\omega(t-t')}
\]

and

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

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

\[
E[\hat{R}_X(\tau)] = \frac{\int_{-\infty}^{\infty} u_d(t) u_d(t + \tau) dt}{\int_{-\infty}^{\infty} u_d^2(t) dt} \quad (C6)
\]

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

\[
\bar{R}_X(\tau) = W_R \int_{-\infty}^{\infty} \hat{S}_X(\omega) e^{i\omega \tau} d\omega
\]
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></td>
<td>Serial number; the input card format for serial number should be 6 digits wide</td>
</tr>
<tr>
<td>3</td>
<td></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>
</tr>
<tr>
<td>4</td>
<td></td>
<td>Words 5 and 6 are additional engineering identification</td>
</tr>
<tr>
<td>5</td>
<td></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>
</tr>
<tr>
<td>6</td>
<td></td>
<td>Frame count which starts at 1 for each new serial number</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>Data channel N, where N is the number given in logical word 1</td>
<td></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>$1 \leq N \leq 10$</td>
<td>20</td>
<td>25</td>
<td>500</td>
</tr>
<tr>
<td>$10 &lt; N \leq 20$</td>
<td>30</td>
<td>17</td>
<td>510</td>
</tr>
<tr>
<td>$20 &lt; N \leq 30$</td>
<td>40</td>
<td>12</td>
<td>480</td>
</tr>
<tr>
<td>$30 &lt; N \leq 40$</td>
<td>50</td>
<td>10</td>
<td>500</td>
</tr>
<tr>
<td>*$30 &lt; N \leq 100$</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</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</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</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:

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

where

\[
NFR = (510 - IWD)/KCH
\]
NFR is the blocking factor (integer)
IWD is the number of nonrepeated words in the record
KCH is the number of data channels

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

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

Therefore,
SIZE = 56 * 9 + 1 + 2 = 507

Tape Format 3

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

<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>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?

No

Yes

Compute histograms and print fanfold plots

Read transform for one channel from random-access file

Apply spectral filter if desired

Compute auto PSD

Is automatic spectral output desired for this channel?

No

Go to C

Yes

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

Yes

Print fanfold plot as indicated by IPLOTA

C Is autocorrelation desired for this channel?

No

Yes

Calculate inverse transform from auto PSD

Calculate autocorrelation

51
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

D

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

Apply spectral filter if desired

Compute cross PSD

Go to E

No

Go to A

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

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

No

Go to G

Go to E

Go to F
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

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?
Yes
Print fanfold plot

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

Is transfer function for this pair desired?
Yes
Go to H
No
Go to I

No
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

54
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, \text{DIMENSION ARRAY (102);} \quad N = 75, \ K = 3, \text{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(\((N\times K)+1\)) and computes a scale factor and stores it in ARRAY(\((N\times K)+1+K\)). The scale factor will be a power of \(10 \times (2,4,5,\text{or } 10)\). The data in the array may be scaled to floating-point inches by using a formula similar to the following:

\[ SV = \frac{(AE-MV)/SF}{SF}, \quad \text{where } SV \text{ is the scaled value, } AE \text{ is the present value of array element, } MV \text{ is either the minimum value or the value desired at the origin, and } SF \text{ is the scale factor computed by the subroutine.} \]

Storage: 262g locations for the CDC 6000 series.

Subprograms used: ALOG, ALOG10.

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

Subroutine date: September 3, 1970.
APPENDIX G

Function GAMMF

Language: FORTRAN

Purpose: To compute the incomplete gamma function

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

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

Use: \( Y = \text{GAMMF}(A, X) \), 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 GAMMF: GSERES, GCHEB, GFRAC, GAMNEG.

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

(a) If \( A = 0 \),

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

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

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

(c) If \( X = 0 \),

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

which is the complete gamma function.

A rational Chebyshev approximation is used:

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

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

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

\[ \Gamma(A, X) = e^{-X} X^A \left( \frac{1}{X} - \frac{A}{1} + \frac{1}{X+1} - \frac{2-A}{X+1} + \frac{1}{X+2} - \cdots \right) \]
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 610ₖ 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: 

```plaintext
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 $|X_1| > \epsilon_1$,

$$\frac{|X_1 - X_{1-1}|}{X_1} \leq \epsilon_1$$

If $X_1 \leq \epsilon_1$,

$$|X_1 - X_{1-1}| \leq \epsilon_2$$

Storage: 260g 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

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

Subroutine date: March 29, 1971.
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 20018.

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 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).
APPENDIX G

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 20018. 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 multifiles using RECOUT, then he must end each file by setting $\text{IEOF} = 1$ and not by using the END FILE statement. Consequently, he should then test for end-of-file in REcin by testing $\text{ICOUNT}$ 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 TAPE\text{nn} — 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 $\text{IT}$ parameter was not 1 or 2.

3. BUFFER TOO SMALL — The buffer size was less than 20018.

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:
  \[ \text{DETERM} = \begin{vmatrix} 10 & 100 \end{vmatrix} \times \text{ISCALE} \times \text{DETERM} \]
- 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: 432g locations.

Subroutine date: August 1, 1968.
APPENDIX G

Subroutine WRITMS

Language: COMPASS

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

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

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

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

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

Storage: 1024 locations.

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

Error messages: (1) UNASSIGNED MEDIUM FILE XXXXXXX
                (2) FILE WAS NOT OPENED BY A CALL TO SUBROUTINE OPENMS
                (3) INDEX BUFFER IS OF INSUFFICIENT LENGTH

Subroutine date: March 29, 1971.
APPENDIX H

SOURCE LISTING OF PATS

OVERLAY(PATS,0,0)

PROGRAM MAIN(INPUT,OUTPUT,TAPE1,TAPE5=INPUT,TAPE6=OUTPUT,TAPE7,TAPE8,TAPE9)

COMMON MAIN(4102)

COMMON MAIN MUST BE DIMENSIONED 4*NMAX+6

COMMON INDEX(129),KNDEX(802)

COMMON BLK1/START,TFTM,NBLK,IP0M2,NCH,NPRINT,ITFLUTA,IPLOT,OFFSC

1AL,DELTAT,SN,NRSKIP,LPAP,NR CROSS,ICPSS(2,20),NCHP,YLABEL1(2),IWINDOW

F1,F2,ITYPESP,WCUT,NFSKIP,TF,LAGL,LAGS

COMMON/BLK2/ICH(14),CHSUM(14),NFF(14),CHSUMSQ(14),SIGMA(14),RMS(14),MEAN(14),SCALFAC(14),CHSUM(14),TRAC(14),ICHAN(14),IFILTER(14)

COMMON/BLK3/NPT,MAX,NPT02,NSPECT,DELF,N64,NPT0128

1,INZERO,NREAD,NPTUT

COMMON/BLK4/NMAX,NCHMAX,NBCMAX

COMMON/BLK5/PCTC,MBINS,MAX(14),MIN(14),OBIN(14),BINS(100,14)

1,CHSOC

COMMON/BLK6/ISAVE14,IRI,IXPLOT,IDATA,IZ,I SPECT

COMMON/BLK7/IWORD(11)

COMMON/BLK8/IASP(14),IAUTO(14),ICRSP(20),ICRCOR(20),ITRA(20),

LCOH(14)

COMMON/BLK9/NFILIP,FREQ(50),WGHTF(50)

COMMON/BLK10/NRCFD7

DATA NRCFD7/0/

NMAX=1024
NCHMAX=14
NBCMAX=80

IZ=IRI=1
IXPLOT=IRI+NMAX+4
ISPECT=ISAVE14=IXPLOT+NMAX+2
IDATA=ISAVE14+64*NCHMAX

PATS=4*PATS

CALL OPENMS(9,INDEX,129,0)

CALL OPENMS(6,KNDEX,NBCMAX+2,0)

PRINT 1000

1000 FORMAT(HE/////////23XI1H*43(2H*1/23XI1H*85XI1H*23XI1H*8XI1H*23XI1H*8XI1H*23XI1H*8XI1H*23XI1H*8XI1H*23XI1H*8XI1H*23XI1H*8XI1H*23XI1H*8XI1H*23XI1H*8XI1H*23XI1H*8XI1H*23XI1H*8XI1H*23XI1H*8XI1H

READ NAMELIST AND CARD INPUT

CALL OVERLAY(PATS,1,0)

READ ONE BLOCK OF INPUT TIME SERIES DATA FOR NCH CHANNELS FROM

BINARY TAPE AND STORE ON RANDOM ACCESS FILE
APPENDIX H

* COMPUTE FOURIER TRANSFORM FOR NC CHANNELS AND STORE ON RANDOM
* ACCESS FILE
* REPEAT UNTIL ALL DATA IS READ AND PROCESSED
*C
* CALL OVERLAY(PATS,2,0)
* CALL OVERLAY(PATS,3,0)

C AVERAGE NBLK POWER SPECTRA FOR EACH CHANNEL FOR AUTO SPECTRA AND
C APPLY SPECTRAL FILTER, COMPUTE AUTOCORRELATION, PRINT FANFOLD PLOTS
*C
* CALL OVERLAY(PATS,4,0)
C TO COMPUTE CROSS SPECTRA
C GET NBLK TRANSFORMS FOR BOTH OF THE CHANNELS IN EACH SELECTED PAIR
C AVERAGE THE PRODUCTS FOR NARROW BAND SPECTRA, COMPUTE CROSSCORRELATIONS,
C COHERENCE, AND TRANSFER FUNCTIONS, PRINT FANFOLD PLOTS.
C
IF(NCROSS.GT.0) CALL OVERLAY(PATS,5,0)
GO TO 100
END
SUBROUTINE PLOTNB(YLABEL,FRAMEL,NF,XPLUJ,I1,NSPCT,ILOG,F1,F2,PLABEL,IL,NP,IFF,SEARCH)
DIMENSION YLABEL(2),FRAME(1),XPLUT(1),RI(1),PLABEL(5),FF10(5),
IDEN(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
ILOGP=ILOG+1
IF(SEARCH.EQ.0) GO TO 6
J1=1 & J2=NSPCT
DO 1 I=1,NSPCT
J=I
IF(XPLUT(I).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 J=J1,NSPCT
J=NSPCT-J+1
IF(XPLUT(J).LE.F2) GO TO 4
3 CONTINUE
4 J2=J
NF=NF/10 & IF(NF*10.LT.NF) NF=NF+1
NP=NF/10 & IF(NP*10.LT.NP) NP=NP+1
IF(J1.LT.J2) GO TO 5
PRINT 990, (PLABEL(I),I=1,NP), (FRAME(I),I=1,NF)
990 FORMAT(*HANDWITH FOR PLOTS TOO NARROW/*/NU PLUT GENERATED FOR
* 110A10)

71
APPENDIX H

RETURN
5 NPL=J2-J1+1
6 IF(NPL.EE.NSPECT) GO TO 100
7 MOVE PLOTTING REGION TO BEGINNING OF ARRAY
8 DO 4002 I=J1,J2
9 J=I-J1+1
10 XPLOT(J)=XPLOT(I)
11 CONTINUE
12 DO 4003 I=J1,J2
13 J=I-J1+1
14 RI(J)=RI(I)
15 CONTINUE
16 CONTINUE
17 GO TO (1C5,105,106,106),ILOGP1
18 C LINEAR VERTICAL SCALE
19 1C5 CALL ASCAle(RI,100.,NPL,10.,10.)
20 YMIN=RI(NPLOT+1) $ YMAX=YMIN+10.9RI(NPLOT+2)
21 GO TO 107
22 C LOG VERTICAL SCALE
23 106 SMAX=-10C.
24 DO 108 I=1,NPL $ IF(RI(I)) 112,112,113
25 RT(I)=RI(I)*10. $ GC TO 108
26 113 RI(I)=ALOG(RI(I)) $ SMAX=AMAX1(RI(I),SMAX)
27 CONTINUE
28 I MAX=SMAX
29 IF (I MAX.LT.100.) 114,114,115
30 115 IMAX=IMAX-5
31 IMAX=100. $ IMIN=IMIN+5 $ YMAX=IMAX $ RI(NPLOT+2)=5
32 DO 114 I=1,NPL
33 IF(RI(I).LT.YMIN) RI(I)=YMIN
34 CONTINUE
35 107 CONTINUE
36 IF(IF.EQ.0) GO TO 24
37 IF(NPL.EE.256) NPL=256
38 FFID(1)=YLABEL(1) $ FFID(2)=YLABEL(2) $ FFID(3)=FRAME(1)
39 FFID(4)=FRAME(2) $ FFID(5)=FRAME(3)
40 IF(ILOG.GT.1) ENCODE(54,101,IDENT) PLABEL
41 IF(ILOG.LE.1) ENCODE(60,92,IDENT) PLABEL
42 902 FORMAT(5X,5A10,5X)
43 CALL FANFOLD(RI,NPL,1,NPL,IDENT,1*1*,1*,YMAX,YMIN,FFID,NEW+2,11320,XPLOT,YFREQUENCY)
44 901 FORMAT(*LOG*5A10)
45 24 RETURN
46 END
47 SUBROUTINE FANFOLD(PLOT,NP,K,NF,NMAX,IDENT,CHAR,PNORM,PMAX,PMIN,1
48 YLABEL,NYL,LINE,WRITEX,XARRAY,XLABEL)
49 DIMENSION PLOT(NMAX,1),CHAR(1),IDENT(5,1),PNORM(1),YLABEL(1)
50 PMAX(1),PSCALE(10),PLINE(12*),XARRAY(1)
51 IFLINE=128) 7,7,8
52 7 NCHAR=104 $ GO TO 9
53 8 NCHAR=124
APPENDIX H

9 CONTINUE
IF(NF.GT.10) NF=10
PRINT 900, (YLABE1(I), I=1,NYL)
900 FORMAT(*1X*48X*10)
IF(NF.EQ.0) GO TO 10
PRINT 907, (XLABEL,J=1,6), (J,XARAY(J), J=1,NPT,K)
907 FORMAT(/4X*THE CODE SCALE IS AS FOLLOWS*/5X*CODE *AIC,4(I
10 CONTINUE
NSKIP=0
IF(NPT/K-15) 16,16,17
16 NSKIP=2 $ GO TO 13
17 IF(NPT/K-25) 18,18,13
18 NSKIP=1
13 CONTINUE
DO 1 =1,NF
IF(PMIN(IN), PMAX(I)) GO TO 1
PMIN(I)=PMAX(I)=PLOT(I,1)
DO 2 J=1,NPT,K
PMIN(I)=MIN(PMIN(I),PLOT(J,I))
PMAX(I)=MAX(PMAX(I),PLOT(J,I))
2 CONTINUE
1 PSCLAE(I)=(PMAX(I)-PMIN(I))*PNORM(I)/(NCHAR-4)
PRINT 901, (I,CHAP(I), (IDEN(J,1), J=1,5), PNORM(I), PMAX(I), PMIN(I),
1PSCLAE(I), I=1,NF)
901 FORMAT(///53X*PLG DESCRIPTION*/ FUNCTION*4X*SCALE*/ NO.*4X*C
1CHARACTER*5X*IDENTIFICATION*3X*FUNCTION*9X*SCALE*/ NCHAR*5X*MINUM*6X*RES
2UTION*/(15,YX,A1,5X,5A1G1,4E15.5))
PLINE(2)=1H \$ PLINE(NCHAR)=1H \$ PLINE(1)=1H
IF(NCHAR.EQ.124) PRINT 902
IF(NCHAR.EQ.104) PRINT 905
902 FORMAT(//20X*1 1 2 2 3
1 3 4 4 5 5 6 6 7 7 8 8 9 9 1
20 10 11 11 12*10X*0.....50....6........0...........9.....0....
30.....50.....50.....50.....0...........9.....0...........9.....0...........
+5.....5.....5.....0*)
905 FORMAT(//20X*1 1 2 2 3
1 3 4 4 5 5 6 6 7 7 8 8 9 9 1
20*10X*0.....50....50....50....50....50....50....50....50....50....50....50
30.....50.....50.....50.....0...........9.....0...........9.....0...........
1IF(NSKIP.EQ.0) GO TO 20
DO 21 I=1,NSKIP
21 PRINT 908
908 FORMAT(1H )
20 CONTINUE
DO 3 J=1,NPT,K
30 5 I=4,NCHAR
3 PLINE(I-1)=1H
DO 4 I=1,NF
P=(PLOT(J,I)-PMIN(I))/PSCLAE(I)
4 P=P+3.5

73
APPENDIX H

IF(IP.LE.2 .OR. IP.GE.NCHAR) GO TO 4
IF(PLINE(IP).NE.1H) GO TO 6
PLINE(IP)=CHR(1)
GO TO 4
6 PLINE(IP)=1HX
4 CONTINUE
PRINT 903, J, (PLINE(I), I=1,NCHAR)
903 FORMAT(18,124,1)
IF(NSKIP.EQ.0) GO TO 3
UN 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...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...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...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...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...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...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...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...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...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...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...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...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...5.0...5.0...5.0...5.0...5.0...5.0...5.
APPENDIX H

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 * .... LET SUM2 = 2*K2, SUMF = 3*K3 +
+ .... AND NF = K3 + K5 + .... THE TIME TAKEN BY A MULTI-
DIMENSIONAL TRANSFORM ON THESE NT DATA IS T = TO + NT**2(T
+SUM2+T3*SUMF+T4*MF). ON THE CDC 3300 (FLOATING POINT ADD
OF SIX MICROSECONDS), T = 3000 + NT**2(1500+43*SUM2+68*SUMF +
320*MF) MICROSECONDS ON COMPLEX DATA. IN ADDITION, THE
ACCURACY IS GREATLY IMPROVED, AS THE RELATIVE ERROR IS
BOUND BY 3*2**(B-SUM(FACTOR(J)**1.5)), WHERE B IS THE NUM
OF BITS IN THE FLOATING POINT FRACTION AND FACTOR(J) ARE THE
PRIME FACTORS OF NT**2.

THE DISCRETE FOURIER TRANSFORM PLACES THREE RESTRICTIONS UPON
DATA:
1. THE NUMBER OF INPUT DATA AND THE NUMBER OF TRANSFORM VAL
MUST BE THE SAME.
2. BOTH THE INPUT DATA AND THE TRANSFORM VALUES MUST BE
EQUISpaced POINTS IN THEIR RESPECTIVE DOMAINS OF TIME AND
FREQUENCY. CALLING THESE SPACINGS DELTAT AND DELTAF, IT MUST
BE TRUE THAT DELTAF=2*PI/(NN*DELTA). OF COURSE, DELTAT 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.
DIMENSION DATA(32,25,13),WORK(56),NN(3)
COMPLEX DATA
DATA NN/32,25,13/
DO 1 J=1,32
DO 1 K=1,13
DATA (I,J,K)=COMPLEX VALUE
CALL FOURT(DATA,NN,-1,1,WORK)

EXAMPLE 2. ONE-DIMENSIONAL FORWARD TRANSFORM OF A REAL ARRA
LENGTH 64 IN FORTRAN II.
DIMENSION DATA(2,64)
DO 2 J=1,64
DATA(1,J)=REAL PART
2 DATA(2,J)=0.
CALL FOURT(DATA,64,1,-1,0,0)

DIMENSION DATA(2),NN(1),IFACT(32),WORK(1)
WI=1.00
WR=1.00
WSTPF=1.00
WSPI=1.00
TWDP=6.283185307
IF (NDIM-1)1920,1,1
APPENDIX H

1  \( \text{NTOT} = 2 \)
2  \( \text{DO 2 IDIM=1,NDIM} \)
3  \( \text{IF(NDIM(1))=920,920,2} \)
4  \( \text{NTOT=NTOT*NDIM(1)} \)
5  \( \text{C} \)
6  \( \text{C MAIN LOOP FOR EACH DIMENSION} \)
7  \( \text{C} \)
8  \( \text{NP1}=2 \)
9  \( \text{DO 9 I=1,NDIM} \)
10  \( \text{N=N*DIM(I)} \)
11  \( \text{NP2=NP1*N} \)
12  \( \text{IF(N-1)=920,920,5} \)
13  \( \text{C} \)
14  \( \text{C FACTOR N} \)
15  \( \text{C} \)
16  \( \text{M=N} \)
17  \( \text{NTWO=NP1} \)
18  \( \text{IF=1} \)
19  \( \text{DIV=2} \)
20  \( \text{IQUOT=M/DIV} \)
21  \( \text{REM=M-1*DIV*IQUOT} \)
22  \( \text{IF(IQUOT-10IV15C,11,11} \)
23  \( \text{IF(REM)=29,12,2C} \)
24  \( \text{NTWO=NTWO+NTWO} \)
25  \( \text{C}=\text{IQUOT} \)
26  \( \text{GO TO 1C} \)
27  \( \text{C} \)
28  \( \text{IQUOT=M/DIV} \)
29  \( \text{REM=M-1*DIV*IQUOT} \)
30  \( \text{IF(IQUOT-10IV16C,31,31} \)
31  \( \text{IF(REM)=40,32,4C} \)
32  \( \text{IFACT(IF)=DIV} \)
33  \( \text{IF=IF+1} \)
34  \( \text{M=IQUOT} \)
35  \( \text{GO TO 3C} \)
36  \( \text{C} \)
37  \( \text{IQUOT=M/DIV+2} \)
38  \( \text{GO TO 3C} \)
39  \( \text{C} \)
40  \( \text{IF(REM)=60,51,6C} \)
41  \( \text{NTWO=NTWO+NTWO} \)
42  \( \text{GO TO 7C} \)
43  \( \text{C} \)
44  \( \text{FACT(IF)=M} \)
45  \( \text{C SEPARATE FOUR CASES}-- \)
46  \( \text{C} \)
47  \( \text{C 1. COMPLEX TRANSFORM OR REAL TRANSFORM FOR THE 4TH, 5TH,} \)
48  \( \text{C DIMENSIONS} \)
49  \( \text{C} \)
50  \( \text{C 2. REAL TRANSFORM FOR THE 2ND OR 3RD DIMENSION, METHOD--} \)
51  \( \text{C TRANSFORM HALF THE DATA, SUPPLYING THE OTHER HALF BY C} \)
52  \( \text{C JUGATE SYMMETRY} \)
53  \( \text{C} \)
54  \( \text{C 3. REAL TRANSFORM FOR THE 1ST DIMENSION, N ODD, METHOD--} \)
55  \( \text{C TRANSFORM HALF THE DATA AT EACH STAGE, SUPPLYING THE C} \)
56  \( \text{C HALF BY CONJUGATE SYMMETRY} \)

76
REAL TRANSFORM FOR THE 1ST DIMENSION, N Even. METHOD-
TRANSFORM A COMPLEX ARRAY OF LENGTH N/2 WHOSE REAL PAR
ARE THE EVEN NUMBERED REAL VALUES AND WHOSE IMAGINARY
ARE THE ODD NUMBERED REAL VALUES. SEPARATE AND SUPPLY
THE SECOND HALF BY CONJUGATE SYMMETRY.

NUN2=NP1*(NP2/NTW0)
ICASE=1
IF(IDIM-4)71,90,90
IF(IDIM)72,90,90
ICASE=2
IF(IDIM-1)73,73,90
ICASE=3
IF(NTW0-NP1)90,90,90
ICASE=4
NTWO=NTWO/2
N=N/2
NP2=NP2/2
NTOT=NTOT/2
I=3
DO 80 J=2,NTOT
DATA(J)=DATA(I)
80 I=I+2
TIPNG=NP1
IF(ICASE=2)100,55,100
TIPNG=NP0*(1+NPREV/2)
CO SHUFFLE THE FACTORS OF TWO IN N, AS THE SHUFFLING
CAN BE DONE BY SIMPLE INTERCHANGE, NO WORKING ARRAY IS NEEDED.

IF(NTWO-NP1)600,60,110
NP2HF=NP2/2
J=1
DO 150 I2=1,NP2,NUN2
IF(J=12)120,130,130
120 I1MAX=I2+NUN2-2
DO 125 I1=I2,I1MAX,2
DO 125 I3=I1,NTCT,NP2
J3=J+I3-12
TEMPR=DATA(I3)
TEMP=DATA(I3+1)
DATA(I3)=DATA(J3)
DATA(I3+1)=DATA(J3+1)
DATA(J3)=TEMPR
125 DATA(J3+1)=TEMP
130 M=NP2HF
140 IF(J-M)150,150,145
145 J=J-M
M=M/2
150 J=J+M
APPENDIX H

C MAIN LOOP FOR FACTORS OF TWO. PERFORM FOURIER TRANSFORMS OF
C LENGTH FOUR, WITH ONE OF LENGTH TWO IF NEEDED. THE TWIDDLE
C W = \exp\left(i \text{SIGN} \cdot 2 \cdot \pi \cdot \text{SIGN} \cdot \text{SQRT}(-1) \cdot \text{M}/(4 \cdot \text{MAX})\right). CHECK FOR \text{SIGN} \cdot \text{SQRT}(-1) \cdot \text{CONJUGATE}(W).
C
C NON2T = NON2 + NON2
C IPAR = \text{NON2}/\text{NP1}
C IF(IPAR-2) GO TO 310
C IPAR = IPAR/4
C GO TO 310
C 310 DO 340 I = 1, I1RNG, 2
C 320 J3 = I, NCR2, NP1
C K1 = J3, NTOT, NON2T
C K2 = K1 + NON2
C TEMP = DATA(K2)
C TEMPI = DATA(K2 + 1)
C DATA(K2) = DATA(K1) - TEMP
C DATA(K2 + 1) = DATA(K1 + 1) - TEMPI
C DATA(K1) = DATA(K1 + 1) + TEMP
C DATA(K1 + 1) = DATA(K1 + 1) + TEMPI
C 340 IF(MMAX = NON2) 350, 330, 320
C 330 DO 350 L = NON2, LMAX, NON2T
C 340 M = L - \text{NON2T}
C IF(MMAX = NON2) 420, 400, 410
C 400 WR = \cos(\Theta)
C WI = \sin(\Theta)
C WSTPR = -2 * \text{NP1} * WI
C WSTPI = 2 * \text{WR} * WI
C 410 DO 450 I = 1, I1RNG, 2
C 420 J3 = I - IPAR * \text{M}
C IF(MMAX = NON2) 430, 420, 410
C 430 KMIN = J3 + IPAR * \text{M}
C 440 KMIN = J3
C 450 KDIF = IPAR * MMAX
C KSTEP = 4 * KDIF
C DO 520 K1 = K Min, NTOT, KSTEP
C K2 = K1 + KDIF
C K3 = K2 + KDIF
C K4 = K3 + KDIF
C IF(MMAX = NON2) 460, 440, 430
APPENDIX H

460  U1R=DATA(K1)+DATA(K2)  
461  U1I=DATA(K1+1)+DATA(K2+1)  
462  U2R=DATA(K3)+DATA(K4)  
463  U2I=DATA(K3+1)+DATA(K4+1)  
464  U3R=DATA(K1)-DATA(K2)  
465  U3I=DATA(K1+1)-DATA(K2+1)  
466  IF(ISIGN)470,475,475  
467  U4R=DATA(K3+1)-DATA(K4+1)  
468  U4I=DATA(K4)-DATA(K3)  
469  GO TO 510  
470  U4R=DATA(K4+1)-DATA(K3+1)  
471  U4I=DATA(K3)-DATA(K4)  
472  GO TO 510  
473  T2R=W2R*DATA(K2)-W2I*DATA(K2+1)  
474  T2I=W2R*DATA(K2+1)+W2I*DATA(K2)  
475  T3R=WR*DATA(K3)-WI*DATA(K3+1)  
476  T3I=WR*DATA(K3+1)+WI*DATA(K3)  
477  T4R=W3R*DATA(K4)-W3I*DATA(K4+1)  
478  T4I=W3R*DATA(K4+1)+W3I*DATA(K4)  
479  U1R=DATA(K1)+T2R  
480  U1I=DATA(K1+1)+T2I  
481  U2R=T3R+T4R  
482  U2I=T3I+T4I  
483  U3R=DATA(K1)-T2R  
484  U3I=DATA(K1+1)-T2I  
485  IF(ISIGN)490,500,500  
486  U4R=T3I-T4I  
487  U4I=T4I-T3I  
488  GO TO 510  
489  U4R=T*I-T3I  
490  U4I=T3R-T4I  
491  DATA(K1)=U1R+U2R  
492  DATA(K1+1)=U1I+U2I  
493  DATA(K2)=U3R+U4R  
494  DATA(K2+1)=U3I+U4I  
495  DATA(K3)=U1R-U2R  
496  DATA(K3+1)=U1I-U2I  
497  DATA(K4)=U3R-U4R  
498  DATA(K4+1)=U3I-U4I  
499  K4I=W4*(KMIN-J3)+J3  
500  KDIF=KSTEP  
501  IF(KDIF=NP21450,530,530  
502  CONTINUE  
503  M=MMAX-M  
504  IF(ISIGN)540,550,550  
505  TEMPR=WR  
506  WR=WI  
507  WI=TEMPR  
508  GO TO 560  
509  TEMPR=WR  
510  WR=WI  

79
APPENDIX H

WI = TEMPP
560 IF (M - LMAX) 565, 565, 410
565 TEMPR = WR
WP = WR * WSTPR - I * WSTPI * WR
570 WI = WI * WSTPR + TEMPR * WSTPI + WI
IPAR = 3 - IPAR
MMAX = MMAX + MMAX
GO TO 360
C
C MAIN LOOP FOR FACTORS NOT EQUAL TO TWO. APPLY THE TWIDDLE F
C W = EXP(I * SIGN * 2 * PI * SQRT(-1) * (J2 - 1) * (J1 - J2) / (N2 * IFACT(IF))), THEN
C PERFORM A FOURIER TRANSFORM OF LENGTH IFACT(IF), MAKING USE
C CONJUGATE SYMMETRIES.
C
600 IF (NTWO - NP2) 605, 760, 700
605 IFPI = NON2
IF = 1
NP1HR = NP1/2
610 IFP2 = IFP1 / IFACT(IF)
J1RNG = NP2
IF (CASE = 3) 612, 612, 612
611 J1RNG = (NP2 * IFP1) / 2
J2STP = NP2 / IFACT(IF)
J1R2 = (J2STP + IFP2) / 2
J2MIN = 1 + IFP2
IF (IFP1 - NP2) 615, 615, 615
615 J2 = J2MIN, IFPI, IFP2
THETA = -TWOP1 * FLCAT(J2 - 1) / FLOAT(NP2)
IF (SIGN) 625, 625, 620
620 THETA = -THETA
625 SINTH = SIN(THETA / 2.)
WSTPR = -2.0 * SINTH * SINTH
WSTPI = SIN(THETA)
WR = WSTPR + 1.
WI = WSTPI
J1MIN = J2 + IFPI
630 J1 = J1MIN, J1RNG, IFPI
J1MAX = J1 + J1RNG - 2
630 J1 = J1, J1MAX + 2
DO 630 J1 = J3, NTCTJ, NP2
DO 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
TEMP = WR
WR = WR * WSTPR - I * WSTPI * WR
635 WI = TEMPR * WSTPI + WI * WSTPR + WI
640 THETA = -TWOP1 / FLOAT(IF)
IF (SIGN) 645, 645, 645
645 THETA = -THETA
APPENDIX H

650  SINTH=SIN(THETA/2.)  0341
655  WSTPR=-2.*SINTH*SINTH  0342
650  WSTPI=SIN(THETA)  0343
650  KSTEP=2*N/FACT(IF)  0344
655  KRANG=KSTEP*(FACT(IF)/2)+1  0345
660  DO 698 I1=1,IFRNG,2  0346
660  DO 698 J3=11,NTCT,NO  0347
650  DO 690 KMIN=1,KRANG,KSTEP  0348
650  J1MAX=I3+1RFNG-IFP1  0349
650  DO 680 J1=13,J1MAX,IFP1  0350
660  J3MAX=J1+IFP2-NP1  0351
660  DO 680 J3=J1,J3MAX,NP1  0352
650  J2MAX=J3+IFP1-IFP2  0353
660  K=KMIN+(J3-J1*(J1-I3)/FACT(IF))/NP1*F  0354
660  IF(KMIN-I)655,655,609  0355
660  SUMR=0.  0356
660  SUM=0.  0357
655  DO 660 J2=J3,J2MAX,IFP2  0358
660  SUMR=SUMR+DATA(J2)  0359
660  SUMI=SUMI+DATA(J2+1)  0360
660  WORK(K)=SUMR  0361
660  WORK(K+1)=SUMI  0362
660  GO TO 680  0363
665  KCONJ=K+2*(N-KMIN+1)  0364
665  J2=J2MAX  0365
665  SUMR=DATA(J2)  0366
665  SUMI=DATA(J2+1)  0367
665  OLDSSR=0.  0368
665  OLDSSI=0.  0369
665  J2=J2-IFP2  0370
670  TEMPR=SUMR  0371
670  TEMPI=SUMI  0372
670  SUMR=WOWR*SUMR-OLDSSR+DATA(J2)  0373
670  SUMI=WOWR*SUMI-OLDSSI+DATA(J2+1)  0374
670  OLDSSR=TEMP  0375
670  OLDSSI=TEMPI  0376
670  J2=J2-IFP2  0377
670  IF(J2-J3)675,675,670  0378
675  TEMPR=WR*SUMF-OLDSSR+DATA(J2)  0379
675  TEMPI=WI*SUMI  0380
675  WORK(K)=TEMPF-TEMPI  0381
675  WORK(KCONJ)=TEMPF+TEMPI  0382
675  TEMPR=WR*SUMF-OLDSSI+DATA(J2+1)  0383
675  TEMPI=WI*SUMI  0384
675  WORK(K+1)=TEMPF*TEMPI  0385
675  WORK(KCONJ+1)=TEMPF*TEMPI  0386
680  CONTINUE  0387
680  IF(KMIN-I)685,685,686  0388
685  WR=WSTPR+1.  0389
685  WI=WSTPI  0390
685  GO TO 690  0391
APPENDIX H

686  \text{TEMPR=WR}
687  \text{WR=WP*WSTPR+WI*WSTPI+WR}
688  \text{WI=TEMPR*WSTPI+WI*WSTPR+WI}
690  \text{THWR=WR+WR}
691  \text{IF(I CASE=3))692,691,692}
692  \text{IF(IFPI-NP2)*655,642,692}
693  \text{K=1}
694  \text{12MAX=I3+NP2-NP1}
695  \text{DO 697 J3=I3,12MAX,NP1}
696  \text{DATA(J3)=WORK(K)}
697  \text{DATA(J3+1)=WORK(K+1)}
698  \text{K=K+2}
699  \text{GO TO 698}

C  COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N ODD, BY CG
C  JUGATE SYMMETRIES AT EACH STAGE.
C
695  \text{J3MAX=I3+IFP2-NP1}
696  \text{DO 699 J3=I3,J3MAX,NP1}
697  \text{J2MAX=J3+NP2-J2STP}
698  \text{DO 697 J2=J3,J2MAX,J2STP}
699  \text{J1MAX=J2+J1RG2-IFP2}
700  \text{J1CNJ=J3+J2MAX+J2STP-J2}
701  \text{DO 697 J1=J2,J1MAX,IFP2}
702  \text{K=1+J1-J3}
703  \text{DATA(J1)=WORK(K)}
704  \text{DATA(J1+1)=WORK(K+1)}
705  \text{IF(J1-J2)*697,627,696}
706  \text{DATA(J1CNJ)=WORK(K)}
707  \text{DATA(J1CNJ+1)=-WORK(K+1)}
708  \text{CONTINUE}
709  \text{IF=IF+1}
710  \text{IFPI=IFP2}
711  \text{IF(IFPI-NP1)*700,701,702}

C  COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N EVEN, BY C
C  JUGATE SYMMETRIES.
C
700  \text{GO TO (90C,8GC,5C0,701),ICASE}
701  \text{NH A LF=N}
702  \text{N=N+N}
703  \text{THETA=-TWQPI/FLCAT(N)}
704  \text{IF(I SIGN=703,7C2,702}
705  \text{THETA=-THETA}
706  \text{SINTH=SIN(THETA/2.)}
707  \text{WSTPR=-2.*SINTH*SINTH}
708  \text{WSTPI=SIN(THETA)}
709  \text{WK=WSTPR+1.}
710  \text{WI=WSTPI}
711  \text{IMIN=3}
712  \text{GO TO 698}
713  \text{IF=IF*1}
714  \text{IFP1=IFP2}
715  \text{IF(IFP1-MP1)*701,7C0,702}

82
APPENDIX H

JMIN=2*NHALF-1
GO TO 725

710 J=JMIN
DO 720 I=IMIN,NTOT,NP2
SUMR=(DATA(I)+DATA(J))/2.
SUMI=(DATA(I+1)+DATA(J+1))/2.
DIFR=(DATA(I)-DATA(J))/2.
DIFI=(DATA(I+1)-DATA(J+1))/2.
TEMPR=WR*SUMI*WI*DIFR
TEMPI=WI*SUMI*WR*DIFR
DATA(I+1)=DIFI*TEMPI
DATA(J)=SUMR-TEMPR
DATA(J+1)=-DIFI*TEMPI

720 J=J+NP2
IMIN=IMIN+2
JMIN=JMIN-2
TEMPR=WR
WP=WR*WSTPR-WI*WSTPI+W
WI=TEMPR*WSTPI+WI*WSTPP+W

725 IF(I-MIN-JMIN) 71C, 730, 740
730 IF(ISIGN) 731, 74C, 740
731 DO 735 I=IMIN,NTOT,NP2
735 DATA(I+1)=-DATA(I+1)
740 NP2=NP2+NP2
NTOT=NTOT+NTOT
J=NTOT+1
IMAX=NTOT/2+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) 150, 76C, 760
760 DATA(J)=DATA(I-MIN)-DATA(IMIN+1)
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, 775, 765
775 DATA(J)=DATA(I-MIN)+DATA(IMIN+1)
DATA(J+1)=O.
IMAX=IMIN
GO TO 745

780 DATA(1)=DATA(1)+DATA(2)
DATA(2)=O.
GO TO 900
APPENDIX H

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

800 IF(I1RNQ-NP1) 805 900, 900
805 DO 860 I3=1,NTCT,NP2
I2MAX=I3+NP2-NP1
DO 860 I2=I3,I2MAX,NP1
IMIN=I2+I1RNQ
IMAX=I2+NP1-2
810 IF(I2-I3(20,82C,810
JMAX=2*I3+NP1-IMIN
820 IF(I2-I3) 850,E5C,830
JMAX=JMAX*NP2
830 J=JMAX+NP0
DO 840 I=IMIN,IMAX,2
DATA(I)=DATA(J)
DATA(I+1)=-DATA(J+1)
840 J=J-2
850 J=JMAX
860 J=J-NPO
2 END OF LOOP ON EACH DIMENSION

OVERLAY(PATS,1,C)
PROGRAM P.PAOIM
.COMMON/BLK1/STAPTT,ITFMT,NBLK,IPDw2,NCH,NPRINT,IPLOUTA,IPLOUTC,OFFSC
1AL,DELTAT,SN,NRSKIP,LAG,NCRROSS,ICROSS(2,20),NCHP,YLABEL(2),IWINDOW
1,F1,F2,ITYPESP,NCH,SCALFAC,IFF,LAG1,LAG2
.COMMON/BLK2/ICH(14),CHSUM(14),NFSKIP,LAG1,LAG2
.COMMON/BLK3/NPT,TMAX,NPTO2,NSPCT,DEL2,N64,NPTO28
.COMMON/BLK4/NMAX,NCHMAX,NBCMAX
.COMMON/BLK5/PCTC,NBINS,DMAX(14),DMIN(14),OBIN(14),BINS(10,14)
1,CHSAM
.COMMON/BLK6/IATUO(14),ICRSP(20),ICRCUR(20),ITRA(20),IICOH(20)
.COMMON/BLK7/NFILT,FREQF(50),WHTF(50)
.COMMON/BLK8/ITFMT,NSKIP,SN,DELTAT,STARTT,OFFSCAL,NCH
1,SCALFAC,NPTO,READ,IAUTO,ICRSP,NCRROSS,ICROSS,ICRSP,ICRCUR
2,ITRA,ICOH,LAG,1MBED3W,ITYSP,PRINT,IPLOUTA,IPLOUTC,FL,F2,LAG1,LAG2
3,PCTC,NBINS,DMAX,DMIN,INZERJ,NFILT,PREFW,WHITF,IFILTER

84
APPENDIX H

DATA NFILT,FPEOF,WHGT,FIFILTER/0.,100.*O,140./
DATA I0AUTODSP,IALTOCO,ICRSP,ICRCUK,ITRA,ICOM/108.*0/
DATA IF,FALG1,LAG2/1.,0./
DATA NSKIP/0/
DATA ITFMT,STARTT,ICH,NRSKIP,SCALFAC,OFFSCAL,LAP,NCROSS,ICROSS,
INPRINT,IPLOTC,IPLCTA,1WINDOW,F1,F2,1TYPESP/
220.,15.*O,141.,1.E+6,42*0,100.,1.,1.,2.,2000.,2/
DATA I0,CHAN/2HID,4HRCH/
DATA NCTC/90./
DATA INZP,NOINS,OMAX,OMIN/2*0,28*0./

C C READ AND PRINT NAMELIST INPUT
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)
50C2 FORMAT(8A10)
PRINT 5001, YLABEL
PRINT 5003, (I,TRACK(I),I=1,NCH)
50C3 FORMAT(//CASE ID/5X,1H*2A10,1H*)
50C4 FORMAT(//-CHANNEL ID/10*/3X*NO.*,*/(15,4X10))
PRINT 5004
5004 FORMAT(*)

C C CHECK INPUT DATA, PRINT ERROR MESSAGES
C
NPT=NREAD
NBLK=NPT/NPT/NPT
IF(NCH.LE.NCHMAX) GOTO 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 DO 312 IFSKIP=1,NFSKIP
313 READ(I1) SKIPREC
IF(EDF,1) 312,313
312 CONTINUE * PRINT 904, NFSKIP
904 FORMAT(//6H * * *,15,20H FILES SKIPPED * * *)
310 IF(I1YPESP.LT.3) GOTO 9C
NCROSS=0
DO 91 I=1,NCH
IF(IATUCO(I).GT.0) IATUCO(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(IAUTOCO(I).NE.0) GO TO 116
  ICORR=1
  IF(IAUTOSPI(I).NE.0) IAUTOSPI(I)=-1
116 CONTINUE
IF(NCROSS) 118,116,108.
108 DO 110 I=1,NCROSS
  J1=ICROSS(I,1) $ J2=ICROSS(I,2)
  IF(IAUTOSP(J1).NE.0) IAUTOSP(J1)=-1
  IF(NCROSS) 117,110
117 ICORP=1
211 IF(ICRSP(I).NE.0) ICRSP(I)=-1
110 CONTINUE
118 IF(ICROR.EQ.1.AND.INZERQ.EQ.0) PRINT 907
907 FORMAT(// * * YOU MAY HAVE CIRCULAR ERROR IN YOUR CORRELATIONS
INS BECAUSE YOU HAVE NOT ASKED FOR ZERO INSERTION * * *)
  IF(INZERQ.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
  CHSUND(IK)=CHSUMSQ(IK)=0.
  CHSUM(IK)=0.
35 CONTINUE
IPOW2=0 NTEMP=NPT
200 IF(2**IPOW2.NE.NPT) GO TO 200
201 IPOW2=IPOW2+1 NTEMP=NTEMP/2 GO TO 200
202 IF(NTEMP.LT.NMAX) GO TO 203
203 IF(NTEMP.LT.200) 200,200,202
204 IF(NTEMP.LT.200) 200,200,202
207 PRINT 207
207 FORMAT(// * BLOCK SIZE IS A POWER OF TWO. FAST FOURIER TRANSFORM W
ILL 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
ILL BE USED*)
  IF(NPT.LE.NMAX) GO TO 205
  PRINT 206
206 FORMAT(// * BLOCK SIZE TOO LARGE FOR DIMENSIONS PROVIDED. JOB TERMIN
INATED*)
APPENDIX H

STOP 20b
205 CONTINUE
NPT02=NPT/2.
NSPCT=NPL0T=NPTC2 & IF(NPRINT.GT.NPT02) NPRINT=NPT02
TMAX=NPT*DELTAT
DELF=1./TMAX
NPT0128=NPT/128
IF(INZERG.EQ.O.CR.LAP.EQ.C) GO TO 111
PRINT 902
LAP=O
932 FORMAT(//, U.50 PERCENT OVERLAP ON ZERO INSERTION RUNS/* INPUT
DATA ALTERED, LAP=O */
111 CONTINUE
IF(LAP.EQ.O) NBLK=2*NBLK-1
NCHP=O
DO 80 I=1,NCH
IF(IAUTOSP(I)) NCHP,NCH+1
80 CONTINUE
IF(NCHP) 32,72,63
82 IFORMAT(///, INPUT INDICATES NO CHANNELS TO BE PROCESSED, CASE ENDED
DATA ALTERED, LAP=O *)
STOP 101
83 CONTINUE
IF(NCHP.NE.O,:.NBLK.LE.NBCMAX) GO TO 306
PRINT 307, NCHP,NBLK,NBCMAX
307 FORMAT(///, NO. OF CHANNELS TO BE PROCESSED (NCHP)=/* NO. OF YL
LOCKS (NBLK) =/* NO EXECU
2TION ENDED. CHANGE DIMENSIONS TO FIT YOUR CASE AND RERUN*)
STOP 07
306 CONTINUE
N64=NCHP*64
C COMPUTE ACCURACY MEASUREMENT OF SPECTRAL ESTIMATORS
C
IF(LAP.EQ.O) NDQF=2*NBLK & IF(LAP.NE.O) NDQF=1.6364*(NBLK-1)
CALL CSQ((10G.*PCTC)/2.,NDQF,BL,ICQOE)
BL=10G.*BL & BU=10G.*BU
PRINT 903, PCTC, BL, BU
903 FORMAT(///, 0H * * * ACCURACY MEASUREMENT OF SPECTRAL ESTIMATORS
* * * ASSUMING NORMALITY OF DATA, USER CAN BE*F5.* CERTAIN*/
THAT THE SPECTRAL ESTIMATE IS WITHIN THE BOUNDS OF */F5.* PERCENT
AND*F5.* PERCENT OF THE TRUE SMOOTHED SPECTRUM*)
C COMPUTE CRITICAL VALUE OF CHISQUARE FOR NORMALITY TEST
C
IF(INBINS.GT.100) NIRS=10G
IF(NBINS*4.O) GO TO 112
DO 112 I=1,NCHP
112
APPENDIX H

\[ K = ICHAN(I) \]

1. \[ DBIN(I) = (OMAX(K) - OMIN(K))/NBINS \]
   \[ JD = 1, NBINS \]

112 \[ BINS(J, I) = 0. \]

\[ NBINS3 = NBINS - 3 \]

CALL CSQ(PCTC, NBINS3, BL, BU, ICODE)

CHISQ = NBINS3/BL

113 CONTINUE

\[ IF(NFILTP.EQ.0) GO TO 114 \]

\[ IF(NFILTP.LE.5C) GO TO 115 \]

PRINT 905

905 FORMAT(///55H * * * INPUT ERROR, NFILTP GT 50, EXECUTION ENDED * * 1 *)

STOP 96

115 PRINT 906

906 FORMAT(///126H * * * SPECTRAL FILTERING OPTION SELECTED. CORRELATIONS WILL BE CALCULATED FROM FILTERED SPECTRA FOR CHANNELS SPECFIED 2. * * *))

114 CONTINUE

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]

\[ C \]
APPENDIX H

GO TO 3
1 BU=N/CHISO
3 Q=1.-Q
C FIND LOWER BOUND
CALL ITP2(1,CHISO,1000,1,FUNC,1.E-6,1.E-6,1,1000,1,1C,1C06)
IF(ICODE=3) 4,4,5
5 BL=0.
GO TO 6
4 BL=N/CHISO
6 RETURN
END
FUNCTION FUNC(CHISQ)
COMMON/PROBF/A,C
DIMENSION H60(15)
DATA H6C/-.0118, -.0067, -.0033, -.0010, .0001, .0004, .0007, .0013, .0020, .0026, 
1.0035, .0042, .0050, .0062, .0072/.
1 X=CHISO/2.
FUNCTION=Q-GAMMF(A, X)/GAMMF(A, 0.)
GO TO 3
2 X=((CHISQ/ (2.*A))**(1./3.))-(1.-1./(4.*A)))/SQRT(1./(4.*A))
IX=2..5 $ \text{IF(X}>5.) IX=IX-1 $ IX=IX+8 $ IF(IX) 0,0,6
7 IF(IX=15) 8,8,6
8 IF(X.LT.1) GO TO 6
9 HNU=(30./A)*H60(IX)
GO TO 11
10 HNU=(30./A)*(H6C(IX)*XREM*(H60(IX-1)-H60(IX))/5)
GO TO 11
6 HNU=0.
11 CONTINUE
X=X+HNU
AX=ABS(X)
X2=AX*AX $ X3=AX*X2 $ X4=AX*X3
PMINUS1 =-.5*(L.19654*AX+.11514*X2+.00344*X3+.019527*X4)*(-4.)
1)
4 IF(IX) 4,5,5
IF(IX) 4,5,5
4 FUNCTION=Q-(1.+PMINUS1)
GO TO 3
5 FUNCTION=Q+PMINUS1
3 RETURN
END
OVERLAY(PATS,2,C)
PROGRAM BLOCKS
COMMON CMAIN(1)
COMMON/BLK1/STARTT,ITFMT,NBLK,IPCW2,NC4,NPRINT,IPLOTA,IPLOTc,UFFSC 
14,DELTA,T,SN,NRSKIP,LAP,NCROSS,ICROSS(2,2),NCHP,YNABLEL(2),IWINUW 
1,F1,F2,ITYPESP,1CC,NFSKIP,IFF,LAG1,LAG2 
COMMON/BLK2/ICH(14),CHSUM(14),NCHS(14),NCHS(14),CHSUMSQ(14),SIGMA(14),RMS(14), 
14,MEAN(14),SCALFAC(14),CHSUML(14),TRACK(14),ICHAN(14)
89
APPENDIX H

COMMON/BLK3/NPT,TMAX,NPT02,NSPCT,DELFT,N64,NPT0120
1,INZER0,NREAD
COMMON/BLK5/PCTC,N81NS,OMAX(14),DMIN(14),DDIN(14),BINS(160,14)
1,CHISQ
COMMON/BLK6/ISAVE0,INR1,IXPL0T,IODATA,IZ,ISPECT
COMMON/BLK7/I(11)
COMMON/BLK8/IAUT0SP(14),IAUToCOL(14),ICRPCLG),ICCR0L(2G),IAUT0(20),
1,IC0H(20)
DO 1 NH=1,NBLK
CALL READTPE(NB,CMAIN(IDATA),CMAIN(ISAVE0),)
CALL TRA(NB,CMAIN(IZ),CMAIN(ISAVE0),CMAIN(ISPECT))
1 CONTINUE
RETURN
END

SUBROUTINE READTPE(NB,DATA,SAVE0)
DIMENSION DATA( U,SAVE6*( II
COMMON/BLK1/STARTT,ITFMT,NN2,IW2,SN,NRSKIP,LAP,NK0W10),ICROSS(2,20),NCHP,NLABEL(2),IWINO
1,F1,F2,ITYPESP,WHIN,NSKIP,IFF,LAG1,LAG2
COMMON/BLK2/ICH(14),CHSUM( 14),NFF(14),CHSUMSQ( 14),SIGMA(14),RMS(1
14),MEAN(14),SCALFAC(14),CHSUM( 14),TRACK(14),ICCHAN(14)
COMMON/BLK3/NPT,TMAX,NPT02,NSPCT,DELFT,N64,NPT0120
1,INZER0,NREAD
COMMON/BLK7/NN,IND,KNCH,FNR,II,KK,NI,JJ,IKEC,NREC,ILOC
IF(ITFMT.EQ.3) GO TO 4000
IF(NB-1I 12,12,11
12 GO TO (3001,301C),ITFMT
C ATRAN FORMAT PARAMETERS SET UP
3001 IF(NRSKIP) 105,105,106
106 GO TO 107 I=1,NRSKIP
107 READ(I)
105 CONTINUE
IF(NCH.GT.10) GC TO3002
KCH=20
1NFR=25
GO TO 3003
3002 IF(NCH.GT.20) GC TO 3004
KCH=30
1NFR=17
GO TO 3003
3004 IF(NCH.GT.30) GC TO 3005
KCH=40
1NFR=12
GO TO 3003
3005 IF(NCH.GT.40) GC TO 3006
KCH=50
1NFR=10
GO TO 3003
3006 IF(NCH.GT.100) GC TO 3007
KCH=110
1NFR=4
90
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(NR_SKIP) 3009,3009,108
108 DO 109 I=1,NR_SKIP
109 READ(1)
C FIND STARTING TIME ON TAPE
C
3009 CONTINUE
C
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 08
3014 READ(1) KEY,NN,(DATA(I),I=1,NN)
IF(EOF,1) 3,3C15
3015 DO 1 J=1,NFR
1 JJ=J
IF(DATA(J)-STARTT) 1,4,4
1 CONTINUE
GO TO 3009
3 PRINT 900, STARTT
900 FORMAT(//-- STARTING TIME=12.5, NOT FOUND ON TAPE*)
STOP 08
4 KK=0
NL=JJ
C START READING DATA TO BE PROCESSED
C
GO TO 13
11 NL=JJ+1 $ KK=C
IF(NL,GT,NFR) GO TO 15
BACKSPACE 1
GO TO 16
15 NL=1 $ I=IWD+1
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=T1-1
  IREC=IREC1-1
  GO TO 14
13 IREC=0
14 ILOC=0
10 CONTINUE
  DO 5 J=N1, NFR
    KK=KK+1
  IF(KK-NREAD) 6, 6, 7
  6 ILOC=ILOC+1
    JJ=(ILOC-1)*NCHP
    DO 30 IKK=1, NCHP
      IK=ICHAN(IKK)
      X=DATA(IJ+IK)*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, M64, IREC)
31 CONTINUE
  JJ=J
    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
  NI=1
    NI=IW+1
    GO TO 10
C ONE BLOCK READ, READY TO BE PROCESSED BY FFT
C 7 CONTINUE
APPENDIX H

IF(ILOC.EQ.0) RETURN
IREC=IREC+1
CALL WRITMS(9,SAVEo4,ILOC*NCHP,IREC)
RETURN
C
C RECIU FORMAT
C
4000 NREC=NREAD/64 $ NCHPZ=NCH+2 $ I1=1
I=0
IF(NH-1) 4001,4029
4001 IF(NNSKIP.EQ.0) GO TO 1650
DO 4051 IRSKIP=1,NNSKIP
4051 CALL RECIN(I,2,NN,DATA,1,NCHP2,1)
PRINT 3052, NNSKIP
3052 FORMAT(/I10,* RECORDS SKIPPED*)
CONTINUE
I=I+1
CALL RECIN(I,2,NN,DATA,1,NCHP2,1)
IF(EUF.I) 3,3127
3027 IF(DATA(2)-STARTT) 3050,3222,3022
3022 IF(DATA(1).EQ.SN) GO TO 3035
PRINT 3016
STOP 05
3035 I=2
PRINT 961, I
901 FORMAT(/* STARTT FOUND AT RECORD */I5)
KK=1
DO 3023 IKK=1,NCHP
I=ICHAN(IKK)
XX=DATA(I+2)SCALFAC(I)
IF(ABS(XX)-OFFSCAL) 3030,3031,3031
3030 CHSUM(IJ)=CHSUM(IJ)+XX
3023 SAVE64(IKK)=XX
IREC1=1
NREC=NREAD/64 $ GO TO 3060
3029 IF(IAP) 3061,3061,3062
3061 KK=0 $ I=1 $ IREC=1 $ NREC=NREAD/64 $ GO TO 3060
3062 KK=0
IWRITE=C
IREC=NPTO2/64
NREC=NREAD/64
ILOC=(NPTO2-IREC*64)*NCHP
IREC=IREC+1 $ NWORDS=N64 $ NLEFT=N64-ILOC
4018 IF(IREC-NREC) 4016,4016,4017
4017 NWORDS=(NREAD-NREC*64)*NCHP
NLEFT=NWORDS-ILOC
4016 CALL READMS(9,SAVEo4,NWORDS,IREC)
IREC=IREC+1
KK=KK+NLEFT/NCHP
IF(ILOC.EQ.0) GO TO 4014
APPENDIX H

DO 4006 I=1,NLEFT
4006 SAVE64(I)=SAVE64(I+ILOC)
IF(KK-NPT02) 4007,4012,4013
4007 IF(REC-NREC) 4024,4009,4010
4009 NWORDS=(NREAD-NREC*64)*NCHP
IF(NWORDS LT 1LCC) ILOC=NWORDS
CALL READMS(9,SAVE64(NLEFT+1),ILOC,IREC)
KK=KK*ILOC/NCHP
4014 IF(KK-NPT02) 4011,4020,4021
4011 IWRITE=WRITE+1
CALL WRITIN(9,SAVE64,N64,IWRITE)
GO TO 4018
4012 IF(NLEFT LT N64) GO TO 4019
4022 IWIter=WRITE+1
CALL WRITIN(9,SAVE64,N64,IWRITE)
I1=1
4015 WRITE=WRITE+1
KK=KK+ILOC/NCHP
GO TO 3060
4013 I1=NLEFT/NCHP-(KK-NPT02)+1
GO TO 4015
4019 I1=NLEFT/NCHP+1
GO TO 4015
4020 IF(ILOC*NLEFT*EC*N64) GO TO 4022
I1=ILOC+NLEFT/NCHP+1
GO TO 4015
4021 I1=(NLEFT+ILOC)/NCHP-(KK-NPT02)+1
GO TO 4015
3060 IF(ITFLT.LT.3) GO TO 206
IF(IREC*GT.NREC) GO TO 3046
DO 3024 IREC=IREC,1,NREC
DO 3025 ILOC=I1,64
CALL RECIN(1,2,NN,DATA,1,NCHP,1)
IF(EOF.LT.1) 3063,3034
3034 KK=KK+1
I1=(ILOC-I1)*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(IK)/(KK-I+NB-1)*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(INREC*GT.NREAD) RETURN
3046 NLEFT=NREAD-64*NRLC
NREC=NREC+1
DO 3042 ILOC=I1,NLEFT
94
APPENDIX H

CALL RECIN(1,2,NN,DATA,1,NCHP2,1)
IF(ENF,1) 3003,3041

3041 KK=KK+1
   IJK=(100-1)*NCHP
   DO 3042 IKK=1,NCHP
   IK=ICHAN(IKK)
   XX=DATA(IK+2)
   IF(XX.EQ.OFFSCAL) 3044,3043
   IF(EOF) 3063,3041
   IK=ICHAN(IK)
   XX=DATA(IK+2)
   IK=ICHAN(IK)
   XX=CHSUM(IK)/((KK-1)+(N8-1)*NREAD)
   NOFF(IK)=NOFF(IK)+1
   NOFF(IK)=NOFF(IK)+1
   CALL WRTMS(9,SAVE64,NLEFT*NCHP,NREC)
   RETURN
END

SUBROUTINE TRAN(NH,Z,SAVE64,SPECT)
DIMENSION Z(1),SAVE64(1),SPECT(1)
COMPLEX Z
COMMON/BLK1/START,TETMT,NBLK,IPOW2,NCH,NPRINT,IPLOTA,IPLOTB,OFFSC
IA,LDELTA,SN,NPSP,M,SINT,ALR,NS,IX,IM,ISH,INCROSS,ICROSS(2,2),NCHP,YLABEL(2),WINDOW
1,FI,FO,IpYPSD
COMMON/BLK2/IHCN(14),CHSUM(14),NGFF(14),CHSUMW(14),SIGMA(14),RMS(14)
1,MEAN(14),SCALE/14),CHSUM(14),TRACK(14),ICHAN(14)
COMMON/BLK3/NPT,TPMAX,NPT02,NSPCT,DELF,N64,NPT028
1,INZEP,NREAD
COMMON/BLK5/PCTC,KMRTMS,DMAX(14),DMIN(14),OBIM(14),BINS(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 II=1,64
   IK=I-64+1
   IL=(II-1)*NCHP+J
   CALL WRTMS(9,SAVE64,N64,NREC)
   IF(NREAD.LT.64) GO TO 10
10 CONTINUE
C
COMPUTE BLOCK MEAN AND COUNTS FOR HISTOGRAM
C

APPENDIX H

C

BLKMEAN=0.
DO 22 I=1,NREAD
CHSUMSQ(J)=CHSUMSQ(J)+Z(I)**2
22 BLKMEAN=BLKMEAN+Z(I)
CHSUM1(J)=BLKMEAN+CHSUM1(J)
BLKMEAN=BLKMEAN/NREAD
IF(NBINS.EQ.0) GO TO 40
DMIN=DMIN(J) $ DEL=0BIN(JJ)
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,JJ)=BINS(IBIN,JJ)+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
GO TO (3C,31,32),IWINDOW
30 CALL HANNING(Z,NREAD)
GO TO 33
31 CALL HAMMING(Z,NREAD)
GO TO 33
32 CALL PARZEN(Z,NREAD)
33 CONTINUE
C INSERT ZEROS
C
IF(INZERO.EQ.0) GO TO 25
DO 35 I=1,NPT02
35 Z(I*NPT02)=0.
C COMPUTE FOURIER TRANSFORM AND STORE ON RANDOM ACCESS FILE
C
24 CALL FOURT(Z,NPT,1,1,0,SPECT)
IJ=(NB-1)*NCHP*JJ
CALL WRITMS(0,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(2.*PI*(I-1.)/NPT))
1 Z(I)=Z(I)*D
RETURN
END
APPENDIX H

SUBROUTINE HAMMING(Z,NPT)
COMPLEX Z(I)
CATA=PI/3.1415926535898
PION=PI/NPT $ NF2=NPT+2
DO 1 I=1,NPT
T=(I+1-NF2)*PION
D=.5+.46*COS(T)
1 Z(I)=Z(I)*D
RETURN
END

SUBROUTINE PARZEMZ(NPT)
COMPLEX Z(I)
TM = NPT/2 $ TM1=TM-1
TM02=TM/2.
DO 1 I=1,NPT
J=I-TM1
IJ=IABS(J)
IF( IJ-TM02) 2,2,3
2 U=1.-6.*J*J/(1.-IJ)
GO TO 1
3 U=2.*IJ**3
1 Z(I)=Z(I)*U
RETURN
END

OVERLAP(PATS,3,C)
PROGRAM AUTOSP
COMMON CMAIN(1)
COMMON/BLK1/STARTT,ITFMT,NBLK,IP0N2,NC4,PRINT,IPLOTA,IPLOTU,UFSC
1AL,DELTAT,SN,NRSKIP,LAP,NCRQSS,MCROSS(2,20),NCHP,YLABLE(2),WINDOW
1,F1,F2,ITYPESP,WCCN,NFSKIP,IFF,LAG1,LAG2
COMMON/BLK2/ICH(14),CHSUM(14),NCHF(14),CHSUMU(14),SIGMA(14),RMS(1
14),MEAN(14),SCALFAC(14),CHSUMU(14),TRACK(14),ICHAN(14)
COMMON/BLK3/NPT,TMAX,NPTO2,NSPCT,DELFL,NO4,NPTO128
1,INZER,NREAD
COMMON/BLK5/PCTC,NBINS,DMAX(14),DMIN(14),DBIN(14),BINS(100,14)
COMMON/BLK6/I4AVE64,IRI,IXPLOT,IUATA,I2,ISPECT
COMMON/BLK8/IALTSP(14),IAUTOCC(14),CRSPC(20),ICACOR(20),ITRA(20),
1ICOCO(20)
COMMON/BLK9/NFILTP,FREQF(50),WGFIL(50)
C
C COMPUTE AND PRINT SPECTRAL FILTER
C
IF(NFILTP.LE.0) GU TO 1
CALL SPLINE(FREQF,WGFIL,NFILTP,NSPCT,CMAIN( IRI),DELFL)
NREC=NBLK*NCHP+1
CALL WRITMS(B,CMAIN( IRI),NSPCT,NREC)
DO 2 I=1,NSPCT
2 CMAIN(IXPLOT+I-1)=(I-1)*DELFL
I=IXPLOT-1 $ I2=IRI-1
PRINT 900, (I,CMAIN(I+I),CMAIN(I+I),I=1,NSPCT)
900 FORMAT(/1 k SPECTRAL FILTER WEIGHTING FUNCTION///5x*I*3x*FREQUENCY*10x*WEIGHT*3(7X*I*3X*FREQUENCY*6X*WEIGHT*)/(16,2E13,5,16,2E13,5,15,
APPENDIX H

22E13.5,16,22E13.5)
1 CONTINUE
   CALL AUTO(CMAIN(I2),CMAIN(ISPECT),CMAIN(I1),CMAIN(I1XPLOT))
   CALL AUTO FUNCTION ROUTINE WITH COMPUTED BLOCK ADDRESSES
   IF(NINS.GT.0) CALL NORMAL
   RETURN
END
SUBROUTINE AUTO (I , SPECT, P1, XPLTT)
DIMENSION SPECT(l),XPLTT(l),P(l),I(l)
COMPLEX I, SPECT
COMMON/BLKI/STARTT,ITFMT,NBLK,IPOW2,NC4,NPRINT,IPLOTA,IPLOT,OFFSC
1AL,DELTAT,SN,NRSKIP,LAP,NCHOS,ICROSS(2,20),NCHP,YLABEL(2),WINOW
1,F1,F2,ITYPESP,WCUN,NFSKIP,IFF,LAG1,LAG2
COMMON/BLK2/ICH(14),CHSUM(14),NUFF(14),CHSUM SQ(14),SIGMA(14),RMS(1
14),MEAN(14),SCALE(14),CHSUM(14),TRACK(14),ICHAN(14),IFILTEK(14)
COMMON/BLK3/NPT,TMAX,NPTN2,NSPCT,NELF,N64,NPTU128
I,NZERO,NREAD
COMMON/BLK8/IAUTOSP(14),IAUTOCO(14),ICRSP(20),ICRCOR(20),ITKA(20),
1ICOH(20)
COMMON/BLK9/NFILT
COMMONT/BLK10/NCRD
DIMENSION PLABEL(6),PPLUT(27),BAND(27),ITEN(5)
REAL MEAN
COMPLEX Li
PRINT 917, (TRACK(J),NOFF(J),J=1,NCH)
912 FORMAT(/// NO. OF OFF-SCALE VALUES FOR EACH CHANNEL*/(1X,A10,T10))
C
C COMPUTE MEAN AND VARIANCE OF EACH CHANNEL
C
00 36 IKK=l,NCHP
  [K=ICHAN(IKK)]
  MEAN(IK)=CHSUM(IK)/(NOLK*NREAD)
  SECMOM=CHSUM SQ(IK)/(NBLK*NREAD)
  SIGMASQ=SECMOM-MEAN( IK)**2 )
  IF(SIGMASQ)38,37,37
38 PRINT 913, IK
913 FORMAT(* NEGATIVE SIGMASQ FOR CH.*I5)
SIGMA(IK)=0.
GO TO 36
37 SIGMA(IK)=SQRT(SIGMASQ)
36 CONTINUE
C
C START OF LEOP FCR COMPUTING AUTO SPECTRA AND CORRELATIONS
C
00 50 JJ=1,NCHP
  J=ICHAN(JJ)
DO 53 I=1,NSPCT
53 SPECT(I)=0.
C
C AVERAGE NBLK SPECTRA FOR ONE CHANNEL

98
APPENDIX H

DO 54 IBLK=1,NBLK
IJ=(IBLK-1)*NCHF+JJ
CALL READMS(8,Z,NPT,IJ)
GO TO (80,DO,81),ITYPESP
80 DO 91 I=1,NSPCT
91 SPECT(I)=SPECT(I)+(REAL(Z(I))**2+AIMAG(Z(I))**2)
GO TO 54
81 DO 82 I=1,NSPCT
Z=Z(I)
AMP=ABS(Z) & ARG=0. $ IF(AMP.NE.0.) ARG=ATAN2(AIMAG(Z),REAL(Z))
82 SPECT(I)=SPECT(I)+CMPLX(AMP,ARG)
54 CONTINUE

C COMPUTE CORRECTED AUTO SPECTRUM AND CALCULATED VARIANCE
C
SUM=0.
GO TO (83,83,84),ITYPESP
84 CON=2./(NBLK*NREAD)
DO 85 I=1,NSPCT
SPECT(I)=SPECT(I)*CON
85 CONTINUE
RMS(J)=0.
GO TO 86
83 CON=DELTA*DELTA/(6.283185308*CON*NBLK)
DO 59 I=1,NSPCT
SPECT(I)=SPECT(I)*CON
59 SUM=SUM+SPECT(I)
RMS(J)=SQRT(SUM*CON**2)
86 CONTINUE
C APPLY SPECTRAL FILTER
IF(NFILTER.EQ.0) CALL FILTER(J,1,NSPCT,NBLK+1)
DO 112 I=1,NSPCT
112 SPECT(I)=SPECT(I)*S(I)
C C PRINT AUTO SPECTRUM AND SET UP PLOTTING ARRAYS
C
58 PRINT 1001,J,TRACK(J),MEAN(J),SIGMA(J),RMS(J)
1001 FORMAT(///ACHANNEL=I3,3XA10,3X*MEAN=*$E12.5,3X*SIGMA=*
*E12.5,3X*PMS=*$E12.5)
1002 FORMAT(///AVERAGE OFTRANSFORMS///5X)I*3X*FREQUENCY*0X*POW
ER=3(6X*I*5X*FREQUENCY*6X*POWER*)
IF(ITYPESP.EQ.3) GO TO 87
DO 60 I=1,NSPCT
XPLT(I)=(1-I)*DELF
AMP=SPECT(I)*2*
SPECT(I)=RI(I)*AMP
60 CONTINUE
911 FORMAT(4(16,2E13.5))
APPENDIX H

CALL WRITMS(9,RI,NSPCT,JJ)
IF IAUTO=0) GO TO 100
GO TO (66,67,87), ITYPESP
66 ENCODE(60,901,PLABEL), DELF, TRACK(J)
901 FORMAT(*AUTO POWER SPECTRUM (BANDWIDTH=**E9.2,**) *9X,A1G)
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(*AMPLITUDE SPECTRUM AND PHASE**5X,A10,16X)
NPLABEL=27
PRINT 1009, (PLABEL(I),PLABEL(2)
1005 FORMAT(* AVERAGE OF 2 TRANSFORMS*//SX*I *JX*»)
REOU//CV*4X*AMPL
1009 FORMAT(* AVERAGE OF 2 TRANSFORMS*//SX*I *JX*»)
PRINT 1007, (I,XPLOT(I),RI(I)), I=1,NSPCT
89 CONTINUE
90 CONTINUE
CONTINUE
WHITE(7),PLABEL,NSPCT,(XPLOT(I),RI(I),I=1,NSPCT)
PRINT 1010, NRCRD7, PLABEL
1010 FORMAT(*//21H ** * * * * RECORD NO.,I5, * ON TAPE7 CONTAINS *6A10,1G
1H ** * * * *)
C
GO TO (7C, 70, 62, 63, 62, 631, IPLUTA
62 ILOG=3 $ GO TO 61
63 ILOG=2
61 CALL PLOTNB(PLABEL,TRACK(J),LO,XPLOT,RI,NSPCT,ILOG,F1,F2,PLABEL,
INPLABEL,IFF,1)
IF ITYPESP,LT.3) GO T) 70
ILOG=0
DO 90 I=1,NSPCT
90 RI(I)=AIMAG(SPECT(I))
ENCODE(50,904,PLABEL)
APPENDIX H

904 FORMAT(*PHASE ANGLE, DEGREES*30x)
NPLABEL=20
CALL PLOTNY(1, LABEL, TRACK(J), 10, XPLUT, RI, NSPCT, ILOG, F1, F2, PLABEL, 
1, NPLABEL, 0, 0)
GO TO 100
70 CONTINUE
IF(IYPESP.EQ.3) GO TO 100

101
114
113
110
1012
1014
1013
101
114
113
110
102
101
1011
1012
CONTINUE

GO TO (100,65,1CC,100,65,65),IPLUTA
CALL HANDS(DLF,NSPCT,SPECT,PLOT,BAND,NPP,IFER,ITYPESp)
IF(INPP.LE.0) GO TO 100
CALL PLOTNY(LABEL,TRACK(J),10,BAND,PLOT,NPP,PLABEL,NPLABEL,IFF)
CONTINUE
GO TO (72,71),ITYPESP
ENCODE(60,13,PLABEL) TRACK(J)
FORMAT(*1/3 OR AUTO POWER SPECTRAL DENSITY*6X,A10,13X)
GO TO 74
ENCODE(60,75,PLABEL) TRACK(J)
FORMAT(*1/3 OR AUTO POWER SPECTRUM*4X,A10,20X)
CONTINUE
100 IF(TAUTOCU(J).LE.6) GO TO 50

WRITE(7) PLABEL; L,NPP, (3AND( I ),PPLOT( 
PRINT 1010, NPCR7,PLABEL
IF(IAUTOCU(J).LE.6) GO TO 50

DO 101 I=2,NSPCT
Z(I)=SPECT(I)
101 Z(I)+NSPCT)=SPECT(NSPCT-I+2)
Z(I)=(NSPCT+1)=SPECT(I)
IF(IYPESP.GT.1) GO TO 113
DO 114 I=1,NPT
114 Z(I)=Z(I)/DEL
113 CONTINUE
CALL FOURT(Z,NPT,1,-1,0,SPECT)
CON=3.1415926535897/MAX
IF(INZERO) 1014,1014,1011
DO 110 I=1,NSPCT
110 SPECT(I)=Z(I)*CCN
GO TO 1012
1011 DO 1013 I=1,NSPCT
1013 SPECT(I)=Z(I)*CCN*NPT/(NPT-I+Z)
1012 CONTINUE

PRINT AUTOCORRELATION

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

RI(I) = SPECT(I)

102 XPLOT(I) = 1

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

1003 FORMAT(*AUTO CORRELATION*, *A10, 32X)

NPLT = 1007 + 1

WRITE(7, *) PLABEL, NSPCT, (XPLOT(I), RI(I), I = 1, NSPCT)

PRINT 1010, NPCRL7, PLABEL

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

C SET UP PLOT ARRAYS AND PLOT AUTO CORRELATION

L1 = LAG1 $ IF (L1.LT.0) L1 = 0 $ L2 = LAG2

IF (LAG2.GT.0) GO TO 109

L1 = -LAG2 $ L2 = -LAG1

109 I1 = L1 - 1 $ I2 = L2 + 1

103 ENCODE(50, 10, 'LABEL') TRACM(J)

920 FORMAT(*AUTO CORRELATION*, *A10, 22X)

PN = RI(NPLT)

PX = RI(NPLT/2) * 10. * PX, PN, Iطبق (2), 10. * 0, 0,

50 CONTINUE

RETURN

END

SUBROUTINE NORMAL

COMMON/BKL1/START, ITFMT, NBLK, IFUW2, NCH, NPRT, IPLOTA, IPLOTB, OFSC

1A1, DELTAT, SN, NRSKP, LAP, NCROSS, ICROSS(2, 20), NCHP, Iطبق (2), IWINDOW

1F1, F2, IgetEmail

COMMON/BKL2/ICH(14), CHSUM(14), NUFF(14), CHSUMSQ(14), SIGMA(14), RMS(1

1A1), MEAN(14), SCALFA(14), CHSUM1(14), TRACK(14), Iطبق (14)

COMMON/BKL3/NPT, TMAX, NPTU2, NSPCT, Iطبق (2), N64, NPT0128

1, INZERO, NREAF, NPTOT

COMMON/BKL5/PCTC, NBINS, CMAX(14), OMIN(14), OBIN(14), BINS(10, 14)

1, CHISQ

REAL MU, MEAN

DIMENSION YL(3), ICEN(5)
DATA IDENT/10H

NBINSM3=NBINS-3
DO 1=1,NCHP
J=ICHAN(I)
MU=MEAN(J)
ENCOD(30,902,YL) TRACK(J)
1 FORMAT(3X*HISTOGRAM FOR *A10,3X)
PN=PX=0.
CALL FANFORD(HINS(1,1),NBINS,1,1,LOG,IDENT,1H*,1.,PN,PX,YL,3,
1LOG,0.*,XAPP*Y,XLABEL)
SIG=SIGMA(J)
FACTOR=1./(2.5066*I2/SIG)
DEL=OBIN(I)
MAX=0.
SUM=0.
IF (DMIN(J).LT.-20.) GO TO 6
FN=PFUN(-20.,DMIN(J),MU,SIG)*FACTOR
GO TO 7
6 PN=0.
7 CONTINUE
PRINT 905, (BINS(K,I),K=1,NBINS)
905 FORMAT(* /BINS(*/10F13.0))
CHISO=0.
DO 2 K=1,NBINS
A=K*DEL+DMIN(J)
SUM= BINS(K,I)
P=SUM/NPTOT
PN= PFUN(A-DEL,A,MU,SIG)*FACTOR
IF (PN.>0.0) GO TO 2
CHISO=NPNTOT*(P-PN)**2/PN*CHISO
2 CONTINUE
ALPHA=1.-?CTC/1CC.
PRINT 906, NBINSM3,CHISO,ALPHA,CHISQ
906 FORMAT(* /12X33h* GOODNESS OF FIT TEST* 12X* 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)
RFAL MU
PFUN=0.
DX=(B-A)/50.
X=A-0X/2.
DO 1=1,50
X=X+DX
PFUN=PFUN+DX*EXP(-((X-MU)**2/(2. SIG))
1 CONTINUE
RETURN
END
SUBROUTINE PLOT((YLABEL,FRAMEL,NF,BAND,PFLP,NNP,PLABEL,NN,IFF)
APPENDIX H

```fortran
DIMENSION BAND(1), PLOT(1), YLABEL(2), FRAMEL(1), PLABEL(5), FFIU(5), 
1, IDEN(6), BCF(24)
DATA KCF/50., 100., 125., 150., 180., 200., 250., 315., 400., 500., 630., 
22000. /
C
C PLOT POWER FOR BAND CENTER FREQUENCIES 50-20K HZ
C
4001 DO 2001 I=1,NPP
2001 BAND(I)=I+4
  PMAX=PLOT(I)
  DO 2002 I=2,NPP
2002 PMAX=MAX1(PMAX,PMAX)
  NMAX=ITX(PMAX)
  IF(NMAX.LT.PMAX) NMAX=NMAX+1
  PMAX=NMAX
  PMIN=PMAX-5.
  I2=NPP
  DO 2004 I=1,I2
2004 IF(PPLCT(I).LT.PMIN) PPLGT(I) = PMIN
  CONTINUE
  PPLQT(NPP+1) = PMIN $ PPLQT(NPP+2) =.5
  BAND(NPP+1) = I $ BAND(NPP+2) =.5
  IF(IF1.EQ.0) RETURN
  FFID(I) = YLABEL(I) $ FFID(2) = YLABEL(2)
  NWPOS = (NF9)/10
  DO 16 I=1,NWPOS .
16 IF(IFF.EQ.0) RETURN
  ENCODE(54,9CS, IDEN) PLABEL
  935 FORMAT(*LOG *5A1°)
  CALL FANFLD(PPLCT,NPP,1,1,NPP,1DEN,1HE,1,1*,PMAX,PMIN,FFIU,NWPOS+2 
1,12G,1,1,BCF,1CHFREQUENCY )
  RETURN .
END
SUBROUTINE BANDS (UELF, NSPCT, SPECT, PLOT, BAND, NPP, IERR, IPOWER)
COMPLEX SPECT
DIMENSION FPLOT(1), IND(46,2), FNC(46), FNL(46)
```

DATA FNL/1.1220,1.4125, 1.7783,2.1587,2.3184,3.5481,4.4668,5.0234, 
17.0795,3.9129,11.220,14.125,17.783,22.367,26.184,35.481,44.008, 
52.8184,35.481,44.008 / 
DATA FNC/1.2549,1.5849,1.9955,2.5119,3.1623,3.9811,5.0119,6.3096, 
17.9466,10.1,14.5851,18.495,19.953,23.119,31.023,39.011,50.119,63.096, 
2.79.33,4.100,12.5,49,195,25,1.19,31.023,39.011,50.119,63.096 / 
DATA BCF/1.2549,1.5849,1.9955,2.5119,3.1623,3.9811,5.0119,6.3096, 
17.9466,10.1,14.5851,18.495,19.953,23.119,31.023,39.011,50.119,63.096, 
2.79.33,4.100,12.5,49,195,25,1.19,31.023,39.011,50.119,63.096, 
36.30.96,79.4,33,130.6,129.8,9,156.49,199.53,251.19,316.23,396.11,501.19, 
3630.96,79.4,33,130.6,129.8,9,156.49,199.53,251.19,316.23,396.11,501.19, 
45011,9,639,6,7543,3,10000,1,12569,1,15849,1,19953,2,25119,3,31623,1, 
539811 / 
DIMENSION PSD(46), PSUPHLZ(46), BAND(27)
DO 3 I=1,27
```
APPENDIX H

3 BAND(I)=1
IERR=0
DO 1 I=1,46
AND=FNL(I+1)-FNL(I)
11=I
IF(AND.GE.DEFL) GO TO 2
1 CONTINUE
IERR=1
RETURN
2 DO 62 I=1,46
PSDPHZ(I)=PSD(I)*C*
62 IND(I,1)=IND(I,2)=0
II=11
DO 63 I=2,NSPECT
F=(I-I)*DEFL
68 IF(II-40) 64,64,65
69 II=46
GO TO 66
64 IF(II-FNL(I)) 66,67,67
66 IF(IND(I,2).LT.0) IND(I,2)=1
IND(I,2)=IND(I,2)+1
GO TO 63
67 II=II+1
GO TO 68
63 CONTINUE
NPP=0
DO 76 I=1,46
IF(IND(I,2)) 76,76,70
70 ISTR=IND(I,1)
NPP=NPP+1
CALL PNDSIM(SPECT(ISTRT),IND(I,2),PSD(I))
PSD(I)=PSD(I)*DEFL
PSDPHZ(I)=PSD(I)/(FNL(I+1)-FNL(I))
76 CONTINUE
NPP=NPP-(17-I1)
IF(NPP.LT.0) NPP=0
IF(NPP.GT.27) NPP=27
PRINT 80, (I,FNL(I),PSD(I),PSDPHZ(I),I=I,46)
FORMAT(15,F12.1,E19.4,E19.4)
IF(NPP.EQ.0) GO TO 3004
GO TO (3001,3002,1) POWPLT
3001 DO 3003 I=1,NPP
PPLT(I)=-100.*IF(PSD(I+16).GT.GA) PPLT(I)=ALOG10(PSD(I+16))
3003 CONTINUE
GO TO 3004
3002 DO 3005 I=1,NPP
PPLT(I)=-100.*IF(PSD(I+16).GT.GA) PPLT(I)=ALOG10(PSD(I+16))
3005 CONTINUE
3004 RETURN
SUBROUTINE BNL(S, PSD)
COMPLEX S(1)
PSD=0.
IF(NUM.GT.1) GO TO 1
PSD=REAL(S(1))
RETURN
1 PSD=REAL(S(1)) + REAL(S(NUM))
IF(NUM.EQ.2) RETURN
NUMM1=NUM-1
DO 2 I=2,NUMM1
2 PSD=PSD+REAL(S(I))
RETURN
END
SUBROUTINE SPLINE (X,Y,KNT,KNTOUT,YOUT,DX)
DIMENSION C(4,4), A(4), IP IV(4), X(1), Y(1), YOUT(1)
XM = 2.*X(1) - X(2)
YM = 2.*Y(1) - Y(2)
XN = X(1)
YN = Y(1)
XO = X(2)
YO = Y(2)
XP = X(3)
YP = Y(3)
SLN = (YN-YM)/(XN-XM)
SLO = (YO-YN)/(XO-XN)
SLP = (YP-YP)/(XP-XO)
RKOUT = KNTOUT
IYNX=X(1)/DX
IF(IYNX*DX.LT.X(1)) IYNX=IYNX-1
VAR=IYNX*DX
DO 1 I=1,IYNX
1 YOUT(I) = Y(1)
IYNX=IYNX+1
LIM = KNT + 1
DO 7000 N=3,LIM
IF(SLN .NE. SLO .OR. SLO .NE. SLP) GO TO 3000
C LINEAR
A(4) = 0.
A(3) = 0.
A(2) = SLO
A(1) = YN - SLO*XN
GO TO 6000
3000 CONTINUE
A(1) = (SLO - SLN)/(XO - XM)
A(2) = (SLP - SLO)/(XP - XN)
IF(A(1) .NE. A(2)) GO TO 5000
C PARABOLIC
C(1,1) = 1.
C(2,1) = 1.
C(3,1) = 1.
APPENDIX H

\[
C(1,2) = XN \\
C(2,2) = XP \\
C(1,3) = XN \times XN \\
C(2,3) = XO \times XO \\
C(3,3) = XP \times XP \\
A(1) = YN \\
A(2) = YO \\
A(3) = YP \\
\text{CALL SIMEO}(C, 3, A, 1, DET, IPIV, 4, ISC) \\
A(4) = 0. \\
\text{GO TO 6000}
\]

5000 CONTINUE

C CUBIC

\[
C(1,1) = 1. \\
C(2,1) = 1. \\
C(3,1) = 0. \\
C(4,1) = 0. \\
C(1,2) = XN \\
C(2,2) = XO \\
C(3,2) = 1. \\
C(4,2) = 1. \\
C(1,3) = XN \times XN \\
C(2,3) = XO \times XO \\
C(3,3) = 2. \times XN \\
C(4,3) = 2. \times XO \\
C(1,4) = XN \times C(1,3) \\
C(2,4) = XO \times C(2,3) \\
C(3,4) = 3. \times C(1,3) \\
C(4,4) = 3. \times C(2,3) \\
A(1) = YN \\
A(2) = YO \\
A(3) = \text{TAN}(0.5 \times (\text{ATAN}(SLN) + \text{ATAN}(SLD))) \\
A(4) = \text{TAN}(0.5 \times (\text{ATAN}(SLD) + \text{ATAN}(SLP))) \\
\text{CALL SIMEO}(C, 4, A, 1, DET, IPIV, 4, ISC)
\]

6000 CONTINUE

\[
\text{SUM} = A(1) \\
\text{VARP} = 1. \\
\text{GO TO 6100} \\
\text{K} = 2 \times 4 \\
\text{VARP} = \text{VARP} \times \text{VAR} \\
\text{SUM} = \text{SUM} + A(K) \times \text{VARP}
\]

6100 CONTINUE

\[
\text{YOU1YNX} = \text{SUM} \\
\text{VAR} = \text{FLOAT}(\text{YOU1YNX}) \times \text{OX} \\
\text{IYNX} = \text{IYNX} + 1 \\
\text{IF (VAR} \leq \text{0.1)} \text{GO TO 6000} \\
\text{XN} = XN \\
\text{YN} = YN \\
\text{XO} = XO \\
\text{YN} = YO \\
\text{XO} = XP
\]
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(I))/XP
7000 CONTINUE
IF(IYNX.GT.KNTOLT) RETURN
DO 2 IYNX=KNTCUT
2 YOUTH = Y(KNT)
RETURN
END

PROGRAM CFOSSGP
COMMON CMFIN(1)
COMMON/BLK1/STARTT,ITFMT,NBLK,IPD2W,NCH,NPRINT,IPLOTA,IPLOTU,OFFSC
1AL,DELTAT,SN,NSKP,LAP,NCROSS,ICROSS(2,20),NCHP,YLABEL(2),IWINDOW
0,F1,F2,ITYPESP,SCUN,NSKIP,IFF,LAG1,LAG2
COMMON/BLK2/ICh(14),CHSUM(14),NJFF(14),CHSUMSW(14),SIGMA(14),RMS(1
4),MEAN(14),SCALFA(14),CHSUM14),TRACK(14),ICHAN(14)
COMMON/BLK3/NPT,TRAX,NPTO2,NSPCT,DELF,N64,NPT0128
1,INZK0,NREAD
COMMON/BLK6/ISAVE64,IRI,IXPLOT,IAUTA,IZ,ISPECT
COMMON/BLK8/IAUTOS(14),IAUTOC(14),ICKP(20),ICRCP(20),ITRA(20),
1ICOH(20)
COMMON/BLK9/NFILTP
COMMON/BLK10/NCRD7
CALL CROSS(CMAIN(I7),CMAIN(ISPECT),CMAIN(IRI),CMAIN(IXPLOT))
RETURN
END

SUBROUTINE CROSS(I,SPECT,F1,XPLOT)
COMPLEX SPECT(1),F1,XPLOT
DIMENSION R(1),XPLOT(1),FRAME(3),LABEL(6)
COMMON/BLK1/STARTT,ITFMT,NBLK,IPD2W,NCH,NPRINT,IPLOTA,IPLOTU,OFFSC
1AL,DELTAT,SN,NSKP,LAP,NCROSS,ICROSS(2,20),NCHP,YLABEL(2),IWINDOW
0,F1,F2,ITYPESP,SCUN,NSKIP,IFF,LAG1,LAG2
COMMON/BLK2/ICh(14),CHSUM(14),NJFF(14),CHSUMSW(14),SIGMA(14),RMS(1
4),MEAN(14),SCALFA(14),CHSUM14),TRACK(14),ICHAN(14),IFILTER(14)
COMMON/BLK3/NPT,TRAX,NPTO2,NSPCT,DELF,N64,NPT0128
1,INZK0
COMMON/BLK8/IAUTOS(14),IAUTOC(14),ICKP(20),ICRCP(20),ITRA(20),
1ICOH(20)
COMMON/BLK9/NFILTP
COMMON/BLK10/NCRD7
RETURN
APPENDIX H

DIMENSION IDEN(5), HAND(27), PPLLOT(27)
DATA RAD/57.29577795/

C START OF LCOP FCR COMPUTING CROSS FUNCTIONS
C
DO 70 ICR=1, NCRCSS
K1=ICROSSU(ICR) & K2=ICROSSS(ICR)
IF (IAUTOS(K1), EQ., 0) & IAUTOS(K2), EQ., 0) GO TO 75
DO 77 I=1, NCHP
IF(K1, EQ., ICAM(I)) J1=1
IF(K2, EQ., ICAM(I)) J2=1
77 CONTINUE
DO 76 I=1, NSPCT
76 SPECT(I)=0.
C AVERAGE CROSS SPECTRA FOR ONE PAIR
C
DO 71 IBLK=1, NBLK
IJ=J1+(IBLK-1)*NCHP
CALL READMS(18, Z, NPT, IJ)
IJ=J2+(IBLK-1)*NCHP
CALL READMS(8, Z(NSPCT+1), NPT, IJ)
DO 71 I=1, NSPCT
71 SPECT(I)=SPECT(I)+Z(I)*CONJG(Z(NSPCT+1))
IF(NFILTER(J1), EQ., 0).AND. IFILTER(J2), EQ., 0) GO TO 111
CALL READMS(8, R1, 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 COMPUTE CORRECTED CROSS SPECTRUM
C
CON=DELTAT*DELTAT/(6.283185308*CON*NBLK)
DO 115 I=1, NSPCT
115 SPECT(I)=SPECT(I)*CON
ENCOD(23, 900, FRAME) TRACK(K1), TRACK(K2)
900 FORMAT(A10, 3H X , A10)
IF(ICRSP(1CR), LE., 0) GO TO 105
GO TO (60, 61), ITYPESP
60 CON=CON*DELF
ENCOD(50, 901, PLABEL) DELF
901 FORMAT(CROSS POWER SPECTRUM (BANDWIDTH = G9.2, *), 6X)
NPLABEL=44
PRINT 898, (PLABEL(I), I=1, 5), FRAME
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, 921, PLABEL)
APPENDIX H

902 FORMAT(*CROSS POWER SPECTRAL DENSITY*22X)
   NPLABEL=28
   PRINT 899, (PLABEL(I),I=1,3),FRAMEL
899 FORMAT(*I*3A1O,5X,3A1O)
   ENCODE(60,922,1DEN) TRACK(K1),TRACK(K2)
922 FORMAT(*CROSS POWER SPECTRAL DENSITY *2A1O,1X,ALG)
62 IF(NPRINTGT,0) PRINT 897
897 FORMAT(*//4X*I*4X*FREQUENCY*9X*REAL*16X*IMAG*7X*AMPLITUDE*7X*_PHASE*1)

C
C SET UP PLOT ARRAYS AND PRINT CROSS SPECTRUM
C
DO 72 I=1,NSPCT.
   F=(I-1)/MAX
   XPLXT(I)=F
   AMP=CAH(I(SPECT(I)))
   IF(AMP) 79,80,79
80 ARG=0. $ GO TO 78
79 ARG=A:TAN2(A:IMAG(I(SPECT(I))),REAL(I-SPECT(I)))R/A
78 CONTINUE
   IF(I-NPRINT)73,73,200
73 PRINT 915, I,F,SPECT(I),AMP,ARG
915 FORMAT(I5,5E14.5)
200 IF(IPLOTC.EQ.0) GO TO 201
   GO TO (201,201,202,202,202,202),IPLOTC
201 RI(I)=SPECT(I)
   RI(I-NSPCT+2)=AIMAG(I(SPECT(I)))
   GO TO 72
202 RI(I)=AMP
   RI(I+NSPCT+2)=ARG.
72 CONTINUE
   IF(IPLOTC.EQ.2) 301,301,302
301 IDENTH=10H (REAL)
   IDENT7=10H (IMAG)
   GOTO 303
302 IDENT6=10H AMPLITUDE
   IDENT6=10H PHASE
   NRCRD7=NRCRD7+1
   NSPCTP2=NSPCT+2
   PRINT 1000, NRCRD7,IDENT,IDENT6
1000 FORMAT(*1H**1H *** RECORD NO.,I5, * ON TAPE7 CONTAINS *6A1O,10*
1H **1H ****)
   WRITE(7) IDENT,IDENT6,NSPCTP2(XPLXT(I),RI(I),I=1,NSPCT)
   NRCRD7=NRCRD7+1
   PRINT 1000, NRCRD7,IDENT,IDENT7
   WRITE(7) IDENT,IDENT7,NSPCTP2(XPLXT(I),RI(I+NSPCTP2),I=1,NSPCT)

C
C PLOT +ANFOLD PLOTS
C
NW=(NPLABEL-11/10+2
NPLABEL=NW*10

110
APPENDIX H

1. IF(IPLTOC.EQ.,0) GO TO 105
2. GO TO (107,108,109,109,109) IPLTOC
3. GO TO 116
4. IPLTOC=2
5. ENDDIM(10,903,PLABEL(NW))
6. FORMAT(* (REAL*))
7. CALL PLOTB(VLABEL,FRAMEL,23,XPLOT,RI,NSPCT,ILUG,F1,F2,PLABEL,NPLA
8. LABEL,IFF,1)
9. ENDDIM(10,904,PLABEL(NW))
10. FORMAT(* (REAL*))
11. CALL PLOTB(VLABEL,FRAMEL,23,XPLOT,RI,NSPCT,ILUG,F1,F2,PLABEL,NPLA
12. LABEL,IFF,1)
13. GO TO 105
14. IPLTOC=3
15. ENDDIM(10,905,PLABEL(NW))
16. FORMAT(* MAGNITUDE*)
17. CALL PLOTB(VLABEL,FRAMEL,23,XPLOT,RI,NSPCT,ILUG,F1,F2,PLABEL,NPLA
18. LABEL,IFF,1)
19. ENDDIM(10,906,PLABEL(NW))
20. FORMAT(* PHASE*)
21. IF(1LOG.GT.1) 1LOG=1LOG-2
22. CALL PLOTB(VLABEL,FRAMEL,23,XPLOT,RI,NSPCT,ILUG,F1,F2,PLABEL,NPLA
23. LABEL,IFF,1)
24. CONTINUE
25. IF(TCRCCF.T.CRN) .LE.0 GO TO 411
C
26. COMPUTE CROSSCORRELATION
C
27. DO 74 I=2,NSPCT
28. Z(I)=SPECT(I)
29. 74 Z(NSPCT+I)=CONJG(SPECT(NSPCT-I+2))
30. Z(I)=Z(NSPCT+I)=SPECT(I)
31. IF(ITYPESP.GT.1) GO TO 113
32. DO 114 I=1,NPT
33. 114 Z(I)=Z(I)/DELF
34. CONTINUE
35. CALL FOURP(7,NPT,1,-1,1,SPECT)
36. PRINT 916, TRACK(K1),TRACK(K2)
37. CON=6.23165328/TMAX
38. DE 110 I=1,NPT
39. SPECT(I)=Z(I)*CCN
C
40. SET UP PLOT ARRAYS, PRINT AND PLOT CORRELATION
C
41. IF(INZERO) 50G,50G,501
42. DO 502 I=1,NSPCT
43. IM1=I-1
44. IPN=I+NSPCT
45. R(I,IPN)=SPECT(I)
46. XPLOT(IPN)=IM1
47. IMN=I-NSPCT-1

111
APPENDIX H

R1(I) = SPECT(IPN)

502 X PLO T(I) = I M N
GO TO 503

501 DO 90 I = 1, NSPCT
IM 1 = I - 1
IPN = I * NSPCT
I2M1 = 2 * I M 1
R1(IPN) = NPT * SPECT(I) / (NPT - 12M1)
X PLOT(IPN) = I M 1
IMN = I - NSPCT - 1
I2MN = 2 * I M N $ IF (I.EQ.1) I2MN = I2MN + 1
R1(I) = NPT * SPECT (IPN) / (NPT + I2MN)
90 XPLO T(I) = I MN

503 CONTINUE

PRINT 917, ( X PLO T(I), R1(I), I = 1, NPT)
917 FORM A T(2X, 8(F5, 9, E 11.3))

916 FORM AT//S8CROSSCO RE LATION, *A10,* X *A10//8(7X*15X*RXY*))
ENC OD(GO, 523, PL ABEL) TRACK(K1), TRACK(K2)

923 FORM AT/*CROSSCORRELATION *A10,1X,A10,2CX)
WRITE(7) PLABEL, NPT, (XPLO T(I), R1(I), I = 1, NPT)
NRCD07 = NRCD07 + 1
PRINT 1000, NRCD07, PLABEL
IF (LAG1.EQ.0) GO TO 411
I1 = LAG1 + NSPCT + 1 $ 12 = LAG2 + NSPCT + 1
206 IF (I1 .GE. I2) GO TO 411
NPLT = I2 - I1 + 1
CALL ASCALE(RI(I1), 100, NPLT, 1, 10.)
IF (I1 .EQ. 0) GO TO 411
IF (NPLT .LT. 256) 101, 101, 102
101 K = 1 GO TO 103
102 K = NPLT / 256
103 ENCODE(53, 920, IDEN) FRAMEL

920 FORM AT/*CROSS CORRELATION*3X, 3A10)
PN = RI(NPLT + 11)
PX = RI(NPLT + 11) * I0. * PN
CALL HAFPOLUT(RI (I1), NPLT, K, 1, NPLT, IDEN, 1H, 1, PX, PN, YLABEL, Z, 1ZC, 0
1, 0, 1H )
GO TO 411
75 PRINT 918, TRACK(K1), TRACK(K2)
918 FORM AT//CROSS SPECTRA FOR *A10,* X *A10,* CANNOT BE COMPUTED*
GO TO 70
411 IF (ICOM*ICR) .LE. 0 GO TO 401
C
C COMPUTE COHERENCE
C
CALL READMS(9, R1, NSPCT, J1)
CALL READMS(9, R1(NSPCT + 1), J2)
CALL READMS(9, SPECT, NPT, NCHP + 1)
CON = DELTAT * 2/(6.283185308*WCON*NOLK)
DO 402 I = 1, NSPCT
XPLCT(I) = (I - 1) / 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)/SQR(T(DENOM)))*CON
402 CONTINUE
PRINT 403, FRAMEL,(I,XPLOT(I),RI(I),I=1,NPRINT)
C C PRINT AND PLOT COHERENCE
C
403 FORMAT('/'$COHERENCE *3AI0/7X*!A3*FREQUENCY*4X*COHERENCE*3(6X*1E
13*X*FREQUENCY*4X*COHERENCE*)/(16,2E13,5,16,2E13,5,16,2E13,5,16,2E13
2,5))
ENCOD(60,924,PLABEL) TRACK(K1),TRACK(K2)
924 FORMAT('/'$COHERENCE *4AI0,7X,A10,23X)
NCRD7=NCRD7+1
WRITE(7) PLABEL,NSPCT,(XPLOT(I),RI(I),I=1,NSPCT)
PRINT 1000, NCRD7,PLABEL
ENCOD(50,4C4,PLABEL)
474 FORMAT('/'$COHERENCE*4X)
CALL PLOTNR(PLABEL,FRAMEL,23,XPLOT,RI,NSPCT,0,F1,F2,PLABEL,10,IFF,
11)
401 IF(TRA(I).EQ.0) GO TO 70
C C COMPUTE TRANSFER FUNCTION
C
CALL REALMS(SPECT,NPT,NCHP+1)
IF(TRA(I).LT.0) GO TO 405
CALL REALMS(RI,NSPCT,J1)
ENCOD(60,406,PLABEL) TRACK(K1),TRACK(K2)
GO TO 401
405 CALL REALMS(RI,NSPCT,J2)
ENCOD(60,406,PLABEL) TRACK(K1),TRACK(K2)
408 FORMAT('/'$TRANSFER FUNCTION, TRACK FOR *A10,3H X ,A10,8X)
406 FORMAT('/'$TRANSFER FUNCTION, TRACK FOR *A10,3H X ,A10,8X)
407 CON=DELTAT**2/(6.26318*360*#CON*#BLK)
DO 409 I=1,NSPCT
XPLOT(I)=(I-1)/TMAX
DENOM=50*I(I)
IF(DENOM)414,414,415
414 RI(I)=0. $ GO TO 409
415 RI(I)=CABS(SPECT(I)/DENOM)*CON
409 CONTINUE
C C PRINT AND PLOT TRANSFER FUNCTION
C
PRINT 410, PLABEL,(I,XPLOT(I),RI(I),I=1,NPRINT)
410 FORMAT('/'$*A10, /7X*15*FREQUENCY,*7X*TRA*3(9X*15*FREQUENCY
*7X*TRA*)/(16,2E13,5,16,2E13,5,16,2E13,5,16,2E13,5))
NCRD7=NCRD7+1
WRITE(7) PLABEL,NSPCT,(XPLOT(I),RI(I),I=1,NSPCT)
PRINT 1000, NCRD7,PLABEL
CALL PLOTNR(PLABEL,FRAMEL,23,XPLOT,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