

AD-A082 082

JAYCOR ALEXANDRIA VA
USER'S MANUAL FOR COMPUTER PROGRAMS TO PERFORM OCEANOGRAPHIC VE--ETC(U)
JAN 80

F/G 9/2

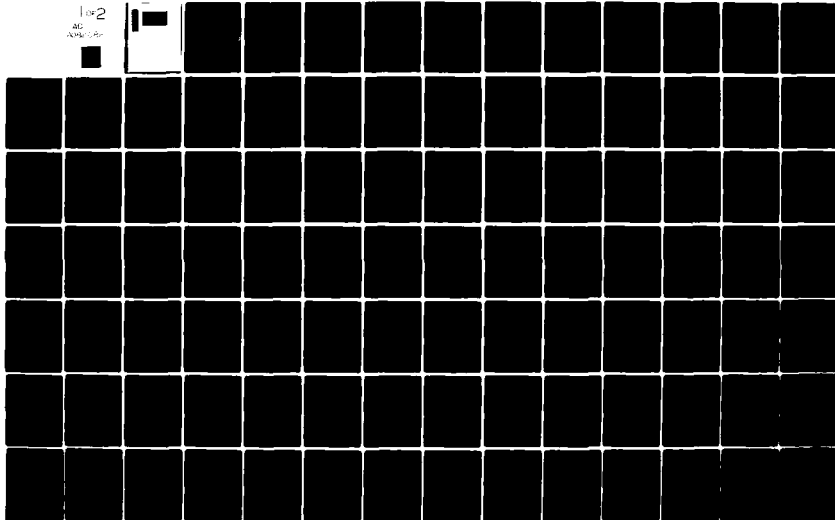
N00014-78-C-0879

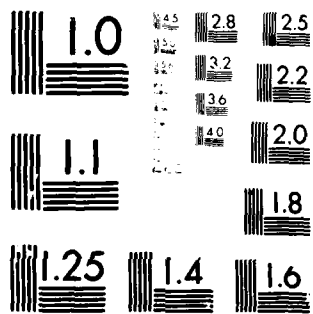
NL

UNCLASSIFIED

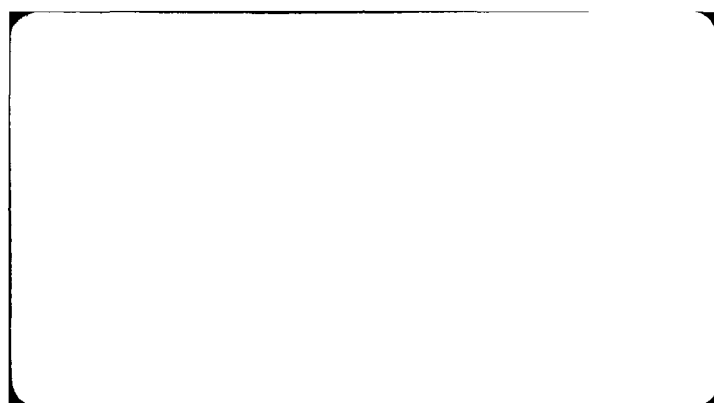
1-2

NO
AD-A082 082





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A



1211

12

6

USER'S MANUAL FOR COMPUTER PROGRAMS
TO PERFORM
OCEANOGRAPHIC VECTOR TIME SERIES
DATA ANALYSIS AND RELATED GRAPHICS

Submitted to:

The Naval Ocean Research and Development Activity,
Ocean Environmental Measurements Program
(Code 500)

In partial fulfillment of requirements under
Contract N00014-78-C-0879

15

DTIC
ELECTE
MAR 12 1980
A

11

January, 1980

12

175

Accession For	
NTIS GRA&I	☒
DDC TAB	☐
Unannounced	☐
Justification	☐
By <i>Butler</i>	
Distribution/	
Availability Codes	
Dist	Avail and/or special
<i>A</i>	

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

JAYCOR, P.O. BOX 5, CARY, NC 27511

2
343422

Preface

This "User's Manual for Computer Programs to Perform Oceanographic Vector Time Series Analysis" was prepared for the Environmental Measurements Program (Code 500) of NORDA in partial fulfillment of requirements under an Office of Naval Research Contract number N00014-78-C-0879. The software is written in Fortran IV specifically for use on the UNIVAC 1108 computer located at NAVOCEANO in NSTL Station, Mississippi. It is capable of accommodating any vector time series data, but the products are specifically directed at meteorologic and physical oceanographic data including current meter and wind data. The guidelines on this effort were that the vector time series software had to be compatible with existing FESTSA software at NAVOCEANO and that the plotting routines were to utilize the sophisticated DISSPLA software currently available at NAVOCEANO; both of these were adhered to. Both the rotary spectral and rotary cross spectral programs are designed to read directly from the FEB files of the FESTSA system. Another critical requirement was to extend the capability of the user to process records of lengths other than 2^n ; this was accomplished by incorporating the mixed radix fast Fourier transform program of Singleton (1969) which is based on prime factors. This feature enables users to select records of length 2^n , if desired, but it is not a limitation as in many other vector time series analysis programs. Subroutines are available for pre-whitening and post-coloring of data, if desired, and there are a number of data "windows" available along with three different types of averaging: block, convolution, and a combination of the two. In all there are a number of valuable user options included in the data analysis and graphics software which make it quite flexible and powerful.

The user's manual is organized into three parts: an introduction including a brief explanation of the theory behind vector time series analysis; a description of all the data analysis and graphics programs; and a series of five appendices with program and data set listings. Two cautions must be emphasized in using this manual:

(1) This is not intended to be, nor is it a complete treatise on the subject of time series data analysis. It is recommended that some preliminary reading be done in one or more of the references cited herein.

(2) Read this manual very carefully before attempting to use the software programs. The very nature of the analysis dictates careful reading, and in order to fully realize the benefits of available user options one must understand how they are called.

Acknowledgements

JAYCOR would like to extend its thanks to Dr. Mike Stanley for all his assistance in this effort. Special thanks are extended to Dr. Kim Saunders for his invaluable aid in working with the UNIVAC system and Mr. Mark Bergin for his perseverance during the testing phase of the draft user's manual.

TABLE OF CONTENTS

1.0	INTRODUCTION	1
2.0	COMPUTATIONS	3
2.1	The Velocity Hodograph	3
2.2	Rotary Spectra	6
2.3	Basic Scaling.	12
2.4	Windowing.	14
2.5	Pre-whitening/Post-coloring.	14
2.6	Averaging.	21
2.7	Confidence Intervals and Significance Levels . . .	21
2.8	Sample Products.	22
3.0	DESCRIPTION OF DATA ANALYSIS PROGRAMS.	26
3.1	GET.	26
3.2	FOURCO	28
3.3	RSPEC and RCSPEC	29
4.0	COMPREHENSIVE RUNSTREAM.	35
5.0	DESCRIPTION OF PLOTTING PROGRAMS	37

APPENDICIES

A.	DATA ANALYSIS PROGRAMS	A-1 to A-74
B.	'JAY.RUNALL'	B-1 to B-2
C.	'MAP' DATA SET LISTINGS.	C-1
D.	'JAY.USER'	D-1
E.	PLOTTING PROGRAM	E-1 to E-38

REFERENCES

1.0 INTRODUCTION

1.0 INTRODUCTION

The application of spectral analysis to oceanographic time series has become routine over the past decade. The purposes have generally been to assess the time scales and the distribution of variance (energy) between differing geophysical processes, to determine the spatial structure of these processes, and to determine interrelationships between processes.

Spectral analysis tools primarily evolved through the field of communications engineering, e.g., Blackman and Tukey (1958) where initial oceanographic applications were for the analysis of scalar time series. Some examples regarding sea level fluctuations include Groves and Hannan (1968) and Wunsch (1972). The increased availability and complexity of oceanographic velocity time series in the mid-1960's led to new problems. Applications of scalar techniques to velocity vector components generally yielded results that depended upon coordinated system orientation. This necessitated the development of invariant statistics and rotary spectral analysis. Pioneering work includes the papers by Fofonoff (1969), Gonella (1972), and Mooers (1973). Applications to internal inertia-gravity waves includes Muller and Siedler (1976) and Muller, Olbers and Willebrand (1978). Recent applications to atmospheric and oceanic planetary wave analyses respectively includes Hayashi (1979) and Weisberg, Horigan, and Colin (1979).

Four separate programs are utilized to generate spectral quantities for plotting and analysis: GET, FOURCO, RSPEC, and RCSPEC. GET removes the desired time series from a Febfile and outputs it to a new dataset for use by FOURCO. FOURCO then computes the time series mean, variance, and Fourier coefficients. A variety of spectral windows and pre-whitening are available as options. The Fourier coefficients are then fed into RSPEC and RCSPEC to perform rotary spectral analysis for a single vector time series and rotary cross spectral analysis for a pair of vector time series respectively. Both spectral programs output data-

sets for plotting purposes. Printouts may also be obtained. The programs are set up so that one runstream can be used with finished, plotted results as an end product.

This user's manual consists of several parts. Section 2 develops the velocity hodograph model and discusses the various computations. The data analysis programs and all options are presented in Section 3. Section 4 describes the plotting routines. Finally, the source listings are given as appendices.

2.0 COMPUTATIONS

2.0 COMPUTATIONS

2.1 The Velocity Hodograph

Consider a time series consisting of a single Fourier constituent at a given frequency f . The trigonometric representation of the east and north velocity components (u , v) in a cartesian coordinate system (x , y) are:

$$\begin{aligned}u &= a_1 \cos 2\pi ft + b_1 \sin 2\pi ft \\v &= a_2 \cos 2\pi ft + b_2 \sin 2\pi ft,\end{aligned}$$

where the a 's and b 's are the Fourier coefficients. Owing to the orthogonality of the trigonometric functions any piecewise continuous time series can be viewed as a linear superposition of Fourier constituents at different frequencies. The Fourier coefficients are then obtained via finite Fourier transformation.

In complex form, u and v may be rewritten as:

$$\begin{aligned}u &= 1/2 \left[(a_1 - ib_1) e^{i2\pi ft} + (a_1 + ib_1) e^{-i2\pi ft} \right] \\v &= 1/2 \left[(a_2 - ib_2) e^{i2\pi ft} + (a_2 + ib_2) e^{-i2\pi ft} \right]\end{aligned}$$

where $i = (-1)^{1/2}$. Defining now, the complex velocity vector in the argand plane,

$$w = u + iv,$$

and upon substitution and rearrangement:

$$\begin{aligned}w &= \left[\frac{a_1 + b_2}{2} \right] e^{i2\pi ft} + i \left[\frac{a_2 - b_1}{2} \right] e^{i2\pi ft} \\&\quad + \left[\frac{a_1 - b_2}{2} \right] e^{-i2\pi ft} + i \left[\frac{a_2 + b_1}{2} \right] e^{-i2\pi ft}\end{aligned}$$

Further definition of the quantities:

$$A = 1/2 \left[(a_1+b_2)^2 + (a_2-b_1)^2 \right]^{1/2}$$

$$C = 1/2 \left[(a_1-b_2)^2 + (a_2+b_1)^2 \right]^{1/2}$$

$$\eta = \tan^{-1} \left(\frac{a_2-b_1}{a_1+b_2} \right)$$

$$-\tau = \tan^{-1} \left(\frac{a_2+b_1}{a_1-b_2} \right)$$

allows w to be expressed as:

$$w = Ae^{i\eta} e^{i2\pi ft} + Ce^{-i\tau} e^{-i2\pi ft}$$

In this form, the quantities defined have a simple conceptual basis. The + and - exponentials correspond to anticlockwise and clockwise rotating unit vectors respectively in the argand plane. Therefore, A and C are the amplitudes of the anticlockwise and clockwise components, and η and τ are their corresponding temporal phase angles.

As the vector w sweeps through a cycle in the argand plane, its tip traces out a hodograph which has a simple geometrical interpretation. Factoring out $e^{i(\eta-\tau)/2}$, w may be rewritten as:

$$w = e^{i(\eta-\tau)/2} \left[Ae^{i\{2\pi ft + 1/2(\eta+\tau)\}} + Ce^{-i\{2\pi ft + 1/2(\eta+\tau)\}} \right]$$

$$= e^{i(\eta-\tau)/2} w'$$

Hence w is expressed as a second vector function w' rotated anticlockwise through an angle $(\eta-\tau)/2$. In trigonometric form:

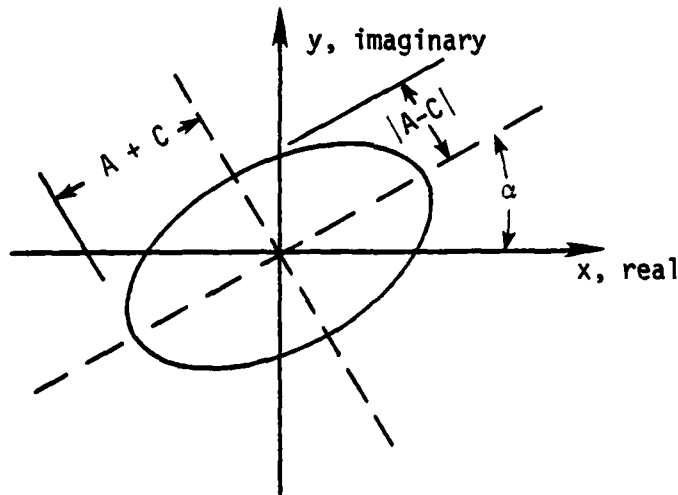
$$w' = (A+C) \cos \{ 2\pi f t + 1/2(\eta+\tau) \} + i(A-C) \sin \{ 2\pi f t + 1/2(\eta+\tau) \}$$

$$= w_R' + i w_I'$$

Dividing w_R' by $A+C$ and w_I' by $A-C$, squaring, and adding, yields an equation for an ellipse in the argand plane:

$$\frac{w_R'^2}{(A+C)^2} + \frac{w_I'^2}{(A-C)^2} = 1.$$

The semimajor axis of the ellipse is $A+C$, the semiminor axis is $|A-C|$, and the orientation of the principal axis of variance (semimajor axis) is $\alpha = (\eta-\tau)/2$. This is shown schematically below.



The ellipse is polarized anticlockwise if $A > C$, clockwise if $A < C$, and it is rectilinear if $A = C$. If either A or C equals zero, then the ellipse is a circle with the appropriate polarization.

The principal axis of variance defines a new coordinate system in which w_R' and w_I' are in quadrature. Consequently the average of their products equals zero ($\langle w_R' w_I' \rangle = 0$) and they are orthogonal. The new coordinate system therefore corresponds to the normal

coordinates of the time series at frequency f . The temporal phase angle of the velocity vector relative to the normal coordinates is given by:

$$\theta = \tan^{-1} \frac{w_I'(0)}{w_R'(0)} = \tan^{-1} \left[\frac{A-C}{A+C} \tan 1/2 (\eta + \tau) \right]$$

2.2 Rotary Spectra

Rotary spectra and hodograph parameters for single vector time series and pairs of vector time series are computed by RSPEC and RCSPEC respectively. The computed quantities are formulated in this section. The next section then discusses scaling, averaging, and other computational aspects.

Following Muller and Siedler (1976), the anticlockwise and clockwise horizontal velocity component transforms (U_+ and U_- respectively) are defined for positive frequency as

$$U_+ = U + iV$$

$$U_- = U - iV$$

where U , V are proportional to the complex Fourier coefficients at a given frequency f computed via finite Fourier transformation. Since f is chosen as positive (as opposed to positive or negative in the previous section) we consider Fourier coefficients of the form $a - ib$ as opposed to $a + ib$. Rotary cross-spectra between two vector time series at locations A and B are given by

$$S_{\nu\mu}^{AB} = U_{\nu}^{A*} U_{\mu}^B$$

where ν and μ are either plus or minus and the asterisk denotes the complex conjugate. Thus:

$$S_{++}^{AB} = U_+^{A*} U_+^B$$

$$S_{--}^{AB} = U_-^{A*} U_-^B$$

$$S_{+-}^{AB} = U_+^{A*} U_-^B$$

$$S_{-+}^{AB} = U_-^{A*} U_+^B$$

In general, the rotary-cross spectra are complex valued functions with an amplitude and phase. They satisfy the relationship,

$$S_{\nu\mu}^{AB} = (S_{\mu\nu}^{BA})^*;$$

therefore the above four cross-spectra are sufficient to describe all of the linear relationships between the vector time series at points A and B. Normalizing the amplitude of the rotary cross-spectra between points A and B by the respective rotary spectra at these points results in the rotary coherencies. Thus:

$$\gamma_{\nu\mu}^{AB2} = \frac{S_{\nu\mu}^{AB} [S_{\nu\mu}^{AB}]^*}{S_{\nu\nu}^{AA} S_{\mu\mu}^{BB}}$$

where $\gamma_{\nu\mu}^{AB2}$, the rotary coherence squared, is a real valued function ranging between zero and unity. The phase difference between rotary components at location A and B follow from the real and imaginary parts of the rotary cross spectra or,

$$\phi_{\nu\mu}^{AB} = \tan^{-1} \left[\frac{\text{Im} \{S_{\nu\mu}^{AB}\}}{\text{Re} \{S_{\nu\mu}^{AB}\}} \right],$$

where the imaginary part (Im) is called the rotary quadrature spectrum and the real part (Re) is called the rotary co-spectrum. The sign convention employed is

$$S_{\nu\mu}^{AB} = \text{Re} \{S_{\nu\mu}^{AB}\} + i \text{Im} \{S_{\nu\mu}^{AB}\} = |S_{\nu\mu}^{AB}| e^{i\phi_{\nu\mu}^{AB}},$$

thus a positive phase implies that oscillations at B lead those at A and conversely.

The programs employ complex arithmetic therefore the equations given above are sufficient for the calculations. With some arithmetic, however, the rotary cross-spectra can be computed in terms of the conventional scalar velocity component spectra. This will now be shown for the case of a single vector time series for which $A = B$. Only 3 rotary cross-spectra need be computed since $S_{-+} = S_{+-}^*$. Thus

$$\begin{aligned} S_{++} &= U_+^* U_+ = (U^* - iV^*) (U + iV) \\ &= U^*U + V^*V + i (U^*V - V^*U) \\ &= S_{uu} + S_{vv} + i (S_{uv} - S_{vu}); \end{aligned}$$

but, $S_{vu} = S_{uv}^*$, so

$$\begin{aligned} S_{++} &= S_{uu} + S_{vv} + i (2 i \text{Im} \{S_{uv}\}) \\ &= S_{uu} + S_{vv} - 2 \text{Im} \{S_{uv}\}. \end{aligned}$$

Similarly:

$$S_{--} = S_{uu} + S_{vv} + 2 \text{Im} \{S_{uv}\}$$

and

$$S_{-+} = S_{uu} - S_{vv} + i 2 \text{Re} \{S_{uv}\}$$

where S_{uu} , S_{vv} , and S_{uv} are the scalar autospectra for the u and v components and the cross-spectrum between u and v . S_{++} and S_{--} are the anticlockwise and clockwise rotary spectra while S_{+-} is the cross-spectrum between the clockwise and the anticlockwise components.

Expanding S_{++} and S_{--} further in terms of the Fourier coefficients yields:

$$\begin{aligned}
 S_{++} &= 1/4 (a_1 + ib_1) (a_1 - ib_1) + 1/4 (a_2 + ib_2) (a_2 - ib_2) + \\
 &\quad \frac{i}{4} (a_1 + ib_1) (a_2 - ib_2) - (a_2 + ib_2) (a_1 - ib_1) \\
 &= 1/4 (a_1^2 + b_1^2 + a_2^2 + b_2^2) + \\
 &\quad \frac{i}{4} [a_1 a_2 + b_1 b_2 + i (a_2 b_1 - a_1 b_2) - a_1 a_2 - b_1 b_2 - i (a_1 b_2 - a_2 b_1)] \\
 &= 1/4 [a_1^2 + b_1^2 + a_2^2 + b_2^2 + 2a_1 b_2 - 2a_2 b_1] \\
 &= 1/4 [(a_1 + b_2)^2 + (a_2 - b_1)^2] \\
 &= A^2
 \end{aligned}$$

where A is the amplitude of the anticlockwise component computed in section II for the velocity hodograph model. Similarly $S_{--} = C^2$.

The coherence squared between the clockwise and anticlockwise components and their phase difference define the orientation of the velocity hodograph and the stability of the ellipse (Gonella, 1972). The coherence squared, or stability, is given by:

$$\gamma_{-+}^2 = \frac{S_{-+} S_{-+}^*}{S_{++} S_{--}} = \gamma_{+-}^2$$

$$= \frac{(S_{uu}-S_{vv})^2 + 4 \operatorname{Re}^2 \{S_{uv}\}}{(S_{uu}+S_{vv})^2 - 4 \operatorname{Im}^2 \{S_{uv}\}}.$$

The phase difference is:

$$\phi_{-+} = \tan^{-1} \left[\frac{2 \operatorname{Re} \{S_{uv}\}}{S_{uu} - S_{vv}} \right]$$

and the orientation $\alpha = 1/2 \phi_{-+}$ (note that α is $\pi/4$ when $S_{uu} = S_{vv}$ whereas $\phi_{-+} = \frac{\pi}{2}$). Thus

$$\tan 2\alpha = \frac{2 \operatorname{Re} \{S_{uv}\}}{S_{uu} - S_{vv}}$$

or in terms of the Fourier coefficients

$$\tan 2\alpha = \frac{(2)(1/4) (a_1 a_2 + b_1 b_2)}{1/4 (a_1^2 + b_1^2 - a_2^2 - b_2^2)}$$

$$\tan 2\alpha = \frac{2a_1 a_2 + 2b_1 b_2}{a_1^2 + b_1^2 - a_2^2 - b_2^2}$$

This agrees with the results obtained from the velocity hodograph model as will now be shown. Recalling that

$$2\alpha = \eta - \tau$$

and performing the trigonometry and arithmetic we see that:

$$\tan 2\alpha = \tan (\eta - \tau) = \frac{\tan \eta - \tan \tau}{1 + \tan \eta \tan \tau}$$

$$= \frac{2a_1a_2 + 2b_1b_2}{a_1^2 + b_1^2 - a_2^2 - b_2^2},$$

which is the same as that found above.

In our formulation, positive ϕ_{-+} means that the anticlockwise component leads the clockwise component. Positive η means that the anticlockwise unit vector is advanced anticlockwise (it lies above the real axis at $t = 0$) while positive τ means that the clockwise unit vector is advanced clockwise (it lies below the real axis at $t = 0$). If the anticlockwise component leads, i.e. it is advanced more than the clockwise is advanced, then the orientation, $(\eta - \tau)/2$, will be positive.

Thus, the rotary spectral representation

$$u_+ = u + iv$$

$$u_- = u - iv$$

for positive frequency with Fourier coefficients

$$\frac{a_1 - ib_1}{2}, \quad \frac{a_2 - ib_2}{2}$$

yields identical results as the conceptually simpler but algebraically more difficult velocity hodograph model with $\pm$ frequencies.

Additional ellipse parameters calculated by RSPEC include the ellipse semimajor axis, the semiminor to semimajor axis ratio and two other coherencies: the minimum and maximum coherencies squared between

the cartesian velocity component of a single vector time series. The semimajor axis, SMAJOR, is the r.m.s. fluctuation along the principal axis of variance and the axis ratio, RATIO, is the speed ratio in the normal coordinate system.

2.3 Basic Scaling

Several steps go into the calculation of averaged and scaled spectral estimates. They begin with the finite Fourier transform over the interval $0 \leq t \leq T$:

$$X(f, T) = \int_0^T x(t) e^{i2\pi ft} dt$$

In discrete form:

$$X(f, T) = \sum_{n=0}^{N-1} x(nh) e^{i2\pi f t_n} h$$

where h is the sampling interval which we call "step" in the programs. The discrete resrepresentation results in frequency domain sampling at the discrete frequencies $f = k/T = k/Nh$; $k = 0, 1, 2, \dots, N-1$. Thus:

$$X(f, T) = h \sum_{n=0}^{N-1} x(n) e^{i2\pi k n/N}, \quad k=0, 1, 2, \dots, N-1.$$

The function $X(f, T)$ is generally complex and is proportional to the Fourier coefficients of the time series.

One-sided power spectra derive from:

$$G_X(f) = \frac{2}{Nh} \left| X(f, T) \right|^2,$$

the factor of two arising from the two-sided nature of the function $X(f, T)$, i.e. it folds about the Nyquist frequency occurring at $k = N/2$. The FFT subroutine actually outputs the quantity $X_k = X(f, T)/h$; therefore, the basic power spectral estimate is computed from:

$$G_{xx}(f) = \frac{2h}{N} |X_k|^2.$$

Assuming that the input time series has units of cm/sec and the sampling interval has units of hours, the spectral density function $G_{xx}(f)$ will have units of $(\text{cm/sec})^2/\text{c.p.h.}$ Note that rotary spectra have the same initial scaling with the exception of the factor of two since G_{++} and G_{--} are each one-sided spectra. Thus,

$$G_{++} + G_{--} = G_{uu} + G_{vv}.$$

The basic power spectral estimate is scaled in accordance with three options: 1) augmenting the record length with zeros; 2) windowing; and 3) pre-whitening. Record length augmentation alters the frequency domain sampling to $f = k/N'h$ from $f = k/Nh$ where N and N' are the original and augmented number of samples. Thus a factor of N/N' is applied to scale the variance up to its nonaugmented value. Windowing also reduces variance by weighting the ends of the records less than the middle. Thus a factor of:

$$\left[\int_0^T w^2(t) dt \right]^{-1}$$

is applied to scale the variance back up to its nonwindowed amount. Note that this procedure is only valid if the window function is uncorrelated with the data. The existence of trends, nonstationary variance, or periodic components in phase with the particular window

could cause the integral of the scaled spectral density to differ slightly from the original time series variance. This is generally the case and attempts at further rectification without an a priori knowledge of the cause would lead to an alteration of the spectral slope which is not desirable.

The basic scaling including the correction for record length augmentation is accomplished via the parameter called FACTOR which is the sampling interval (STEP) divided by the original number of samples (LENGTH). Further corrections for windowing and/or pre-whitening are chosen as options as discussed in the following two sections.

2.4 Windowing

The following six windows are available: 1) Boxcar or no-shading; 2) 10% cosine taper; 3) Hanning; 4) Hamming; 5) Lanczos; and 6) Parzen. Their shape and discrete formulations are given in the accompanying figures (1 through 5) with the exception of the boxcar which is unity. The 10% cosine taper is generally adequate.

2.5 Pre-whitening/Post-coloring

Pre-whitening/Post-coloring is a tool used for reducing the leakage from low to high frequency estimates e. g. see Blackman and Tukey (1956) for a discussion of finite record length effects. The pre-whitening filter available herein is a first difference high pass filter of the form:

$$y(t) = x(t + h) - x(t)$$

where $x(t)$ is the original time series and $y(t)$ is the pre-whitened time series. Its frequency response function may be easily calculated by considering its effect upon a component sinusoid. Let $x(t) = C_n e^{i\omega_n t}$

10% COSINE WINDOW

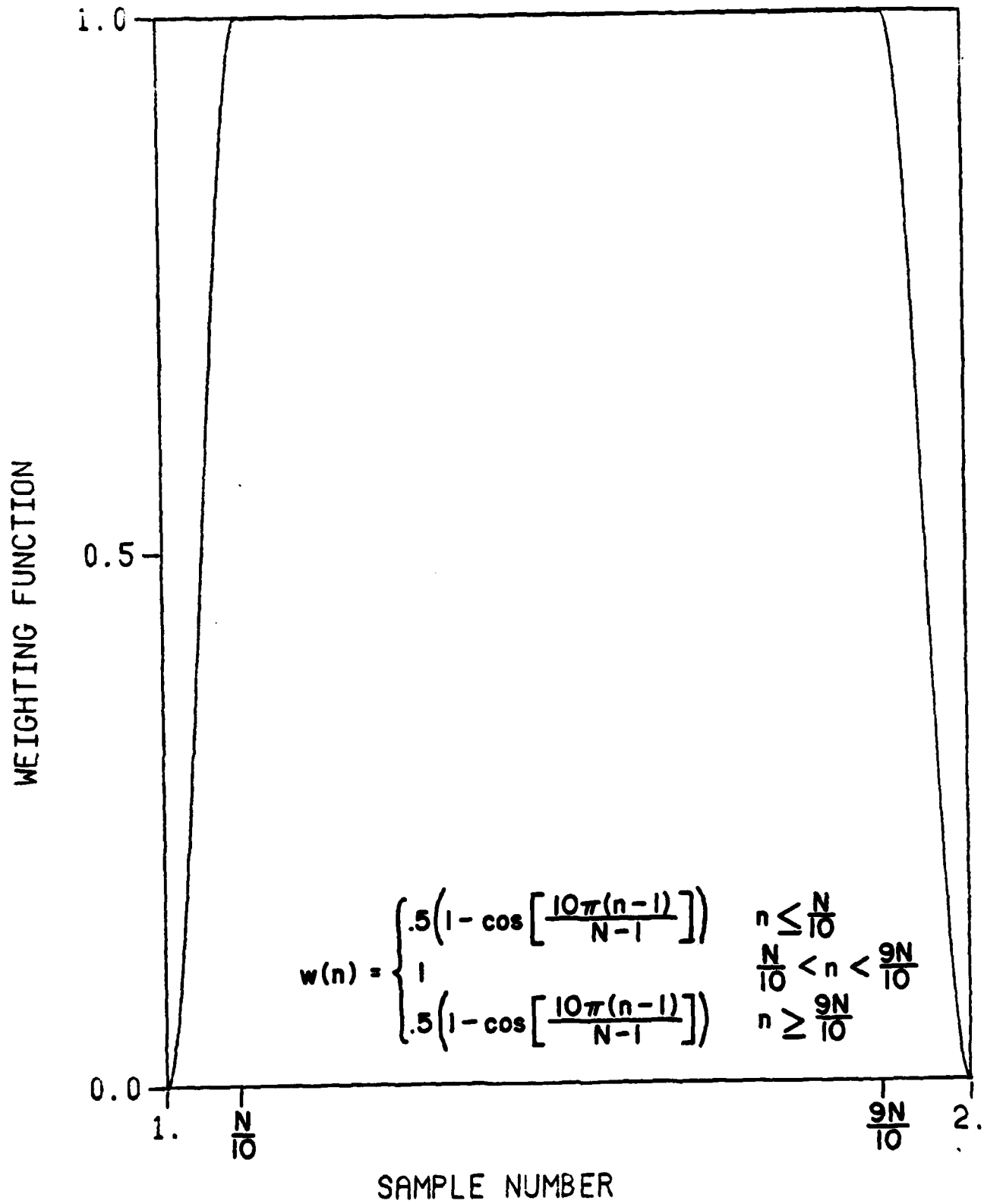


Figure 1.

HANNING WINDOW

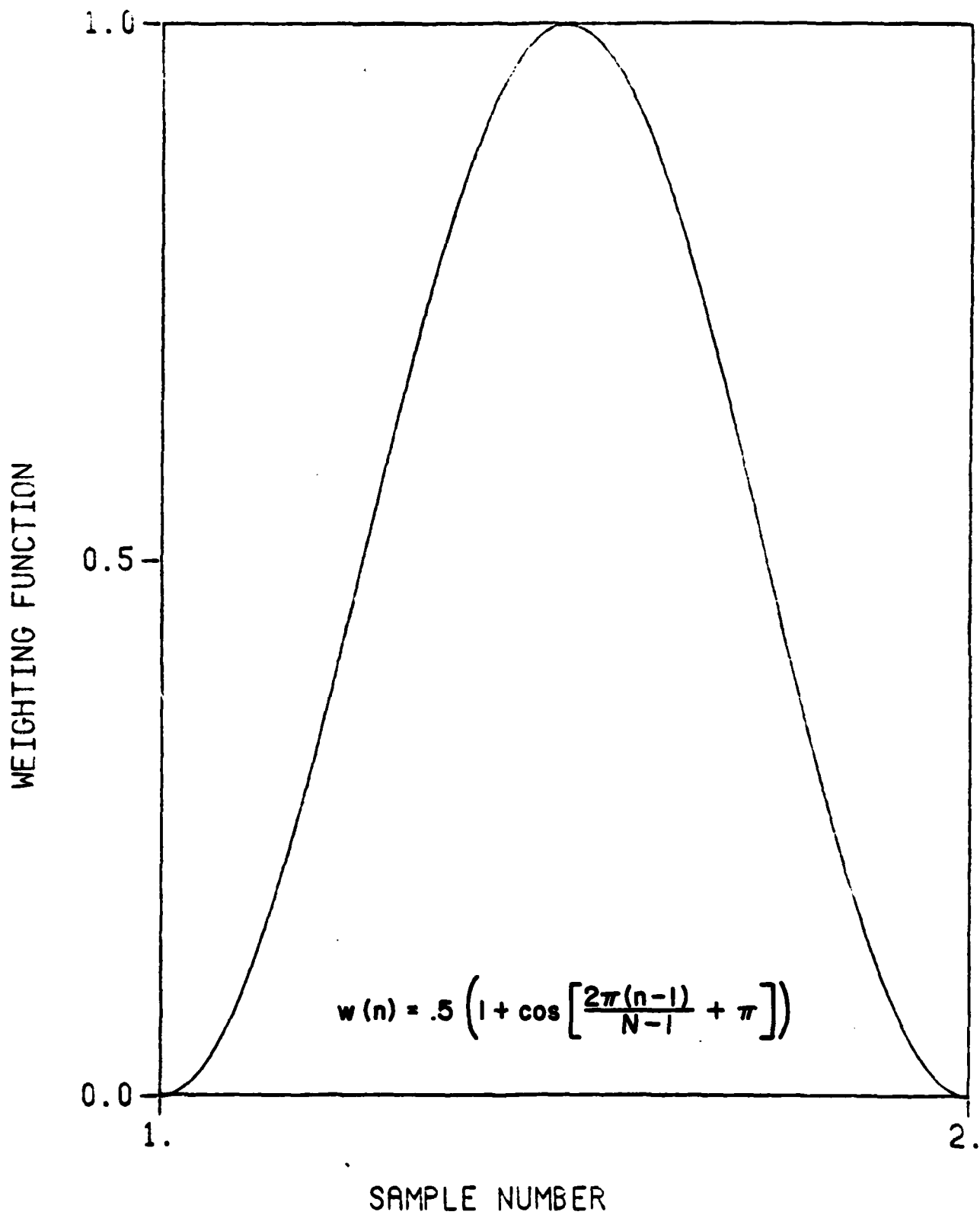


Figure 2.

HAMMING WINDOW

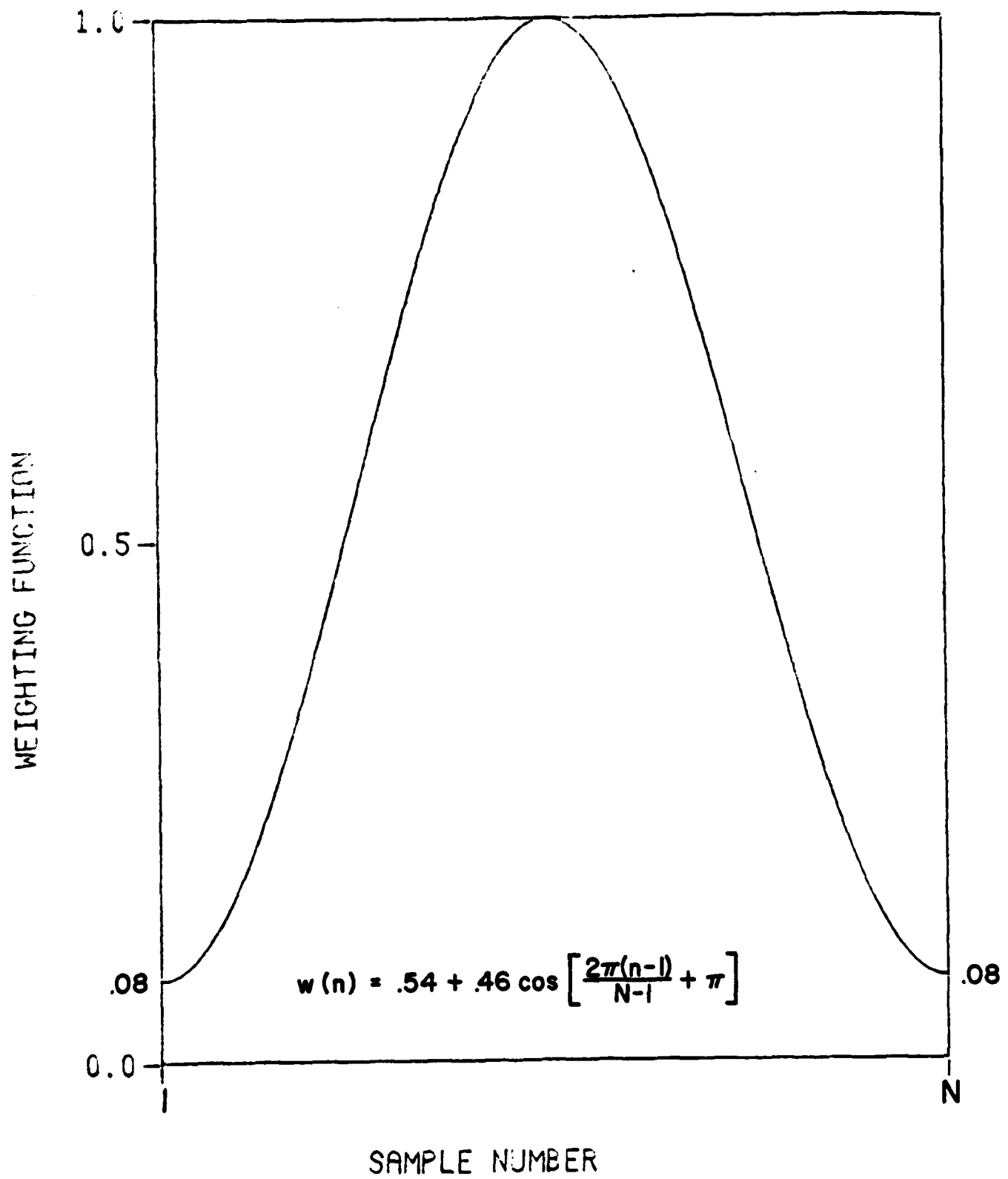


Figure 3.

LANCZOS WINDOW

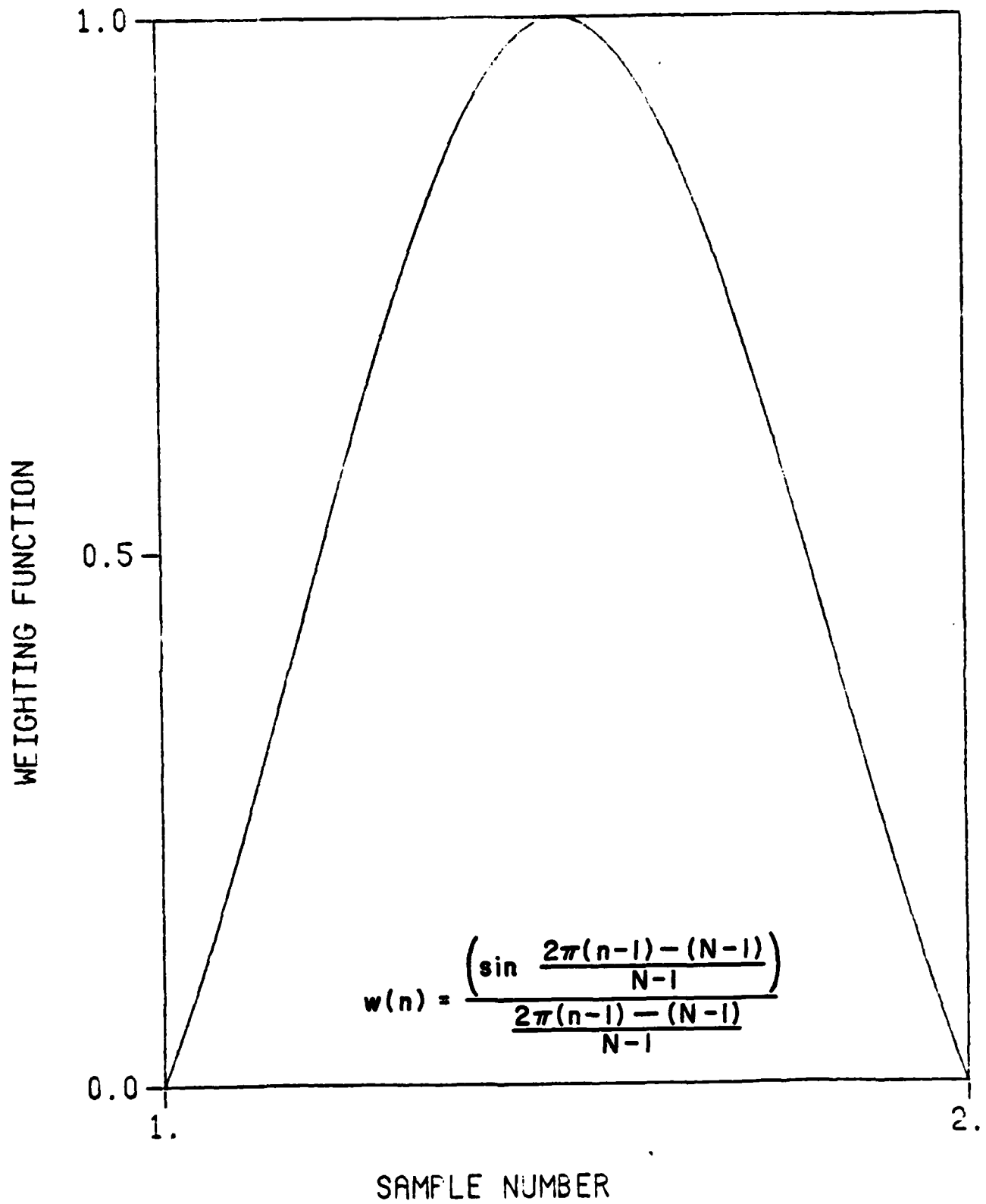
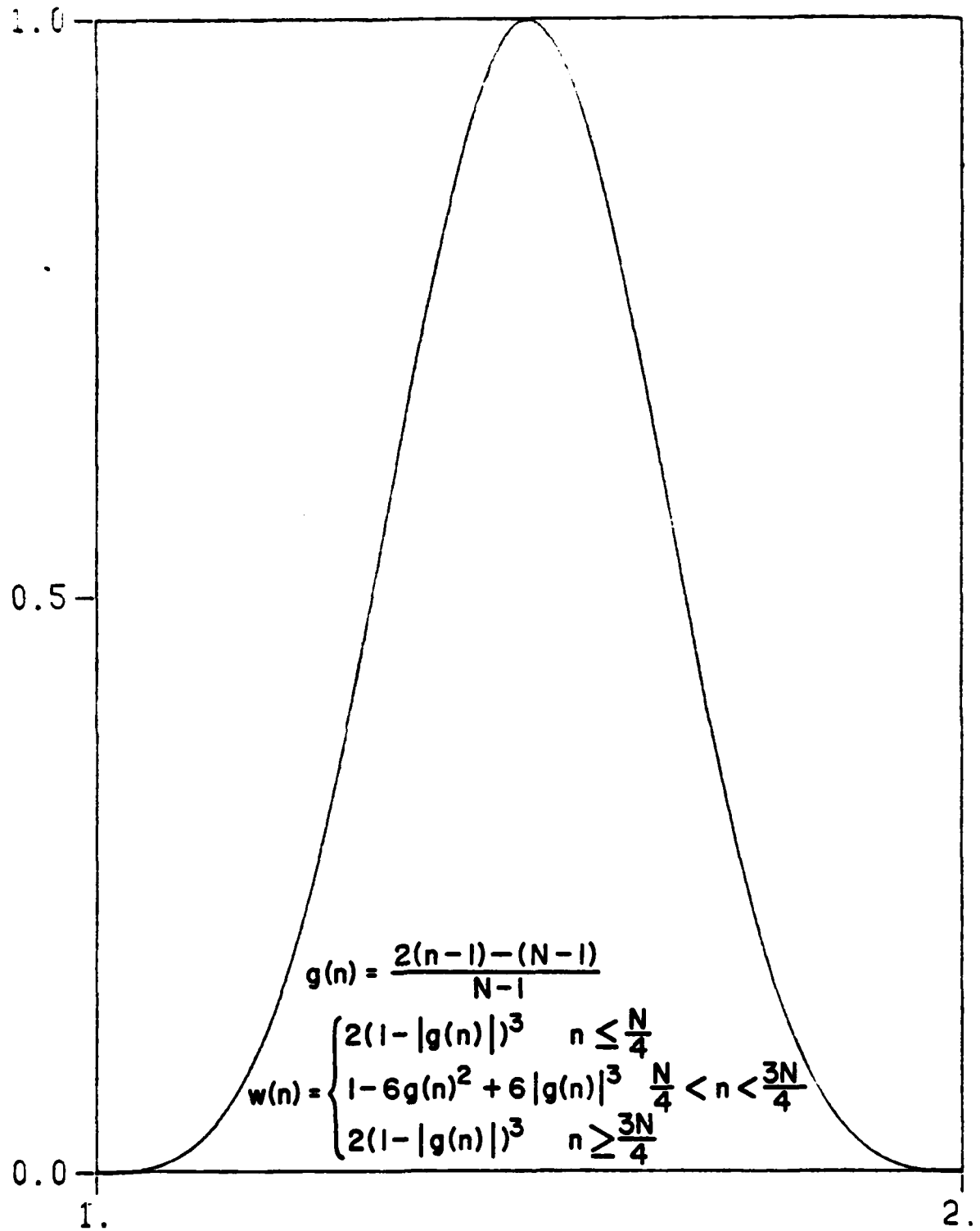


Figure 4.

PARZEN WINDOW

WEIGHTING FUNCTION



SAMPLE NUMBER

Figure 5.

Therefore

$$y(t) = C_n e^{i\omega_n t} \begin{bmatrix} e^{i\omega_n h} & -1 \end{bmatrix}$$

Taking the Fourier transform of both sides results in:

$$Y = \begin{bmatrix} e^{i\omega_n h} & -1 \end{bmatrix} X .$$

where Y and X are the Fourier transforms of y and x . The spectral density functions G_{yy} and G_{xx} are proportional to Y^*Y , and X^*X , thus:

$$G_{yy} \doteq 2(1 - \cos \omega_n h) G_{xx}$$

and employing the identity $\sin^2 = \frac{1}{2} (1 - \cos 2)$ we get

$$G_{yy} \doteq 4 \sin^2 \frac{\omega_n h}{2} G_{xx} .$$

Since $\omega_n \frac{2\pi n}{T}$, $n = 0, 1, 2, \dots, N-1$, the frequency response for the first difference pre-whitening filter is:

$$R(n) = 4 \sin^2 \frac{\pi n h}{T} = 4 \sin^2 \frac{\pi n}{N}$$

Thus it varies from zero at $n = 0$ to 4 at $n = N/2$, the Nyquist frequency. Since the pre-whitening response function for spectral estimates (auto and cross) is real, a correction may be performed by scalar multiplication in the frequency domain by its inverse. This correction procedure is called post-coloring.

2.6 Averaging

Unaveraged spectral estimates computed in the manner described for Gaussian random variables are distributed as chi-squared random variables with two degrees of freedom. Frequency averaging is employed in one of three ways to increase the number of degrees of freedom thereby enabling confidence intervals to be placed about the spectral estimates. The three choices are termed convolution, block, and combination averaging. The first involves a unit boxcar of bandwidth $(2nn + 1)/N'h$ with the raw spectrum (or periodogram), where nn is the number of adjacent fundamental bands to be averaged on each side of the center band, N' is the augmented number of samples (LTRANS) and h is the sampling interval (STEP). This results in a nominal number of degrees of freedom $\nu = 2BT = (2)(2nn + 1)(N/N')$. Nominal is emphasized; the actual number of degrees of freedom depends upon an effective bandwidth and an effective record length T both of which are usually somewhat less than the nominal ones used above as discussed by Blackman and Tukey (1956). Generally, the nominal values will suffice if a modest taper like a 10% cosine is employed and if the smoothed spectrum is fairly uniform over the averaging bandwidth.

2.7 Confidence Interval and Significance Levels

Confidence interval multipliers at the 90% level are printed and plotted for each choice of averaging. These specify the upper and lower limits of the true spectrum with 90% confidence given the estimated spectrum. They are obtained using standard tables for a chi-squared distribution.

The 90% significance level for the null hypothesis on the various estimates of coherence squared (rotary, maximum, minimum, and stability) are printed for each choice of averaging. The null hypothesis that the true coherence is zero can be rejected with 90% confidence is the estimated coherence squared lies above the significance level. Values were obtained by linear least squares fit to the conditional distri-

bution function for sample coherence given that the true coherence is zero as tabulated by Amos and Koopmans (1963).

2.8 Sample Products

The following three figures (6 through 8) show sample products from RSPEC and RCSPEC. Figure 6 shows the anticlockwise and clockwise spectra overplotted on a log-log scale with their associated confidence intervals. Figure 7 shows, from top to bottom, the semiminor to semimajor axis ratio, principal axis orientation, ellipse stability, and maximum and minimum coherence squared. Figure 8 shows the rotary coherencies squared and phase differences for a pair of vector times series computed by RCSPEC.

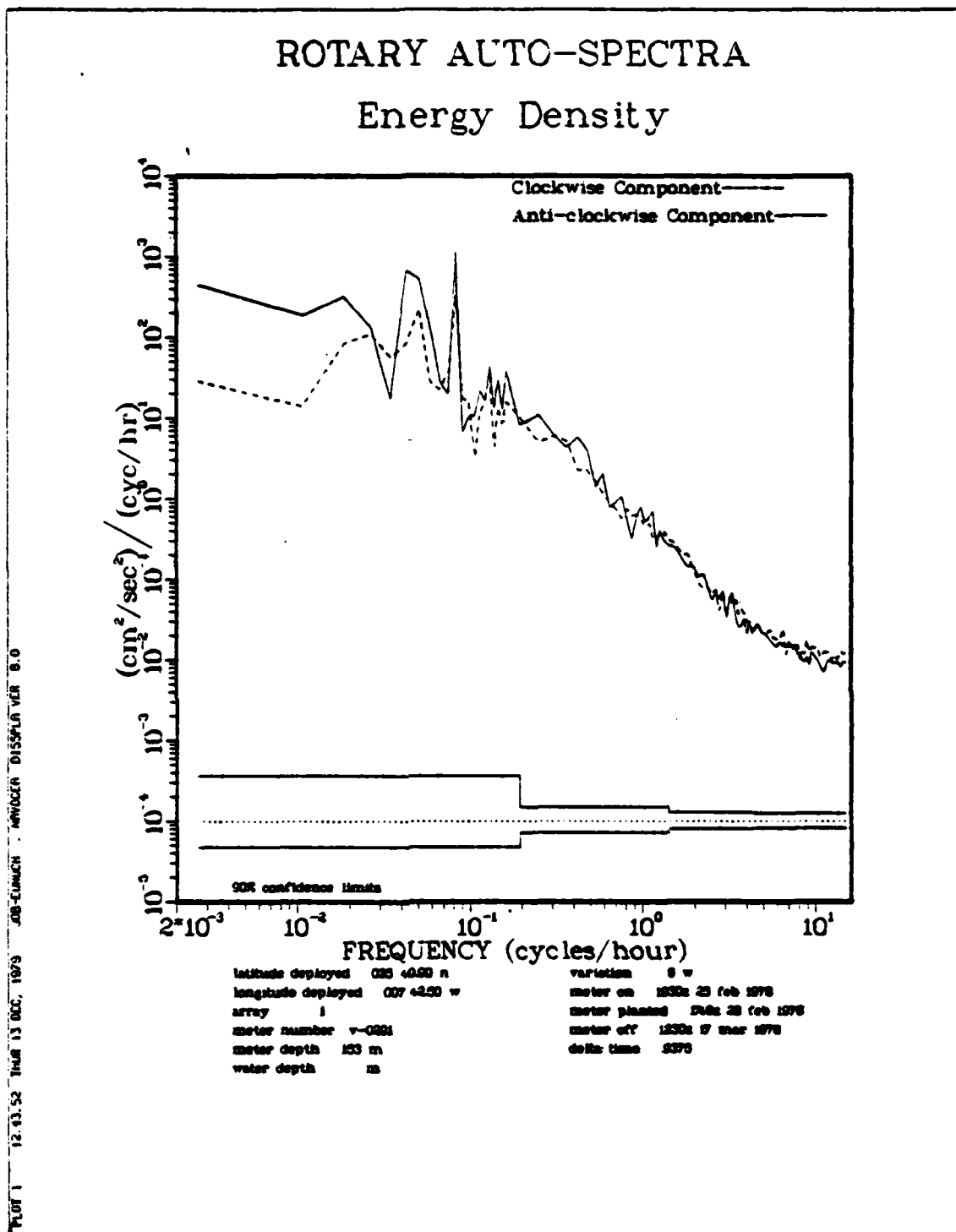
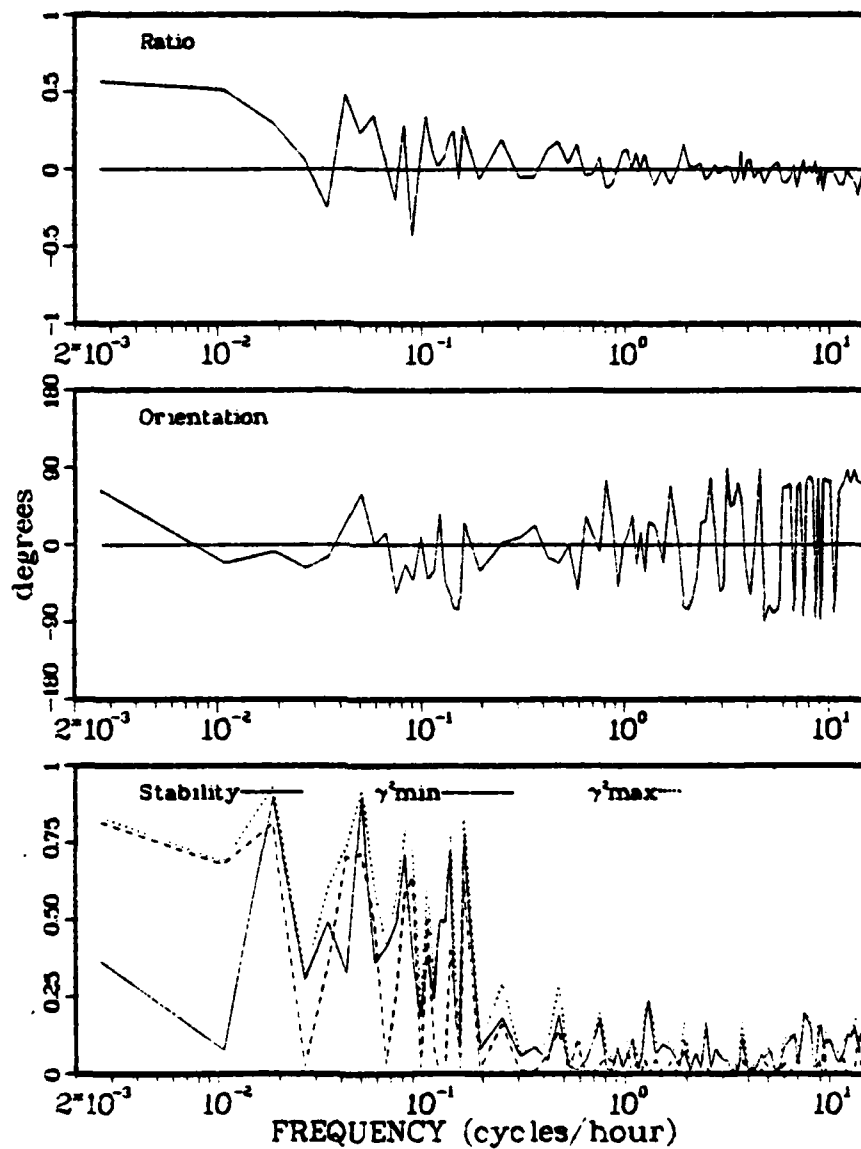


Figure 6. Sample output for rotary spectral analysis program showing energy density plot.

ROTARY AUTO-SPECTRA Stability Ratio Orientation Coherence Squared



PLOT 3 12.46.34 THUR 13 DEC, 1979 JDS-ELUNCH INVOGRA DISPLAY VER 6.0

Figure 7. Sample output for rotary spectral analysis program showing hodograph ellipse parameters.

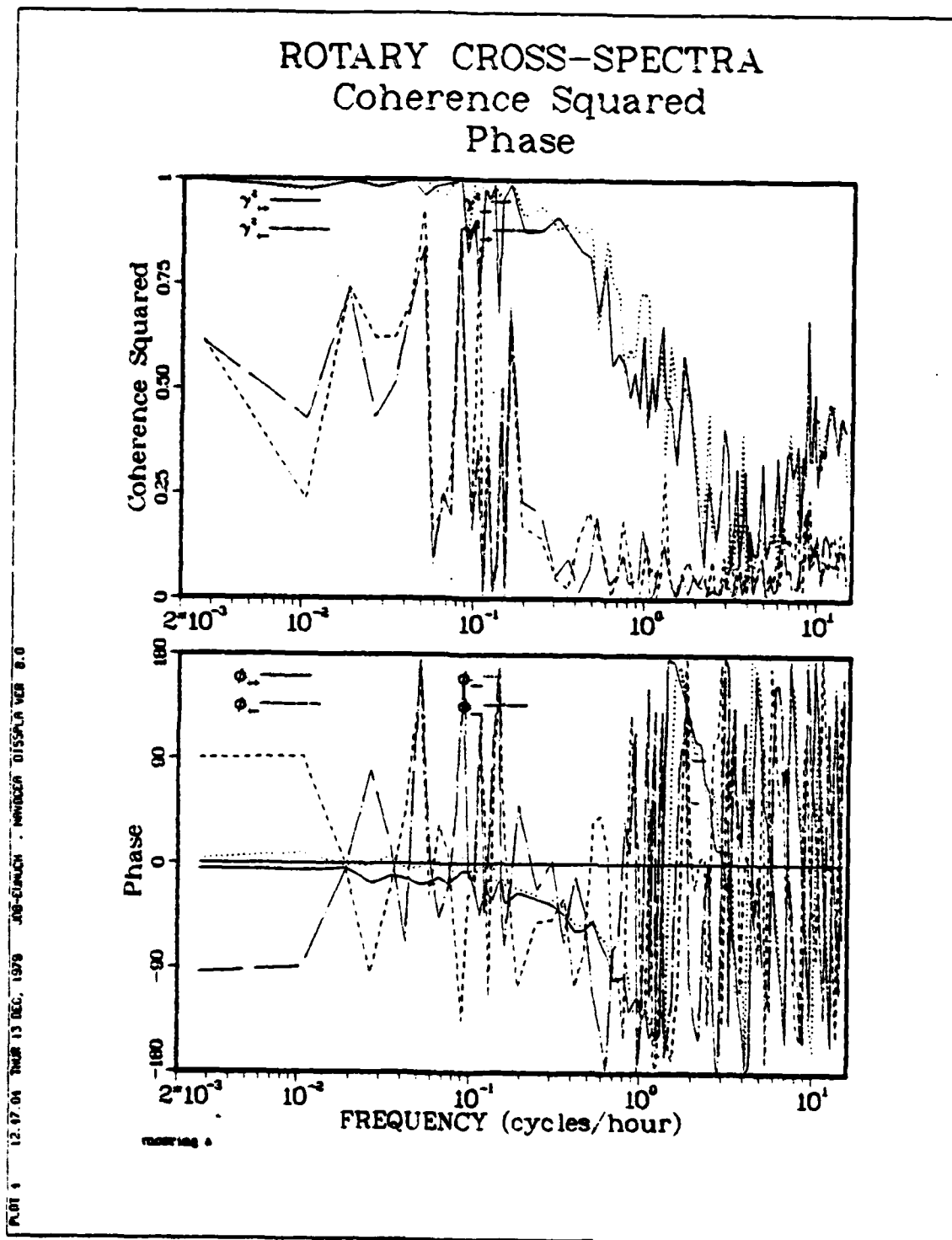


Figure 8. Sample output for rotary spectral analysis program showing plots of cross-spectral products: coherence squared and phase.

3.0 DESCRIPTION OF DATA ANALYSIS PROGRAMS

3.0 DESCRIPTION OF DATA ANALYSIS PROGRAMS

3.1 GET

The GET program should be utilized to remove time series data from a Febfile and store it in datasets which can be accessed by the FOURCO program. The following is a brief discussion of the minimal amount of information necessary to use the GET program. For a more detailed description, contact David Hale (Ext. 4241).

Input Parameters

The Febfile dataset first must be assigned to unit 4. The first input line is read with a format of (A6, 6I5, 2F5.0, A6). These parameters are:

ITYPE Denotes the type of current meter used. The choices are:

AANDER
CTD
TWDCTD
VACM

IBGN The beginning segment for the data desired.

IEND The ending segment for the data desired.

IBGCY The first cycle to read in the beginning segment.

IPTS The total number of data estimates to be stored in the output datasets.

ITV The interval or decimation factor. A value of 2 means every other point will be used. For example, every fourth point will be used if a value of 4 is supplied.

IFILT The type of filter to use:

0 No filter
1 Lopass filter
2 Hipass filter

FREQ These parameters are optional and are not necessary
 DELT for executing GET. Use of these parameters should
 DETND be with guidance of appropriate NAVOCEANO personnel
 only.

The second parameter card is read with a format of (16I5).

NPAIRS The number of variables to be stored. Always use 2
 for both components of velocity.

IVAR (I) These parameters are paired and there should be
 IUN (I) NPAIRS of them. IVAR represents the variable to
 output and IUN is the FORTRAN unit reference number
 that will be associated with the dataset. The GET
 program allocates these files so the user is not
 required to do so.

Time series data is stored in a FEBFILE in u, v component pairs. The FOURCO program requires four input datasets: a v and u component from each of two time series. The v component from the first current meter should be associated with file 7 and the u component with file 8. The v and u components of the second time series should correspond to units 9 and 10, respectively. For example, after first moving the FEBFILE from tape to a file named 'FEBFILE,' the following should be entered:

```

@ USE      4, FEBFILE.
@ XQT      JAY.GET
VACM       1   4 200 4000   1   0
           2   2   7   1   8

```

After a message is printed saying that processing is complete, the program should be executed again to create files 9 and 10.

Should the user desire to examine the header label of a particular segment of a FEBFILE, he can use GET. The header label is output to each of the output datasets so, by editing them, the user can print the label. To save execution time, the value for IPTS may be entered as a zero and NPAIRS may be set to one.

3.2 FOURCO

FOURCO generates complex Fourier coefficients given two sets of time series data. The Singleton Fast Fourier Transform (IEEE Transactions on Audio and Electroacoustics, June, 1969) is incorporated which allows for a wide range of transform lengths. If the user desires, the data may be pre-whitened, and the capability to apply a variety of spectral windows is included.

Input consists of four datasets: one u and v component from each of two time series. Fortran unit number 7 should correspond to the v component of the first time series. The u component of the same time series is referenced by unit number 8. Units 9 and 10 correspond to the v and u components, respectively, of the second time series. All of these datasets should be originally generated from Febfile using the GET program. In addition to these datasets, some parameters are also necessary. They are explained below. The Fourier coefficients are output to units 17, 18, 19 and 20 for use by the RSPEC and RCSPEC programs. Printed output includes the mean and mean square values for each dataset and the header information from each of the Febfiles accessed.

Input Parameters

The following parameters must be supplied by the user after a prompting message is printed. The format is (3I10).

LTRANS The length of the fast Fourier transform to apply. The maximum prime factor is 23 and the size is usually limited by the computer involved. For a more detailed discussion of transform lengths, refer to Singleton's paper.

NWINDO Denotes which spectral window to apply to the time series:

- 0 Boxcar (NO) window
- 1 10% Cosine window
- 2 Hanning window

- 3 Hamming window
- 4 Parzen window
- 5 Lanczos window

KWHITE Denotes if data should be pre-whitened:

- 0 Do not pre-whiten
- 1 Pre-whiten

Caution: Should pre-whitening be implemented, the user is advised to use post-coloring in both RSPEC and RCSPEC.

To execute FOURCO, simply enter @XQT JAY.FOURCO. Then enter the input parameters immediately thereafter.

3.3 RSPEC and RCSPEC

The RSPEC and RCSPEC programs are similar in that they both require the same input parameters and incorporate the same averaging schemes (a discussion of these averaging schemes appears later). However, since RSPEC analyzes one pair of u,v components and RCSPEC analyzes two pairs of u,v components, RSPEC is actually executed twice (using a simple do-loop). Therefore, two sets of input parameters are required (one following the other) and three pairs of output files are generated. Four of these files are for plotting purposes: two confidence interval files (units 14 and 15) and a pair of rotary spectral analysis files (12 and 13). Another pair of files store data printouts (22 and 23). RCSPEC outputs two files: one for plotting purposes (unit 11) and another for outputting the rotary cross spectral statistics (21). The input FOURIER coefficients datasets are associated with unit numbers 17, 18, 19 and 20.

One primary goal in the writing of both programs was to avoid potential core region problems. This was accomplished by swapping data in and out of arrays during the calculation of statistics and the averaging thereof. The input arrays for the Fourier coefficients (U and V in RSPEC and AU, AV, BU, BV in RCSPEC) must be large enough to store

all of the FOURIER coefficients - one half the transform length plus one. However, the arrays used to store the spectral estimates (the 'G' arrays: GPP, GMM, G11, and G22 in RSPEC and the 'U' arrays: UACBPB, UACNBN, UACPNB, UACPAP, UBCBPB, UACNAN, and UBCNBN in RCSPEC) are of variable length. They must be at least greater than double the NN value (or the largest NN value if block averaging). The smaller these arrays are, the more swapping is required thus reducing turnaround time. All of these arrays are complex and also common so if the size is changed, corrections must be made in all of the subroutines that have access to them. The value of 'IASIZE' must be set to the size of the arrays.

Input Parameters

These first three parameters are read from the first record of one of the Fourier coefficient datasets. The format is (2I5,I1).

LTRANS	The size of the Fourier transform applied to the time series data.
LENGTH	The number of estimates in the original time series before the Fourier transform is applied.
NWINDO	Denotes the spectral window used before the Fourier transform was applied.

The following parameters must be supplied by the user after a prompting message is printed. The format is (F10.0,5I10).

STEP	The time increment in hours between the estimates in the original time series.
PSTCLR	Denotes if the spectral estimates should be post-colored or not: 0 Do not apply post-coloring 1 Apply post-coloring
IWSOPT	Denotes whether spectral window scaling should be applied: 0 Do not apply spectral window scaling 1 Apply spectral window scaling

IOPT Denotes the type of averaging to use:

- 1 Block averaging
- 2 Convolution averaging
- 3 Combination of block and convolution averaging

NN The number of adjacent frequency bands to use when convolution averaging. Not applicable when implementing only block averaging.

NUMBNN The number of NN values used when block averaging. This number must be greater than 0 and less than 31. Not applicable when only convolution averaging.

The next parameters are optional and are only required for block or combination averaging. They are pairs of ISTART and NNS values. NUMBNN pairs must be supplied after a prompting message is printed. The format is 14I5 and if more than 7 pairs are required 2 or more lines of input will be necessary.

ISTART Refers to a position in the Fourier coefficient arrays to begin calculating specific estimates for block averaging with the current NNS value supplied.

NNS The number of adjacent frequency bands to use when block averaging a particular group of spectral estimates.

Execution of RSPEC is initiated with an @XQT JAY.RSPEC. To begin the RCSPEC program, simply enter @XQT JAY.RCSPEC.

Block Averaging

Block averaging produces independent estimates. To specify block averaging, input a value of 1 for IOPT. NUMBNN is the number of NN values to use and should be greater than, or equal to, 1 and less than 31. Also, the number of ISTART and NNS pairs should equal to the value input for NUMBNN. The value of NN applies only to convolution or combination averaging and is meaningless for block averaging. NNS

values specify the number of adjacent frequency bands averaged on each side of the central estimate. For example, if NNS = 2:

Unaveraged Estimates	. . . <u>X X X X X</u> <u>X X X X X</u> X . . .
Averaged Estimates	. . . Y Y . . .

In this example note that 5 estimates are averaged to produce one averaged estimate. The 5 new estimates are averaged for the next independent value. This will produce an averaged dataset 1/5 the length of the number of Fourier coefficients input. The ISTART values refer to the Fourier coefficient pairs to use when beginning block averaging. Often, the user will input the first ISTART value as 2 to avoid the zero frequency estimate. For example, if:

ISTART (1) = 2	NNS (1) = 2
ISTART (2) = 7	NNS (2) = 1

Unaverage Estimates	X ₁ <u>X₂ X₃ X₄ X₅ X₆</u> <u>X₇ X₈ X₉ X₁₀ X₁₁ X₁₂</u> X ₁₃ . . .
Averaged Estimates	Y ₁ Y ₂ Y ₃

If NUMBNN is equal to 2, the rest of the estimates will be averaged in blocks of three. The printout will print the corresponding periods and frequencies for each value. Note in the previous example that if ISTART (2) had equaled 8, estimate #7 would have not been averaged at all. Care should be taken to avoid this situation. Block averaging offers the user the capability to use different NN values and therefore to vary the numbers of degrees of freedom with frequency. To relate the ISTART values to specific frequencies, one could first run the data through the same program implementing convolution averaging. The sequence numbers generated correspond to ISTART values. Any RSPEC or RCSPEC output that used convolution averaging and used the same transform length and step value will contain sequence numbers applicable to all other datasets using like transform lengths and steps. NNS values may be repeated and

0 may be used when unaveraged estimates are desired. However, should unaveraged estimates be desired for the first frequency bands (the low frequency end) use combination averaging (see next section) with NN equal to 0. If unaveraged estimates are desired for all frequencies use convolution averaging with NN equal to 0.

Convolution Averaging

Convolution, or running, averaging maintains resolution by sampling each point in the frequency domain. To specify convolution averaging, input a value of 2 for IOPT and any value greater than, or equal to, 0 for NN. Any value specified for NUMBNN is meaningless and no input is required for the ISTART and NNS arrays. NN specifies the number of adjacent frequency bands to be used in averaging on each side of the central estimate. For example if NN = 2:

Unaveraged Estimates	$X_1 \ X_2 \ X_3 \ X_4 \ . \ . \ .$
New Unaveraged Estimates	$\underbrace{X_3 \ X_2 \ X_1 \ X_2 \ X_3}_{\text{Averaging Window}} \ X_4 \ . \ . \ .$
Averaged Estimates	$Y_1 \ Y_2$

The same method is used with the last estimates. Therefore, when convolution averaging, the same number of values are output as the number of Fourier coefficient pairs input. Should the user not desire any averaging, an NN value of 0 will produce unaveraged periodogram estimates.

Combination Averaging

Combination averaging allows the user to incorporate both convolution and block averaging. An IOPT value of 3 specifies convolution averaging. Appropriate values must be input for NN, NUMBNN, and the ISTART and NNS arrays. Convolution averaging must be implemented first

on the low frequency end of the estimates and then block averaging is used for the rest of the estimates. Convolution averaging involves the first ISTART (1) - 1 estimates. Values are "folded" around the first estimate but not where block averaging begins. For example, if:

NN = 1
 ISTART (1) = 5
 NNS (1) = 2

Unaveraged Estimates	<u>X₂ X₁ X₂ X₃ X₄</u>	<u>X₅ X₆ X₇ X₈ X₉</u>	<u>X₁₀ X₁₁ X₁₂ X₁₃ X₁₄</u>	...
Averaged Estimates	Y ₁ Y ₂ Y ₃	Y ₄	Y ₅	...

This would continue for the remainder of the dataset. NN and NNS values may be any number greater than or equal to zero. NUMBNN must be 30 or less. The first ISTART value should be large enough to allow for the desired amount of convolution averaging and other ISTART values should be chosen carefully to involve all estimates in the averaging.

4.0 COMPREHENSIVE RUNSTREAM

4.0 COMPREHENSIVE RUNSTREAM

To simplify its operation, all of the components of this package have been combined into one runstream. The user is required only to copy the desired Febfile from tape into a dataset named 'FEBFILE.' Then he simply enters, '@ADD JAY.RUNALL' and the entire package is executed. The runstream assigns all necessary files and even deletes old files referred by the same file names. Afterwards, the printed output from RCSPEC can be referenced by FORTRAN unit 21 and the two RSPEC printouts are associated with units 22 and 23. If desired, these may be routed to a printer.

The input parameters to the various programs may be altered by editing the 'JAY.RUNALL' file. Also, should the user desire to analyze the data from two different Febfiles, a slight modification would be necessary. For example, the Feb files would have to be copied to two different data sets and another '@USE' statement should be placed before the second '@XQT JAY.GET' command.

5.0 DESCRIPTION OF PLOTTING
PROGRAM

5.0 DESCRIPTION OF PLOTTING PROGRAMS

The graphics output program is written in four sections each of which controls a different plot.

FIRST PLOT

THIS PROGRAM SEGMENT PLOTS THE CLOCKWISE AND ANTI-CLOCKWISE COMPONENTS OF ENERGY DENSITY ON A SINGLE COORDINATE FRAME. 90% CONFIDENCE LIMITS ARE PLOTTED ON THE SAME AXIS SYSTEM USING A DECADE LINE AS REFERENCE BASE.

NOTE: IF USER DOES NOT SPECIFY A LOG-LOG AXIS SET-UP (NPL (1) =3 OR -3), CONFIDENCE LIMITS PLOT WILL APPEAR DISTORTED.

SECOND PLOT

THIS PROGRAM SEGMENT PLOTS THE SAME INFORMATION AS THE FIRST PLOT. THE FORMAT IS CHANGED, HOWEVER, SO THAT THE CLOCKWISE AND ANTICLOCKWISE COMPONENTS ARE PLOTTED ON SEPARATE COORDINATE FRAMES.

THIRD PLOT

THIS PROGRAM SEGMENT PLOTS VARIOUS HODOGRAPH PARAMETERS FOR ROTARY AUTO-SPECTRAL DATA.

- THE LOWER FRAME CONTAINS PLOTS OF STABILITY AND MAXIMUM AND MINIMUM COHERENCE SQUARED VS. FREQUENCY
- THE MIDDLE FRAME CONTAINS A PLOT OF ORIENTATION (IN DEGREES) VS. FREQUENCY
- THE UPPER FRAME CONTAINS A PLOT OF RATIO VS. FREQUENCY

FOURTH PLOT

THIS PROGRAM SEGMENT PLOTS ROTARY CROSS SPECTRAL DATA ON TWO COORDINATE FRAMES:

- THE UPPER FRAME CONTAINS PLOTS OF THE VARIOUS COHERENCIES (SQUARED) OF CLOCKWISE AND ANTICLOCKWISE COMPONENTS AT THE TWO MOORINGS AS FUNCTIONS OF FREQUENCY
- THE LOWER FRAME CONTAINS PLOTS OF THE CORRESPONDING PHASES

NOTATIONAL CONVENTIONS:

(++) =>	ANTICLOCKWISE AT A	ANTICLOCKWISE AT B
(+-) =>	ANTICLOCKWISE AT A	CLOCKWISE AT B
(--) =>	CLOCKWISE AT A	CLOCKWISE AT B
(-+) =>	CLOCKWISE AT A	ANTICLOCKWISE AT B

One of the most important aspects of this plotting package is user control. The user can exercise the following options:

1. Which of the four available plots will be drawn.

On the plots to be drawn:

2. What titles and labels will appear on the plot.
3. What type of axis scaling will be used.
4. Whether, or not, mooring data will be included on the plot.
5. The number of data points which will be plotted.

These options are exercised by means of a control file containing records on card images in (10I5) format. Basically the first card controls option 2; the second controls options 1 and 3; and the remaining cards control options 4 and 5.

USER CONTROL OF THE GRAPHICS OUTPUT IS ACCOMPLISHED VIA TWO DATA SETS WHOSE FORTRAN REFERENCE NUMBERS ARE CALLED KU AND KD IN THE MAIN PROGRAM. KU CONTAINS CONTROL DATA AND KD CONTAINS USER SUPPLIED LABELS (SEE TABLE 1 FOR LIST OF AVAILABLE LABELS).

KU (ALL RECORDS ARE 15I5 FORMAT)

FIRST CARD - ONE FIELD READ (ENTERS AS KLAB)

FIELD CONTAINS:

1	=>	USER SUPPLIED LABELS ARE TO BE READ INTO ARRAY ICAT FROM KD
0	=>	NO USER LABELS SUPPLIED

TABLE 1. LIST OF AVAILABLE LABELS

Item	Label
1	- (FREQUENCY (cycles/hour))
2	- ($(\text{cm}^2/\text{sec}^2)/(\text{cyc}/\text{hr})$)
3	- (Energy Density)
4	- (Clockwise Components)
5	- (Anticlockwise Components)
6	- (ROTARY AUTO-SPECTRA)
7	- (Maximum and Minimum Coherence Squared)
8	- (Stability Ratio Orientation)
9	- (degrees)
10	- (max)
11	- (min)
12	- (Stability)
13	- (ROTARY CROSS-SPECTRA)
14	- (Orientation)
15	- (Coherence Squared)
16	- (-)
17	- (—)
18	- (.....)
19	- (-----)
20	- (—•—)
21	- (γ^2)
22	- (ϕ)
23	- (++)
24	- (--)
25	- (+-)
26	- (-+)
27	- (Ratio)

SECOND CARD - FOUR FIELDS READ (ENTERS AS NPL (1) - (4))

THE FOUR FIELDS CORRESPOND IN ORDER TO THE FOUR PLOTS

FIELD CONTAINS:

0	=>	CORRESPONDING PLOT NOT DRAWN
NEG. NUMBER	=>	CATALOG OF USER SUPPLIED LABELS TO BE USED FOR THE CORRESPONDING PLOT
POS. NUMBER	=>	DEFAULT CATALOG TO BE USED FOR THE CORRESPONDING PLOT

NUMBER WHOSE
ABSOLUTE VALUE

IS 1	=>	X-AXIS LOG	Y-AXIS LINEAR
2	=>	X-AXIS LINEAR	Y-AXIS LINEAR
3	=>	X-AXIS LOG	Y-AXIS LOG

REMAINING CARDS - NINE FIELDS READ (ENTERED AS NBOT, NPTS,
AND IT (1) - (7))

THE REMAINING CARDS CORRESPOND IN ORDER TO THE
PLOTS WHICH ARE TO BE DRAWN

FIELD #1 CONTAINS

0	=>	MOORING DATA NOT INCLUDED ON PLOT
1	=>	MOORING DATA APPEARS ON PLOT

FIELD #2 CONTAINS THE NUMBER OF DATA POINTS WHICH
THE PLOT WILL CONTAIN. IF THIS FIELD IS ZERO OR
BLANK, THE PROGRAM WILL ASSUME ALL POINTS ARE TO BE
INCLUDED.

FIELDS #3 - 9 WILL ONLY BE READ IF USER SUPPLIED
LABELS ARE TO BE USED IN THE CORRESPONDING PLOT.
THE PROGRAM READS THESE VALUES INTO ARRAY IT. SEE
DOCUMENTATION OF ROUTINES TOP AND SETUP FOR FURTHER
EXPLANATION.

KD (ALL RECORDS ARE 12A6 FORMAT)

EACH RECORD SHOULD CONTAIN A CHARACTER STRING NOT TO EXCEED
71 CHARACTERS FOLLOWED BY "\$". THESE STRINGS WILL BE WRITTEN
USING THE ALPHABET CONVENTIONS OF SUBROUTINE ALFSET, I.E.,
INSTRUCTIONAL STRING OPTIONS ARE AVAILABLE.

NOTE: KD SHOULD CONTAIN NO MORE THAN 15 RECORDS

EXAMPLE:

SUPPOSE DATA SET KU CONTAINS THE FOLLOWING CARDS

```
1
3      0  0  -1
1  3500
0  4098  1  2  3  4  2  5
```

THE PROGRAM WOULD:

- READ THE LABELS FROM KD INTO ICAT
- DRAW THE FIRST PLOT USING DEFAULT TITLE AND LABELS ON A LOG-LOG AXIS
- USE ONLY THE FIRST 3500 FREQUENCY VALUES AND INCLUDE MOORING DATA AT THE BOTTOM OF THE FIRST PLOT
- DRAW THE FOURTH PLOT ON A LOG-LINEAR AXIS USING THE FIRST THREE RECORDS FROM KD AS THE LINES OF TITLE, THE FOURTH RECORD AS X-AXIS LABEL, AND THE SECOND AND FIFTH RECORDS AS Y-AXIS LABELS
- THE FOURTH PLOT WOULD USE 4098 POINTS AND WOULD NOT CONTAIN ANY MOORING INFORMATION

FORTRAN REFERENCE NUMBERS FOR DATA SETS ARE:

```
KR  ROTARY DATA
KRX  ROTARY CROSS DATA
KCI  CONFIDENCE INTERVAL DATA
KDD  DEFAULT LABELS CATALOG
KF   DATA SET CONTAINING CURRENT METER INFORMATION
KU   USER CONTROL CARDS
KD   USER SUPPLIED LABELS CATALOG
```

Following is an explanation of the internal sizing parameters which control the physical dimensions of the plots. These parameters are preset, based on the assumption that plots will be on an 8 1/2" x 11" frame which would be suitable for publication. Sizing parameters can be changed to create plots of any desired size and are limited only by the capabilities and physical dimensions of the plotter itself. There are four parameters:

- P TOTAL HORIZONTAL SIZE OF THE SPACE WHICH WILL BE DEDICATED TO AXIS FRAMES.
- Q TOTAL VERTICAL SIZE OF THE SPACE WHICH WILL BE DEDICATED TO AXIS FRAME(S). THIS DOES NOT INCLUDE SPACE FOR TITLE OR STORY.

G SIX OF THE VERTICAL SEPARATION OF GRAPH FRAMES WHEN
THERE IS MORE THAN ONE FRAME PER PAGE.

XPOS,YPOS HORIZONTAL AND VERTICAL DISTANCES SEPARATING THE
PHYSICAL ORIGIN OF THE LOWEST PLOT FRAME FROM THE
LOWER LEFT CORNER OF THE PAGE.

The present values for these parameters are*:

P=6.
Q=6.5
G=.5
XPOS=1.5
YPOS=3.0

Figure 9 shows the relationship of the various parameters and introduces
the additional parameter QQ which is internally calculated based on the
values of Q, G, and N selected by the user.

* All values are in inches.

(binding margin allowance)

Q

(space for lines of title)

SIZING PARAMETERS - P,Q,G, & QQ

- P,Q, AND G ARE SET IN THE PROGRAM.
- QQ IS COMPUTED USING THE FORMULA

$$QQ = (Q + G) / N - G$$

WHERE N (=1,2,OR 3) IS THE NUMBER OF
FRAMES ON THE PAGE

QQ

G

(frame for plots)

P

(space for current meter data)

Figure 9. Relationship of Sizing Parameters

APPENDICES

- A. SOURCE LISTING - DATA ANALYSIS PROGRAMS
- B. SOURCE LISTING - 'JAY.RUNALL'
- C. SOURCE LISTING - 'MAP' DATASET LISTINGS
- D. SOURCE LISTING - 'JAY.USER'
- E. SOURCE LISTING - PLOTTING PROGRAM

APPENDIX A - SOURCE LISTINGS

DATA ANALYSIS PROGRAMS

I. GET

II. FOURCO

A. MAIN PROGRAM

B. SUBPROGRAMS

1. FFT
2. PWHITE
3. WINDOW

III. RSPEC

A. MAIN PROGRAM

B. SUBPROGRAMS

1. CALC
2. SWAP
3. AVRG
4. BAVRG
5. STAT
6. SL
7. CONINT

IV. RCSPEC

A. MAIN PROGRAM

B. SUBPROGRAMS

1. CALC
2. SWAP
3. AVRG
4. BAVRG
5. PHACOH
6. SL

I. GET

```

COMPILER (DIAG=3)
CALL GETDAT
STOP
SUBROUTINE GETDAT
COMMON/DIAGS/MSGR,MSGW,NNNR,NNNW,NNIP,NNA,NNI,NNF,IRST,IWST
COMMON/XD/LF1,LF2,NPAIRS,IVAR(10),IUN(10),ICT(10)
DIMENSION NTYPE(5),ICYCLS(5),INUM(5)
DATA NTYPE/6HARNDER,6HCTD,6HTWDCTD,6HVACM,6HTEST/
DATA ICYCLS/1660,3000,1000,3000,3000/INUM/6,3,10,3,3/
DATA NNI,NNA,NNF/100,100,100/
READ(5,10) ITYPE,IBGN,IEND,IBGCY,IPTS,ITV,IFILT,FREQ,DELT,DETND
10 FORMAT(A6,6I5,2F5.0,A6)
READ(5,15) NPAIRS,(IVAR(I),IUN(I),I=1,NPAIRS)
15 FORMAT(16I5)
CALL FILGEN(ITV,IFILT,DELT,FREQ)
DO 25 I=1,5
25 IF(ITYPE.EQ.NTYPE(I)) I1=I
   NNNR=ICYCLS(I1)
   NNIP=INUM(I1)
   CALL BREAK(IBGN,IEND,IBGCY,IPTS,ITV)
   IF(ITV.GT.1.OR.IFILT.NE.0) CALL FILTER(ITV,IFILT,IBGCY)
   IF(DETND.EQ.6HDETRND) CALL DETRND
   WRITE(6,90)
90  FORMAT(/,5X,/'♦♦ DATA RETRIEVAL COMPLETE, READY FOR PROCESSING')
   CALL ERTRAN(6,/'@FREE 2. . ')
   RETURN
SUBROUTINE FILGEN(ITV,IFILT,DELT,FREQ)
COMMON/XD/LF1,LF2,NPAIRS
COMMON/FILL/FILT(201),FILT1(201),SER(201)
LF1=101
IF(IFILT.EQ.0) LF1=1
LF2=101
IF(ITV.EQ.1) LF2=1
IF(ITV.EQ.1) GO TO 10
FITV=1./FLOAT(ITV)
CALL GENFLT(LF2,2,1,5,FITV,FILT)
10 CONTINUE
IF(IFILT.EQ.0) GO TO 15
DD=DELT*(1./60.)*FLOAT(ITV)*FREQ*2.
CALL GENFLT(LF1,2,IFILT,5,DD,FILT1)
15 CONTINUE
RETURN
SUBROUTINE GENFLT(M,JA,JB,JC,F,SER)
WRITE(6,6666)M,JA,JB,JC,F
6666 FORMAT(/ GENFLT: M=',I6,' JA=',I3,' JB=',I3,
/ JC=',I3,' F=',G12.6)
DIMENSION SER(1)

```

GET (CONT'D.)

```

PI=3.14159265
FM=M
DO 100 I=1,M
FI=I
U=(FI-1.)/FM
GO TO(40,50,60,61,61),JC
61 IF(U-.00000001)40,40,63
63 IF(JC-4)64,64,65
64 SER(I)= SIN(PI*U) / (PI*U)
GO TO 70
65 SER(I)=(SIN(PI*U) / (PI*U)) **2
GO TO 70
40 SER(I)=1.-U
GO TO 70
60 SER(I)=.5*(1.+COS(PI*U))
GO TO 70
50 IF(ABS(U)-.5)51,52,52
51 SER(I)=1.-6.*U**2+6.*ABS(U)**3
GO TO 70
52 SER(I)=2.*(1.-ABS(U))**3
70 IF(F-.00000001)100,100,71
71 IF(ABS(PI*(FI-1.)*F)-.00001)100,100,73
73 SER(I)=SER(I)*SIN(PI*(FI-1.)*F) / (PI*(FI-1.)*F)
100 CONTINUE
102 IF(JB)130,130,103
103 SUM=.5*SER(I)
DO 104 I=2,M
104 SUM=SUM+SER(I)
DO 129 I=1,M
110 SER(I)=SER(I) / (2.*SUM)
IF(JB-1)129,129,120
120 SER(I)=-SER(I)
IF(I-1)129,121,129
121 SER(I)=SER(I)+1.
129 CONTINUE
130 JAO=JA+2
GO TO(150,150,140,180),JAO
140 CONTINUE
RETURN
150 MB2=M/2
DO 155 I=1,MB2
X=SER(I)
II=M-I+1
SER(I)=SER(II)
155 SER(II)=X
RETURN
180 MM=2*M-1
OR WRITE(6,6900) (I,SER(I),I=1,MM)
6900 FORMAT(15,612.6)
DO 185 I=M,1,-1
II=I+M-1

```

GET (CONT'D.)

```

185 SER(I)=SER(I)
    MM1=M-1
    DO 191 I=1,MM1
        II=MM-I+1
191 SER(I)=SER(II)
    RETURN
    SUBROUTINE BREAK(IBGN,IEND,IBGCY,IPTS,ITV)
    COMMON/RHDR/LR,NR,NBR,NMBR,NMER,NFR,NIR,NAR,IPR(10)
    COMMON/RDATA/VR(10000)
    COMMON/RDOCI/IDOCR(100)
    COMMON/RDOCF/FDOCR(100)
    COMMON/RDOCA/ADOCR(100)
    COMMON/DIAGS/MSGR,MSGW,NNNR,NNNW,NNIP,NNF,NNI,NNA,IRST,IWST
    COMMON/XD/LF1,LF2,NPAIRS,IVAR(10),IUN(10),ICT(10)
    DIMENSION LABL(4)
    CALL ERTRAN(6,'@MSG,T 2.,F//POS/6 . ')
    MSGR=0
    MSGW=0
    L1=LF1-1
    L2=LF2-1
    I1=IBGCY-L1-L2
    I2=IBGCY+(IPTS-1)*ITV+L1+L2
    I3=(I2-I1+1)
    IF(I1.GT.0) GO TO 20
    WRITE(6,5) I1
    5 FORMAT(5X,'ERROR IN BREAK ROUTINE, STARTING INDEX IS ',I5)
    STOP
20 CONTINUE
    DO 30 I=1,NPAIRS
        IJ=IUN(I)
        ICT(I)=I3
        ENCODE(24,25,LABL) IJ
25 FORMAT('@MSG,UP ',I2,'.',F//POS/6 . ')
    CALL ERTRAN(6,LABL)
    REWIND IJ
30 CONTINUE
    DO 60 I=IBGN,IEND
        CALL ZREAD(4,IF,I)
        IF(IF.NE.0) GO TO 75
        KR1=LR*(NR-1)
        DO 50 IZ=1,NPAIRS
            IK=IUN(IZ)
            IC=ICT(IZ)
            IJ=IVAR(IZ)
            IF(I.NE.IBGN) GO TO 35
            NVAR=IPR(IJ)
            AB=FDOCR(2)
            AC=FDOCR(1)
            FDOCR(2)=AB+(1./1440.)*(IBGCY-1)*AC
            FDOCR(1)=AC*FLOAT(ITV)
            WRITE(IK,37) (ADOCR(J),J=1,40)

```

@STARTING CYCLE
 @ENDING CYCLE
 @TOTAL NUMBER OF CYCLES

@LAST INDEX FOR THIS SEGMENT
 @LOOP FOR EACH VARIABLE

GET (CONT'D.)

```

WRITE(1K,38) (FDOCR(J),J=1,7)
WRITE(1K,39) IPTS,NVAR,NMFR,NMNR
36 FORMAT(F10.5)
37 FORMAT(1X,10A6)
38 FORMAT(7F9.4)
39 FORMAT(15,3A6)
FDOCR(2)=AB
FDOCR(1)=AC
35 CONTINUE
K1=(I1-1)*LR+1J
K2=KR1+1J
IF(K1.LE.K2) GO TO 40
I1=I1-NR
GO TO 60
40 CONTINUE
DO 45 I1=K1,K2,LR
WRITE(1K,36) VR(I1)
IC=IC-1
ICT(I2)=IC
IF(IC.EQ.0) GO TO 50
45 CONTINUE
50 IF(ICT(NPAIRS).EQ.0) GO TO 65
I1=1
60 CONTINUE
65 CONTINUE
DO 70 I=1,NPAIRS
IJ=IUN(I)
ENDFILE IJ
REWIND IJ
70 CONTINUE
GO TO 85
75 WRITE(6,80) IF
80 FORMAT(5X,'ERROR IN ZREAD. IF = ',I5)
85 CONTINUE
RETURN
SUBROUTINE ZREAD(IU,IF,IBL)

```

THIS SUBROUTINE IS THE READ HALF OF A NON-FORMATTED
INPUT-OUTPUT PACKAGE. A COPY OF THE DOCUMENTATION CAN
BE OBTAINED FROM Z. R. HALLOCK, X4220.

```

COMMON / RHDR / LR,NR,NBR,NMNR,NMFR,NFR,NIR,NAR,IPR(1)
COMMON / RDOCF/FDOCR(1) /RDOC1/IDOCR(1) /RDOCA/ADOCR(1)
COMMON / RDATA / VR(1)

```

```

COMMON / DIAGS / MSGR,MSGW,NNNR,NNNW,NNIP,NNF,NNI,NNA,IRST,IWST
COMMON / JPOS / JUNIT(30)
DIMENSION IUNIT(30)
LOGICAL B1,B210,B10,B35,B45,B69
DATA MSGR / 2 /
DATA LLSW/1/, IRST/1/

```

GET (CONT'D.)

```

B1=MSGR.EQ.1
B210=MSGR.GE.2.AND.MSGR.LE.10
B10=MSGR.EQ.10
B35=MSGR.EQ.3.OR.MSGR.EQ.5.OR.MSGR.EQ.7.OR.MSGR.EQ.9.OR.MSGR.EQ.10
B45=MSGR.EQ.4.OR.MSGR.EQ.5.OR.MSGR.GE.8.AND.MSGR.LE.10
B69=MSGR.GE.6.AND.MSGR.LE.9

```

```

IBLK=IBL
IPOS=JUNIT(IU)
IF(IPOS.EQ.0) IPOS=1
IF(IBL.EQ.0) IBLK=IUNIT(IU)
IF(IBLK.LT.IPOS) GO TO 5
4 IF(IBLK.EQ.IPOS) GO TO 3

READ(IU,END=99,ERR=98) LQ, (XQ,I=1,4), (XQ,J=1,LQ), NFQ,NIQ,NAQ
IF((NFQ+NIQ+NAQ).EQ.0) GO TO 10
IF(NFQ.GT.0) READ(IU,END=99,ERR=98)
IF(NIQ.GT.0) READ(IU,END=99,ERR=98)
IF(NAQ.GT.0) READ(IU,END=99,ERR=98)
10 CONTINUE

```

```

C1 READ(IU,END=99,ERR=98)
IPOS=IPOS+1
IUNIT(IU)=IPOS
JUNIT(IU)=IPOS
GO TO 4

```

```

5 IF=0
REWIND IU
IF(IBL.EQ.0) IBLK=1
IPOS=1
IUNIT(IU)=IPOS
JUNIT(IU)=IPOS
GO TO 4
3 CONTINUE

```

```

C READ(IU,END=99,ERR=98) LR,NR,NBR,NMBR,NMFR,(IPR(I),I=1,LR),
  NFR,NIR,NAR
  IF(NR.GT.NMNR.OR.LR.GT.NNIP.OR.NFR.GT.NNF.
  OR.NIR.GT.NNI.OR.NAR.GT.NNA) GO TO 95
  IF((NFR+NIR+NAR).EQ.0) GO TO 11
  IF(NFR.GT.0) READ(IU,END=99,ERR=98) (FDOCR(I),I=1,NFR)
  IF(NIR.GT.0) READ(IU,END=99,ERR=98) (IDOCR(I),I=1,NIR)
  IF(NAR.GT.0) READ(IU,END=99,ERR=98) (ADOCR(I),I=1,NAR)
11 CONTINUE

```

```

C1 NL=NR+LR
C1 N1=(IRST-1)+LR+1

```

GET (CONT'D.)

N2=N1+NL-1

READ(IU,END=99,ERR=98)(VR(J),J=N1,N2)

IPDS=IPDS+1

IUNIT(IU)=IPDS

JUNIT(IU)=IPDS

IF(MSGR.EQ.0) GO TO 109

IF(B210) WRITE(6,1000)IU,NMFR,NBR,NMBR,NR,LR,NFR,NIR,NAR

1000 FORMAT(' READ UNIT',I3,'; FILE ',A6,

• ' ; SEGNUM',I4,'; SEGNAM ',A6,'; N=',I6,

• ' ; L=',I4,' NF=',I4,' NI=',I4,' NA=',I4)

IF(B1) WRITE(6,1011) IU,NMFR,NBR,NMBR,NR,LR,NFR,NIR,NAR

1011 FORMAT(' RD ',I4,2X,A6,2X,I4,2X,A6,2X,I6,4I4)

IF(B35) WRITE(6,1012)(IPR(I),I=1,LR)

1012 FORMAT(' PARAMETERS: '12(2X,A6)/(13X,12(2X,A6)))

IF(.NOT.B45) GO TO 110

IF((NFR+NIR+NAR).EQ.0) GO TO 110

WRITE(6,1013)

1013 FORMAT(' ADDL DATA:')

IF(NFR.GT.0)WRITE(6,1100)(FDOCR(I),I=1,NFR)

IF(NIR.GT.0)WRITE(6,1101)(IDOCR(I),I=1,NIR)

IF(NAR.GT.0)WRITE(6,1102)(ADOCR(I),I=1,NAR)

1100 FORMAT(10G11.5)

1101 FORMAT(1X,12I6)

1102 FORMAT(1X,12(A6))

110 IF(.NOT.B69) GO TO 107

JL=IRST+LR

J1=JL-LR+1

WRITE(6,1014)(VR(I),I=J1,JL)

JL=(NR+IRST-1)+LR

J1=JL+1-LR

WRITE(6,1015)(VR(J),J=J1,JL)

• 1014 FORMAT(' FIRST CYCLE:',10G11.5/(13X,10G11.5))

1015 FORMAT(' LAST CYCLE: ',10G11.5/(13X,10G11.5))

107 IF(.NOT.B10) GO TO 108

WRITE(6,1017)

IQ1=IRST

IQ2=IQ1+NR-1

DO 106 I=IQ1,IQ2

JL=I+LR

J1=JL+1-LR

WRITE(6,1016) I,(VR(J),J=J1,JL)

106 CONTINUE

• 1016 FORMAT(5X,I5,3X,10G11.5/(13X,10G11.5))

GET (CONT'D.)

1017 FORMAT(/// LISTING OF DATA///)

C
108 IF=0
IUP=IU
RETURN

C
C
95 IF=5
WRITE(6,1005)NNMR,NNIP,NNF,NNI,NNA,
NR,LR,NFR,NIR,NAR

1005 FORMAT(/// A DIMENSION IS TOO SMALL.///
◆ ' NNMR=',I6,' NNIP=',I6,' NNF=',I6,
◆ ' NNI=',I6,' NNA=',I6/// NR=',I6,
◆ ' LR=',I6,' NFR=',I6,' NIR=',I6,' NAR=',I6//)

1 RETURN

99 IF=2

WRITE(6,1002) IU

1002 FORMAT(' READ ERROR ON UNIT ',I3)
RETURN

99 IF=1

WRITE(6,1001) IU

1001 FORMAT(' EOF ON UNIT ',I3)
RETURN

SUBROUTINE FILTER(ITV,IFILT,IBGCY)

COMMON/RDOCA/ADDCR(1)

COMMON/RDOCF/FDOCR(1)

COMMON/RHDR/LR

COMMON/XD/LF1,LF2,NPAIRS,IVAR(10),IUN(10)

COMMON/FILL/FILT(201),FILT1(201),SER(201)

REWIND 2

LF=201

L1=200

DO 100 I=1,NPAIRS

IJ=IUN(I)

IF(ITV.EQ.1) GO TO 45

READ(IJ,12) (ADDCR(J),J=1,40)

READ(IJ,13) (FDOCR(J),J=1,7)

READ(IJ,11) NPTS,NVAR,NMFR,NMBR

WRITE(2,12) (ADDCR(J),J=1,40)

WRITE(2,13) (FDOCR(J),J=1,7)

WRITE(2,11) NPTS,NVAR,NMFR,NMBR

11 FORMAT(I5,3A6)

12 FORMAT(1X,10A6)

13 FORMAT(7F9.4)

READ(IJ,14) (SER(J),J=1,LF)

14 FORMAT(F10.4)

15 CONTINUE

SUM=0.

DO 20 J=1,LF

20 SUM=SUM+(SER(J)*FILT(J))

21 FORMAT(3A6)

GET (CONT'D.)

```

WRITE(2,14) SUM
DO 25 J=1,L1
25 SER(J)=SER(J+1)
READ(IJ,14,END=30) SER(LF)
GO TO 15
30 ENDFILE 2
REWIND 2
REWIND IJ
READ(2,12) (ADOCR(J),J=1,40)
READ(2,13) (FDOCR(J),J=1,7)
READ(2,11) NPTS,NVAR,NMFR,NMBR
WRITE(IJ,12) (ADOCR(J),J=1,40)
WRITE(IJ,13) (FDOCR(J),J=1,7)
WRITE(IJ,11) NPTS,NVAR,NMFR,NMBR
35 CONTINUE
READ(2,14,END=40) X
WRITE(IJ,14) X
GO TO 35
40 ENDFILE IJ
REWIND 2
REWIND IJ
45 CONTINUE
IF(IFILT.EQ.0) GO TO 70
READ(IJ,12) (ADOCR(J),J=1,40)
READ(IJ,13) (FDOCR(J),J=1,7)
READ(IJ,11) NPTS,NVAR,NMFR,NMBR
WRITE(2,12) (ADOCR(J),J=1,40)
WRITE(2,13) (FDOCR(J),J=1,7)
WRITE(2,11) NPTS,NVAR,NMFR,NMBR
READ(IJ,14) (SER(J),J=1,LF)
50 CONTINUE
SUM=0.
DO 55 J=1,LF
55 SUM=SUM+(SER(J)*FILT1(J))
WRITE(2,14) SUM
DO 60 J=1,LF
60 SER(J)=SER(J+1)
READ(IJ,14,END=65) SER(LF)
GO TO 50
65 CONTINUE
ENDFILE 2
REWIND IJ
REWIND 2
70 CONTINUE
IR=2
IF(IFILT.EQ.0) IR=IJ
IW=IJ
IF(IFILT.EQ.0) IW=2
IITV=ITV-1
READ(IR,12) (ADOCR(J),J=1,40)
READ(IR,13) (FDOCR(J),J=1,7)

```

GET (CONT'D.)

```

READ(IR,11) NPTS,NVAR,NMFR,NMBR
WRITE(IW,12) (ADOCR(J),J=1,40)
WRITE(IW,13) (FDOCR(J),J=1,7)
WRITE(IW,11) NPTS,NVAR,NMFR,NMBR
75 CONTINUE
IITV=IITV+1
READ(IR,14,END=80) X
IKZ=MOD(IITV,ITV)
IF(IKZ.EQ.0) WRITE(IW,14) X
GO TO 75
90 CONTINUE
ENDFILE IW
REWIND IW
REWIND IR
IF(IW.NE.2) GO TO 95
READ(IW,12) (ADOCR(J),J=1,40)
READ(IW,13) (FDOCR(J),J=1,7)
READ(IW,11) NPTS,NVAR,NMFR,NMBR
WRITE(IR,12) (ADOCR(J),J=1,40)
WRITE(IR,13) (FDOCR(J),J=1,7)
WRITE(IR,11) NPTS,NVAR,NMFR,NMBR
95 CONTINUE
READ(IW,14,END=90) X
WRITE(IR,14) X
GO TO 95
90 ENDFILE IR
REWIND IR
REWIND IW
95 CONTINUE
WRITE(6,105) NVAR,IJ
105 FORMAT(5X,A6,' ASSIGNED TO UNIT ',I2)
100 CONTINUE
RETURN
SUBROUTINE DETRND
COMMON/RDOCA/ADOCR(1)
COMMON/RDOCF/FDOCR(1)
COMMON/XD/LF1,LF2,NPAIRS,IVAR(10),IUN(10)
WRITE(6,5)
5 FORMAT(5X,' INTO SUBROUTINE DETRND')
DO 50 J=1,NPAIRS
IU=IUN(J)
REWIND 2
REWIND IU
IFG=1
9 CONTINUE
READ(IU,10) (ADOCR(I),I=1,40)
READ(IU,11) (FDOCR(K),K=1,7)
READ(IU,12) NPTS,NVAR,NMFR,NMBR
10 FORMAT(1X,10A6)
11 FORMAT(7F9.4)
12 FORMAT(15,3A6)

```

GET (CONT'D.)

```

13 FORMAT(F10.5)
   GO TO (15,30,60),IF6
15 S1=0.
   S2=0.
   K=1.
20 READ(IU,13,END=25) X
   S1=S1+X
   S2=S2+X*FLOAT(K)
   K=K+1
   GO TO 20
25 CONTINUE
   REWIND IU
   IF6=2
   GO TO 8
30 CONTINUE
   A1=(NPTS+1)
   A2=NPTS*(NPTS-1)
   H=FDOCR(1)
   T=0.
   B0=((2.+(2.+(NPTS+1)*S1)-6.*S2)/A2
   B1=((12.*S2)-6.*A1*S1)/(H*A2*(NPTS+1))
35 CONTINUE
   READ(IU,13,END=45) X
   XX=X-(B0+B1*T)
   WRITE(2,13) XX
   T=T+H
   GO TO 35
45 CONTINUE
   ENDFILE 2
   REWIND 2
   REWIND IU
   IF6=3
   GO TO 8
60 CONTINUE
   READ(2,13,END=65) X
   WRITE(IU,13) X
   GO TO 60
65 CONTINUE
   ENDFILE IU
   REWIND IU
   REWIND 2
   WRITE(6,46) NVAR
46 FORMAT(5X,A6,' DETRENDED')
50 CONTINUE
   RETURN
   END

```

II. A. FOURCO

PROGRAM- FOURCO
PROGRAMMER- JACK HICKMAN
DATE WRITTEN- MARCH, 1979

SUBROUTINES REQUIRED

PWHITE
WINDOW
FFT

INPUT PARAMETERS

LTRANS THE LENGTH OF THE FOURIER TRANSFORM TO APPLY TO THE
TIME SERIES.
NUMBER THE NUMBER OF ESTIMATES TO BE TRANSFORMED.
NWINDO DENOTES THE SPECTRAL WINDOW TO BE APPLIED TO THE TIME
SERIES BEFORE THE FFT IS PERFORMED:
0 BOXCAR(NO) WINDOW
1 10% COSINE WINDOW
2 HANNING WINDOW
3 HAMMING WINDOW
4 PARZEN WINDOW
5 LANZOS WINDOW.
KWHITE DENOTES IF DATA SHOULD BE PREWHITENED:
0 DO NOT PREWHITEN.
1 PREWHITEN.
A ARRAY CONTAINING THE TIME SERIES ESTIMATES.

VARIABLE LIST

B ARRAY FOR THE IMAGINARY FOURIER COEFFICIENTS. FILLED
WITH ZEROES BEFORE THE TRANSFORM IS APPLIED.
OUT FORTRAN UNIT REFERENCE NUMBER FOR THE PRINT DATASET.
IN FORTRAN UNIT REFERENCE NUMBER FOR THE INPUT PARAMETER
DATASET.
SUM1 KEEPS A RUNNING SUM OF THE TIME SERIES ESTIMATES TO
USE WHEN CALCULATING THE MEAN.
SUM2 KEEPS A RUNNING SUM OF THE SQUARE OF THE TIME SERIES
ESTIMATES TO USE WHEN CALCULATING THE MEAN SQUARED
VALUE.
RNUMB SAME AS NUMBER BUT REAL.
YBAR THE MEAN OF THE INPUT TIME SERIES DATA.
YBARSQ THE MEAN SQUARE VALUE OF THE TIME SERIES DATA.

FOURCO (CONT'D.)

```

C OUTDS          FORTRAN UNIT REFERENCE NUMBER FOR THE FOURIER COEFFI-
C                CIENT DATASET.
C IHD,FHD,IVAR,  VARIABLES TO STORE INFORMATION FROM THE FEBFILE HEADER
C NMFF,NMBF     RECORD.
C I,J          INDEXES.
C
C      DIMENSION IHD(40),FHD(10)
C      REAL A(10000),B(10000)
C      INTEGER OUT,OUTDS
C      COMMON A,B
C      IN=5
C      OUT=6
C
C INPUT PARAMETERS AND ECHO CHECK.
C
C      READ(IN,10) LTRANS,NWINDO,KWHITE
10  FORMAT(3I10)
C      WRITE(OUT,20) LTRANS,NWINDO,KWHITE
20  FORMAT('1'//6X,'LTRANS',5X,'NWINDO',4X,'KWHITE'///3I10)
C
C INPUT TIME SERIES DATA FROM FILE CREATED BY GET PROGRAM
C
C      DO 9999 INDS=7,10
C      READ(INDS,71) (IHD(I),I=1,40)
71  FORMAT(1X,10A6)
C      WRITE(OUT,71) (IHD(I),I=1,40)
C      READ(INDS,72) (FHD(I),I=1,7)
72  FORMAT(7F9,4)
C      WRITE(OUT,72) (FHD(I),I=1,7)
C      READ(INDS,73) NUMBER,IVAR,NMFR,NMBF
73  FORMAT(15,3A6)
C      WRITE(OUT,77) NUMBER,IVAR,NMFR,NMBF
77  FORMAT(' ',15,3A6)
C      READ(INDS,74) (A(I),I=1,NUMBER)
74  FORMAT(F10,5)
C      REWIND 7
C      NFFT=LTRANS/2+1
C
C DETERMINE MEAN AND MEAN SQUARE OF INPUT TIME SERIES.
C
C      SUM1=0.
C      SUM2=0.
C      DO 102 I=1,NUMBER
C          SUM1=SUM1+A(I)
C          SUM2=SUM2+A(I)*A(I)
102  CONTINUE
C      RNUMB=FLOAT(NUMBER)
C      YBAR=SUM1/RNUMB
C      YBARSQ=SUM2/RNUMB-YBAR*YBAR
C      WRITE(OUT,13) YBAR,YBARSQ
13  FORMAT('/// ',3X,'MEAN OF TIME SERIES: ',F12,3///      MEAN SQUARE',
C          ' OF TIME SERIES: ',F12,3)

```

FOURCO (CONT'D.)

```

C
C SUBTRACT MEAN FROM DATA VALUES.
C
      DO 103 I=1,NUMBER
        A(I)=A(I)-YBAR
103    CONTINUE
C
C CALL SUBROUTINE PWHITE TO PREWHITEN DATA IF USER DESIRES.
C
      IF(KWHITE.GT.0)CALL PWHITE(NUMBER)
C
C CALL SUBROUTINE WINDOW TO WINDOW THE DATA IN THE TIME DOMAIN.
C
      CALL WINDOW(NWINDO,NUMBER)
C
C ADD ZEROES TO FILL REMAINDER OF REAL ARRAY TO THE TRANSFORM LENGTH.
C ALSO FILL IMAGINARY ARRAY B WITH ZEROES.
C
      J=NUMBER+1
      DO 104 I=J,LTRANS
        A(I)=0.
104    CONTINUE
      DO 105 I=1,LTRANS
        B(I)=0.
105    CONTINUE
C
C CALL FFT TO DO FAST FOURIER TRANSFORM. MAKE ISN PARAMETER NEGATIVE
C SO COMPLEX NUMBERS WILL HAVE THE FORM A-BI.
C
      CALL FFT(A,B,LTRANS,LTRANS,LTRANS,-1)
      OUTDS=INDS+10
      WRITE(OUTDS,75)LTRANS,NUMBER,NWINDO
75    FORMAT(2I5,11)
      WRITE(OUTDS,76)(A(I),B(I),I=1,NFFT)
76    FORMAT(2A6)
9999 CONTINUE
      STOP
      END

```

II. B.1 FFT

SUBROUTINE FFT (A,B,NTOT,N,NSPAN,ISN)

```

C MULTIVARIATE COMPLEX FOURIER TRANSFORM, COMPUTED IN PLACE
C BY USING MIXED RADIX FAST FOURIER TRANSFORM ALGORITHM.
C BY R. C. SINGLETON, STANFORD RESEARCH INSTITUTE, OCT. 1968
C IEEE TRANS. AU., JUNE 1969.
C ARRAYS A AND B ORIGINALLY HOLD THE REAL AND IMAGINARY
C COMPONENTS OF THE DATA, AND RETURN THE REAL AND IMAGINARY
C COMPONENTS OF THE RESULTING FOURIER COEFFICIENTS.
C MULTIVARIATE DATA IS INDEXED ACCORDING TO THE FORTRAN
C ARRAY ELEMENT SUCCESSOR FUNCTION. WITHOUT LIMIT
C ON THE NUMBER OF IMPLIED MULTIPLE SUBSCRIPTS.
C THE SUBROUTINE IS CALLED ONCE FOR EACH VARIATE.
C THE CALLS FOR A MULTIPLE VARIATE TRANSFORM MAY BE IN ANY ORDER.
C NTOT IS THE TOTAL NUMBER OF COMPLEX DATA VALUES.
C N IS THE DIMENSION OF THE CURRENT VARIABLE.
C NSPAN/N IS THE SPACING OF THE CONSECUTIVE DATA VALUES
C WHILE INDEXING THE CURRENT VARIABLE.
C THE SIGN OF ISN DETERMINES THE SIGN OF THE COMPLEX
C EXPONENTIAL, AND THE MAGNITUDE OF ISN IS NORMALLY ONE.
C A TRI-VARIATE TRANSFORM WITH A(N1,N2,N3), B(N1,N2,N3) IS
C COMPUTED BY:
C   CALL FFT(A,B,N1+N2+N3,N1,N1,1)
C   CALL FFT(A,B,N1+N2+N3,N2,N1+N2,1)
C   CALL FFT(A,B,N1+N2+N3,N3,N1+N2+N3,1)
C FOR A SINGLE-VARIATE TRANSFORM,
C   NTOT = N = NSPAN = (NUMBER OF COMPLEX DATA VALUES), E.G.,
C
C   CALL FFT(A,B,N*N,N,1)
C
C THE DATA MAY ALTERNATELY BE STORED IN A SINGLE COMPLEX ARRAY A,
C THEN THE MAGNITUDE OF ISN IS CHANGED TO TWO TO GIVE
C THE CORRECT INDEXING INCREMENT, AND A(2) USED TO PASS
C THE INITIAL ADDRESS FOR THE SEQUENCE OF IMAGINARY VALUES. E.G.,
C
C   CALL FFT(A,A(2),NTOT,N*NSPAN,2)
C
C ARRAYS AT(MAXP), BT(MAXP), CK(MAXP), SK(MAXP), AND NP(MAXP)
C ARE USED FOR TEMPORARY STORAGE. IF THE AVAILABLE STORAGE IS
C INSUFFICIENT, THE PROGRAM IS TERMINATED BY A STOP AND ISN=0.
C MAXP MUST BE .GE. THE MAXIMUM PRIME FACTOR OF N.
C MAXP MUST BE .GT. THE NUMBER OF PRIME FACTORS OF N.
C IN ADDITION, IF THE SQUARE-FREE PORTION, K, OF N HAS TWO OR MORE
C PRIME FACTORS, THEN MAXP MUST BE .GE. K-1

```

FFT (CONT'D.)

COMPLEX CMLX

DIMENSION A(10000),B(10000)

C ARRAY STORAGE IN NFAC FOR A MAXIMUM OF 11 FACTORS OF N.

! FREE FACTORS MUST BE .LE. 210.

DIMENSION NFAC(11), NP(209)

C ARRAY STORAGE FOR MAXIMUM PRIME FACTOR OF 23

DIMENSION AT(23), CK(23), BT(23), SK(23)

EQUIVALENCE (I,11)

C THE FOLLOWING TWO CONSTANTS SHOULD AGREE WITH THE ARRAY DIMENSIONS.

! MAXF=23

MAXP=209

IF (N .LT. 2) RETURN

INC=15N

RAD=9.0*ATAN(1.0)

S72=RAD/5.0

C72=COS(S72)

S72=SIN(S72)

S120=SQRT(0.75)

IF (15N .GE. 0) GO TO 10

S72=-S72

S120=-S120

RAD=-RAD

INC=-INC

10 NT=INC*NTOT

KS=INC*NSPAN

KSPAN=KS

NN=NT-INC

JC=KS/N

RADF=RAD*FLOAT(JC)*0.5

I=0

JF=0

C DETERMINE FACTORS OF N

M=0

8 K=N

GO TO 20

15 M=M+1

NFAC(M)=4

K=K/16

20 IF (K-(K/16)*16 .EQ. 0) GO TO 15

J=3

JJ=9

GO TO 30

25 M=M+1

NFAC(M)=J

K=K/JJ

30 IF (MOD(K,JJ) .EQ. 0) GO TO 25

J=J+2

JJ=J*J

IF (JJ .LE. K) GO TO 30

IF (K .GT. 4) GO TO 40

KT=M

NFAC(M+1)=K

FFT (CONT'D.)

```

      IF (K .NE. 1) M=M+1
      GO TO 90
40    IF (K-(K/4) .NE. 0) GO TO 50
      M=M+1
      NFAC(M)=2
      K=K/4
50    KT=M
      J=2
60    IF (MOD(K,J) .NE. 0) GO TO 70
      M=M+1
      NFAC(M)=J
      K=K/J
70    J=((J+1)/2)*2+1
      IF (J.LE. K) GO TO 60
80    IF (KT .EQ. 0) GO TO 100
      J=KT
90    M=M+1
      NFAC(M)=NFAC(J)
      J=J-1
      IF (J .NE. 0) GO TO 90
C
C END 000 BLOCK
C COMPUTE THE FOURIER TRANSFORM.
C
100   SD=RADF/FLOAT(KSPAN)
      CD=2.0*SIN(SD)*2
      SD=SIN(SD+SD)
      KK=1
      I=I+1
      IF (NFAC(I) .NE. 2) GO TO 400
C
C TRANSFORM FOR FACTOR OF 2 (INCLUDES ROTATION FACTOR).
C
210   KSPAN=KSPAN/2
      K1=KSPAN+2
      K2=KK+KSPAN
      AK=A(K2)
      BK=B(K2)
      A(K2)=A(KK)-AK
      B(K2)=B(KK)-BK
      A(KK)=A(KK)+AK
      B(KK)=B(KK)+BK
      KK=K2+KSPAN
      IF (KK .LE. NN) GO TO 210
      KK=KK-NN
      IF (KK .LE. JC) GO TO 210
      IF (KK .GT. KSPAN) GO TO 800
220   C1=1.0-CD
      S1=SD
230   K2=KK+KSPAN
      AK=A(KK)-A(K2)
      BK=B(KK)-B(K2)

```

FFT (CONT'D.)

```

A(KK)=A(KK)+A(K2)
B(KK)=B(KK)+B(K2)
A(K2)=C1*AK-S1*BK
B(K2)=S1*AK+C1*BK
KK=K2+KSPAN
IF(KK.LT. NT) GO TO 230
K2=KK-NT
C1=-C1
KK=K1-K2
IF(KK.GT. K2) GO TO 230
AK=C1-(CD*C1+SD*S1)
S1=S1+(SD*C1-CD*S1)

```

```

C
C THE FOLLOWING THREE STATEMENTS COMPENSATE FOR TRUNCATION ERROR.
C IF ROUNDED ARITHMETIC IS USED, SUBSTITUTE:
C

```

```

C1=AK
C1=0.5/(AK*AK+S1*S1)+0.5
S1=C1*S1
C1=C1*AK
KK=KK+JC
IF(KK.LT. K2) GO TO 230
K1=K1+INC+INC
KK=(K1-KSPAN)/2+JC
IF(KK.LE. JC+JC) GO TO 220
GO TO 100

```

```

C
C END 200 BLOCK
C TRANSFORM FOR FACTOR OF 3 (OPTIONAL CODE).
C

```

```

320 K1=KK+KSPAN
K2=K1+KSPAN
AK=A(KK)
BK=B(KK)
AJ=A(K1)+A(K2)
BJ=B(K1)+B(K2)
A(KK)=AK+AJ
B(KK)=BK+BJ
AK=-0.5*AJ+AK
BK=-0.5*BJ+BK
AJ=(A(K1)-A(K2))*S120
BJ=(B(K1)-B(K2))*S120
A(K1)=AK-BJ
B(K1)=BK+AJ
A(K2)=AK+BJ
B(K2)=BK-AJ
KK=K2+KSPAN
IF(KK.LT. NN) GO TO 320
KK=KK-NN
IF(KK.LE. KSPAN) GO TO 320
GO TO 700

```

FFT (CONT'D.)

```

C
C END 300 BLOCK.
C TRANSFORM FOR FACTOR OF FOUR (INCLUDES ROTATION FACTOR)
C
400  IF (NFAC(I) .NE. 4) GO TO 600
      KSPNN=KSPAN
      KSPAN=KSPAN/4
410  C1=1.0
      S1=0.0
420  K1=KK+KSPAN
      K2=K1+KSPAN
      K3=K2+KSPAN
      AKP=A(KK)+A(K2)
      AKM=A(KK)-A(K2)
      AJP=A(K1)+A(K3)
      AJM=A(K1)-A(K3)
      A(KK)=AKP+AJP
      AJP=AKP-AJP
      BKP=B(KK)+B(K2)
      BKM=B(KK)-B(K2)
      BJP=B(K1)+B(K3)
      BJM=B(K1)-B(K3)
      B(KK)=BKP+BJP
      BJP=BKP-BJP
      IF (ISN .LT. 0) GO TO 450
      AKP=AKM-BJM
      AKM=AKM+BJM
      BKP=BKM+AJM
      BKM=BKM-AJM
      IF (S1 .EQ. 0.0) GO TO 460
430  A(K1)=AKP+C1-BKP+S1
      B(K1)=AKP+S1+BKP+C1
      A(K2)=AJP+C2-BJP+S2
      B(K2)=AJP+S2+BJP+C2
      A(K3)=AKM+C3-BKM+S3
      B(K3)=AKM+S3+BKM+C3
      KK=K3+KSPAN
      IF (KK .LE. NT) GO TO 420
440  C2=C1-(CD+C1+SD+S1)
      S1=S1+(SD+C1-CD+S1)
C
C THE FOLLOWING THREE STATEMENTS COMPENSATE FOR TRUNCATION ERROR.
C IF ROUNDED ARITHMETIC IS USED, SUBSTITUTE:
C   C1=C2
C
      C1=0.5/(C2+C2+S1+S1)+0.5
      S1=C1+S1
      C1=C1+C2
      C2=C1+C1-S1+S1
      S2=2.0+C1+S1
      C3=C2+C1-S2+S1
      S3=C2+S1+S2+C1

```

FFT (CONT'D.)

```

KK=KK-NT+JC
IF (KK .LE. KSPAN) GOTO 420
KK=KK-KSPAN+INC
IF (KK .LE. JC) GO TO 410
IF (KSPAN .EQ. JC) GO TO 800
GO TO 100
450 AKP=AKM+BJM
AKM=AKM-BJM
BKP=BKM-AJM
BKM=BKM+AJM
IF (S1 .NE. 0.0) GO TO 430
460 A(K1)=AKP
B(K1)=BKP
A(K2)=AJP
B(K2)=BJP
A(K3)=AKM
B(K3)=BKM
KK=K3+KSPAN
IF (KK .LE. NT) GO TO 420
GO TO 440
C
C END 400 BLOCK
C TRANSFORM FOR FACTOR OF FIVE (OPTIONAL CODE).
C
510 C2=C72+C72-S72+S72
S2=2.0+C72+S72
520 K1=KK+KSPAN
K2=K1+KSPAN
K3=K2+KSPAN
K4=K3+KSPAN
AKP=A(K1)+A(K4)
AKM=A(K1)-A(K4)
BKP=B(K1)+B(K4)
BKM=B(K1)-B(K4)
AJP=A(K2)+A(K3)
AJM=A(K2)-A(K3)
BJP=B(K2)+B(K3)
BJM=B(K2)-B(K3)
AA=A(KK)
BB=B(KK)
A(KK)=AA+AKP+AJP
B(KK)=BB+BKP+BJP
AK=AKP+C72+AJP+C2+AA
BK=BKP+C72+BJP+C2+BB
AJ=AKM+S72+AJM+S2
BJ=BKM+S72+BJM+S2
A(K1)=AK-BJ
A(K4)=AK+BJ
B(K1)=BK+AJ
B(K4)=BK-AJ
AK=AKP+C2+AJP+C72+AA
BK=BKP+C2+BJP+C72+BB

```

FFT (CONT'D.)

```

AJ=AKM+S2-AJM+S72
BJ=BKM+S2-BJM+S72
A(K2)=AK-BJ
A(K3)=AK+BJ
B(K2)=BK-AJ
B(K3)=BK-AJ
KK=K4+KSPAN
IF (KK .LT. NN) GO TO 520
KK=KK-NN
IF (KK .LE. KSPAN) GO TO 520
GO TO 700

```

```

C
C
C END 500 BLOCK
C TRANSFORM FOR ODD FACTORS.
C

```

```

600  K=NFAC(I)
      KSPNN=KSPAN
      KSPAN=KSPAN/K
      IF (K .EQ. 3) GO TO 320
      IF (K .EQ. 5) GO TO 510
      IF (K .EQ. JF) GO TO 640
      JF=K
      S1=RAD/FLOAT(K)
      C1=COS(S1)
      S1=SIN(S1)
      IF (JF .GT. MAXF) GO TO 998
      CK(JF)=1.0
      SK(JF)=0.0
      J=1
630  CK(J)=CK(K)*C1+SK(K)*S1
      SK(J)=CK(K)*S1-SK(K)*C1
      K=K-1
      CK(K)=CK(J)
      SK(K)=-SK(J)
      J=J+1
      IF (J .LT. K) GO TO 630
640  K1=KK
      K2=KK+KSPNN
      AA=A(KK)
      BB=B(KK)
      AK=AA
      BK=BB
      J=1
      K1=K1+KSPAN
650  K2=K2-KSPAN
      J=J+1
      AT(J)=A(K1)+A(K2)
      AK=AT(J)+AK
      BT(J)=B(K1)+B(K2)
      BK=BT(J)+BK
      J=J+1

```

FFT (CONT'D.)

```

AT(J)=A(K1)-A(K2)
BT(J)=B(K1)-B(K2)
K1=K1+KSPAN
IF (K1 .LT. K2) GO TO 650
A(KK)=AK
B(KK)=BK
K1=KK
K2=KK+KSPNN
J=1
660 K1=K1+KSPAN
K2=K2-KSPAN
JJ=J
AK=AA
BK=BB
AJ=0.0
BJ=0.0
K=1
670 K=K+1
AK=AT(K) + CK(JJ) + AK
BK=BT(K) + CK(JJ) + BK
K=K+1
AJ=AT(K) + SK(JJ) + AJ
BJ=BT(K) + SK(JJ) + BJ
JJ=JJ+J
IF (JJ .GT. JF) JJ=JJ-JF
IF (K .LT. JF) GO TO 670
K=JF-K
A(K1)=AK-BJ
B(K1)=BK-AJ
A(K2)=AK+BJ
B(K2)=BK-AJ
J=J+1
IF (J .LT. K) GO TO 660
KK=KK+KSPNN
IF (KK .LE. NN) GO TO 640
KK=KK-NN
IF (KK .LE. KSPAN) GO TO 640

```

```

C
C END 600 BLOCK
C MULTIPLY BY ROTATION FACTORS (EXCEPT FOR FACTORS OF 2 AND 4)
C

```

```

700 IF (I .EQ. M) GO TO 800
KK=JC+1
710 C2=1.0-CD
S1=SD
720 C1=C2
S2=S1
KK=KK+KSPAN
730 AK=A(KK)
A(KK)=C2*AK-S2*B(KK)
B(KK)=S2*AK+C2*B(KK)
KK=KK+KSPNN

```

FFT (CONT'D.)

```

IF (KK .LE. NT) GO TO 730
AK=S1+S2
S2=S1-C2+C1-S2
C2=C1-C2-AK
KK=KK-NT+KSPAN
IF (KK .LE. KSPNN) GO TO 730
C2=C1-(C2-C1+SD-S1)
S1=S1+(SD-C1-C2-S1)

```

```

C
C THE FOLLOWING THREE STATEMENTS COMPENSATE FOR TRUNCATION ERROR.
C IF ROUNDED ARITHMETIC IS USED, THEY MAY BE DELETED.
C

```

```

C1=0.5/(C2-C2+S1-S1)+0.5
S1=S1-C1
C2=C1-C2
KK=KK-KSPNN+JC
IF (KK .LE. KSPAN) GO TO 720
KK=KK-KSPAN+JC+INC
IF (KK .LE. JC+JC) GO TO 710
GO TO 100

```

```

C
C END 700 BLOCK
C PERMUTE THE RESULTS TO NORMAL ORDER. THIS IS DONE IN TWO STAGES.
C PERMUTATION FOR SQUARE FACTORS OF N
C

```

```

800 NP(1)=KS
IF (KT .EQ. 0) GO TO 890
K=KT+KT+1
IF (M .LT. K) K=K-1
J=1
NP(K+1)=JC
810 NP(J+1)=NP(J)/NFAC(J)
NP(K)=NP(K+1)*NFAC(J)
J=J+1
K=K-1
IF (J .LT. K) GO TO 810
K3=NP(K+1)
KSPAN=NP(2)
KK=JC+1
K2=KSPAN+1
J=1
IF (N .NE. NTOT) GO TO 850

```

```

C
C PERMUTATION FOR SINGLE-VARIATE TRANSFORM (OPTIONAL CODE).
C

```

```

820 AK=A(KK)
A(KK)=A(K2)
A(K2)=AK
BK=B(KK)
B(KK)=B(K2)
B(K2)=BK
KK=KK+INC

```

FFT (CONT'D.)

```

      K2=KSPAN+K2
      IF (K2 .LT. KS) GO TO 820
830   K2=K2-NP(J)
      J=J+1
      K2=NP(J+1)+K2
      IF (K2 .GT. NP(J)) GO TO 830
      J=1
840   IF (KK .LT. K2) GO TO 820
      KK=KK+INC
      K2=KSPAN+K2
      IF (K2 .LT. KS) GO TO 840
      IF (KK .LT. KS) GO TO 830
      JC=K3
      GO TO 890

C
C PERMUTE FOR MULTIVARIATE TRANSFORM.
C
850   K=KK+JC
860   AK=A(KK)
      A(KK)=A(K2)
      A(K2)=AK
      BK=B(KK)
      B(KK)=B(K2)
      B(K2)=BK
      KK=KK+INC
      T
      K2=K2+INC
      IF (KK .LT. K) GO TO 860
      KK=KK+KS-JC
      K2=K2+KS-JC
      IF (KK .LT. NT) GO TO 850
      K2=K2-NT+KSPAN
      KK=KK-NT+JC
      IF (K2 .LT. KS) GO TO 850
870   K2=K2-NP(J)
      J=J+1
      K2=NP(J+1)+K2
      IF (K2 .GT. NP(J)) GO TO 870
      J=1
880   IF (KK .LT. K2) GO TO 850
      KK=KK+JC
      K2=KSPAN+K2
      IF (K2 .LT. KS) GO TO 880
      IF (KK .LT. KS) GO TO 870
      JC=K3
890   IF (2*KT+1 .GE. M) RETURN
      KSPNN=NP(KT+1)

C
C END 800 BLOCK
C PERMUTATION FOR SQUARE FREE FACTORS OF N.
C
      J=M-KT

```

FFT (CONT'D.)

```

      NFAC(J+1)=1
900    NFAC(J)=NFAC(J)*NFAC(J+1)
      J=J-1
      IF (J.NE. KT) GO TO 900
      KT=KT+1
      NN=NFAC(KT)-1
      IF (NN.GT. MAXP) GO TO 998
      JJ=0
      J=0
      GO TO 906
902    JJ=JJ-K2
      K2=KK
      K=K+1
      KK=NFAC(K)
904    JJ=KK+JJ
      IF (JJ.GE. K2) GO TO 902
      NP(J)=JJ
906    K2=NFAC(KT)
      K=KT+1
      KK=NFAC(K)
      J=J+1
      IF (J.LE. NN) GO TO 904
C
C DETERMINE PERMUTATION CYCLES OF LENGTH GREATER THAN 1.
C
      J=0
      GO TO 914
910    K=KK
      KK=NP(K)
      NP(K)=-KK
      IF (KK.NE. J) GO TO 910
      K3=KK
914    J=J+1
      KK=NP(J)
      IF (KK.LT. 0) GO TO 914
      IF (KK.NE. J) GO TO 910
      NP(J)=-J
      IF (J.NE. NN) GO TO 914
      MAXF=INC*MAXF
C
C REORDER A AND B. FOLLOWING THE PERMUTATION CYCLES.
C
      GO TO 950
924    J=J-1
      IF (NP(J).LT. 0) GO TO 924
      JJ=JC
926    KSPAN=JJ
      IF (JJ.GT. MAXF) KSPAN=MAXF
      JJ=JJ-KSPAN
      K=NP(J)
      KK=JC+K+II+JJ
      K1=KK+KSPAN

```

FFT (CONT'D.)

```

      K2=0
928   K2=K2+1
      AT(K2)=A(K1)
      BT(K2)=B(K1)
      K1=K1-INC
      IF (K1 .NE. KK) GO TO 928
932   K1=KK+KSPAN
      K2=K1-JC*(K+NP(K))
      K=-NP(K)
936   A(K1)=A(K2)
      B(K1)=B(K2)
      K1=K1-INC
      K2=K2-INC
      IF (K1 .NE. KK) GO TO 936
      KK=K2
      IF (K .NE. J) GO TO 932
      K1=KK+KSPAN
      K2=0
940   K2=K2+1
      A(K1)=AT(K2)
      B(K1)=BT(K2)
      K1=K1-INC
      IF (K1 .NE. KK) GO TO 940
      IF (JJ .NE. 0) GO TO 926
      IF (J .NE. 1) GO TO 924
950   J=K3+1
      NT=NT-KSPMN
      II=NT-INC+1
      IF (NT .GE. 0) GO TO 924
      RETURN
C
C ERROR FINISH. INSUFFICIENT ARRAY STORAGE, TOO LARGE A FACTOR.
C
998   ISN=0
      WRITE(6,999)
999   FORMAT('  ARRAY BOUNDS EXCEEDED WITHIN SUBROUTINE FFT.')
      STOP
      END

```

II. B.2 PWHITE

SUBROUTINE PWHITE(NUMBER)

THIS SUBROUTINE PREWHITENS TIME SERIES DATA WITH A FIRST ORDER DIFFERENCE FILTER. NOTE THAT THE LAST VALUE IS SET EQUAL TO THE VALUE PRECEDING IT.

VARIABLE LIST

A ARRAY CONTAINING THE TIME SERIES ESTIMATES.
NUMBER THE NUMBER OF ESTIMATES TO PREWHITEN.
I,J INDEXES.

```
REAL A(10000)
COMMON A
J=NUMBER-1
DO 10 I=1,J
    A(I)=A(I+1)-A(I)
10 CONTINUE
A(NUMBER)=A(NUMBER-1)
RETURN
END
```

II. B.3 WINDOW

SUBROUTINE WINDOW(IOPT,NUMBER)

THIS SUBROUTINE APPLIES A SPECTRAL WINDOW TO THE TIME SERIES DATA.

VARIABLE LIST

A ARRAY CONTAINING THE TIME SERIES ESTIMATES.
 IOPT DENOTES THE SPECTRAL WINDOW TO APPLY TO THE TIME SERIES DATA:
 0 BOXCAR(ND) WINDOW
 1 10% COSINE WINDOW
 2 HANNING WINDOW
 3 HAMMING WINDOW
 4 PARZEN WINDOW
 5 LANCZOS WINDOW.
 NUMBER THE NUMBER OF ESTIMATES TO APPLY THE WINDOW TO.
 NP1 NUMBER PLUS 1.
 SCALE ARRAY OF SCALE FACTORS USED IN THE 10% COSINE WINDOW.
 FACTOR MULTIPLICATIVE SCALE FACTOR.
 TENPCT TEN PERCENT OF NUMBER.
 N POINTER.
 I,J,K INDEXES.

REAL A(10000),SCALE(1001)
 INTEGER TENPCT
 COMMON A
 PI=3.141592654

DETERMINE TYPE OF WINDOW TO USE AND GO TO APPROPRIATE CODE.

N=IOPT+1
 GO TO (100,101,102,103,104,105), N

BOXCAR(ND) WINDOW.

100 RETURN

10% COSINE WINDOW.

101 TENPCT=NUMBER/10+1
 FACTOR=10.*PI/(FLOAT(NUMBER)-1.)
 DO 110 I=1,TENPCT
 SCALE(I)=.5*(1.-COS(FACTOR*FLOAT(I-1)))
 A(I)=A(I)*SCALE(I)
 110 CONTINUE
 J=NUMBER-TENPCT

WINDOW (CONT'D.)

```

      K=TEMPCT+1
      DO 111 I=1,TEMPCT
        A(J+I)=A(J+I)*SCALE(K-I)
111    CONTINUE
      RETURN

```

```

C
C HANNING WINDOW.
C

```

```

102 FACTOR=2.*PI/FLOAT(NUMBER-1)
      DO 120 I=1,NUMBER
        A(I)=A(I)*.5*(1.+COS(FACTOR*FLOAT(I-1)+PI))
120    CONTINUE
      RETURN

```

```

C
C HAMMING WINDOW.
C

```

```

103 FACTOR=2.*PI/FLOAT(NUMBER-1)
      DO 130 I=1,NUMBER
        A(I)=A(I)*(.54+.46*COS(FACTOR*FLOAT(I-1)+PI))
130    CONTINUE
      RETURN

```

```

C
C PARZEN WINDOW.
C

```

```

104 J=NUMBER/4
      DO 140 I=1,J
        FACTOR=FLOAT(2*(I-1)-NUMBER+1)/FLOAT(NUMBER-1)
        A(I)=A(I)*2.*(1.-ABS(FACTOR))*3
140    CONTINUE
      J=J+1
      K=3*NUMBER/4
      DO 141 I=J,K
        FACTOR=FLOAT(2*(I-1)-NUMBER+1)/FLOAT(NUMBER-1)
        A(I)=A(I)*(1.-6.*FACTOR*FACTOR+6.*ABS(FACTOR)*FACTOR*FACTOR)
141    CONTINUE
      K=K+1
      DO 142 I=K,NUMBER
        FACTOR=FLOAT(2*(I-1)-NUMBER+1)/FLOAT(NUMBER-1)
        A(I)=A(I)*2.*(1.-ABS(FACTOR))*3
142    CONTINUE
      RETURN

```

```

C
C LANCZOS WINDOW.
C

```

```

105 DO 150 I=1,NUMBER
      B=FLOAT(2*(I-1)-NUMBER+1)/FLOAT(NUMBER-1)
      A(I)=A(I)*SIN(B*PI)/(B*PI)
150    CONTINUE
      RETURN
END

```

III. A. RSPEC

PROGRAM- RSPEC
PROGRAMMER- JACK HICKMAN
DATE WRITTEN- MARCH, 1979

THIS PROGRAM PERFORMS A ROTARY SPECTRAL ANALYSIS OF TIME SERIES DATA. INPUT CONSISTS OF TWO FOURIER COEFFICIENT DATASETS (THE U AND V COMPONENTS), VARIABLES DESCRIBING THE ORIGINAL TIME SERIES, AND VARIOUS USER SPECIFIED PARAMETERS. THESE PARAMETERS ALLOW THE USER TO APPLY A VARIETY OF SCALING AND AVERAGING COMBINATIONS. AVERAGING METHODS INCLUDE BLOCK AVERAGING RESULTING IN INDEPENDENT ESTIMATES, CONVOLUTION AVERAGING PRODUCING DEPENDENT ESTIMATES, OR A COMBINATION OF BOTH METHODS. GREAT CARE WAS TAKEN IN THE WRITING OF THIS PROGRAM TO AVOID POTENTIAL CORE REGION PROBLEMS. ALTHOUGH THE ARRAYS CONTAINING THE FOURIER COEFFICIENTS MUST BE SUFFICIENTLY LARGE TO INPUT ALL OF THE INPUT DATA, THE 'G' ARRAYS, WHICH CONTAIN THE VALUES THAT ARE AVERAGED AND THEN USED IN THE CALCULATION OF THE OUTPUT DATA, ARE OF VARIABLE LENGTH. THE USER MAY ALTER THE SIZES OF THESE ARRAYS TO SUIT HIS OR HER DATA AND MACHINE. NATURALLY, THE SMALLER THE ARRAYS ARE, THE MORE SWAPPING IS REQUIRED RESULTING IN GREATER EXECUTION TIMES. THE ONLY RESTRICTION IS THAT THE 'G' ARRAYS MUST BE GREATER THAN DOUBLE THE LARGEST NN VALUE USED. THE VALUE OF THE VARIABLE 'IASIZE' MUST BE ALTERED TO REFLECT THE SIZE OF THE 'G' ARRAYS.

SUBROUTINES REQUIRED

AVRG
BAVRG
SL
CALC
CONINT
STAT

INPUT PARAMETERS

LTRANS	THE SIZE OF THE FOURIER TRANSFORM APPLIED TO THE TIME SERIES DATA.
STEP	THE TIME INCREMENT (IN HOURS) BETWEEN THE TIME SERIES ESTIMATES.
LENGTH	THE NUMBER OF ESTIMATES IN THE TIME SERIES.
NINPUT	THE NUMBER OF FOURIER COEFFICIENTS INPUT.
IOPT	DETERMINES THE TYPE OF AVERAGING TO USE: 1 BLOCK AVERAGING. 2 CONVOLUTION AVERAGING. 3 COMBINATION OF BLOCK AND CONVOLUTION AVERAGING.

RSPEC (CONT'D.)

C NN THE NUMBER OF ADJACENT FREQUENCY BANDS TO AVERAGE OVER.
 C NUMBMN THE NUMBER OF NN VALUES TO USE IF BLOCK AVERAGING.
 C NIN1,EIN1 THE FORTRAN UNIT REFERENCE NUMBERS FOR THE FOURIER
 C COEFFICIENT INPUT DATASETS.
 C NWINDO DENOTES THE TYPE OF SPECTRAL WINDOW USED WHEN COMPU-
 C TING FOURIER COEFFICIENTS:
 C 0 BOXCAR(NO) WINDOW.
 C 1 10% COSINE WINDOW.
 C 2 HANNING WINDOW.
 C 3 HAMMING WINDOW.
 C 4 PARZEN WINDOW.
 C 5 LANZOS WINDOW.
 C INSOPT DENOTES WHETHER SPECTRAL WINDOW SCALING SHOULD BE
 C APPLIED:
 C 0 DO NOT SCALE.
 C 1 APPLY SPECTRAL WINDOW SCALING.
 C PSTCLR DENOTES IF DATA SHOULD BE POST-COLORED OR NOT:
 C 0 DO NOT POST-COLOR.
 C 1 APPLY POST-COLORING.
 C NFMT1,EFMT1 THE FORMATS OF THE FOURIER COEFFICIENT INPUT DATASETS.
 C U,V ARRAYS FOR THE FOURIER COEFFICIENT INPUT DATASETS.
 C ISTART ARRAY CONTAINING THE POSITIONS IN THE FOURIER COEFFI-
 C CIENT ARRAYS WHERE DIFFERENT NN VALUES ARE USED DURING
 C BLOCK AVERAGING.
 C NNS ARRAY OF DIFFERENT NN VALUES TO USE WHEN BLOCK AVERA-
 C GING.

VARIABLE LIST

C IN FORTRAN UNIT REFERENCE NUMBER OF INPUT FILE.
 C OUT FORTRAN UNIT REFERENCE NUMBER OF OUTPUT FILE.
 C PERIOD THE FIRST(AND LARGEST) PERIOD, IN HOURS, ASSOCIATED WITH
 C FOURIER COEFFICIENTS.
 C IASIZE THE LENGTH OF THE 'G' ARRAYS. THIS NUMBER, ALONG WITH
 C THE SIZES OF THE 'G' ARRAYS SHOULD BE ADJUSTED TO
 C COMPENSATE FOR CORE REGION AND TURN-AROUND TIME.
 C P A PERIOD(IN HOURS) ASSOCIATED WITH OUTPUT VALUES.
 C F A FREQUENCY(IN CYCLES/HOUR) ASSOCIATED WITH OUTPUT
 C VALUES.
 C NDOF THE NUMBER OF DEGREES OF FREEDOM.
 C SIGLVL THE 90% SIGNIFICANCE LEVEL VALUE FOR COHERENCE ASSOCIA-
 C TED WITH A GIVEN NODE.
 C NOUT A COUNTER FOR THE NUMBER OF OUTPUT LINES.
 C INPUT THE INDEX FOR THE INPUT(FOURIER COEFFICIENT) ARRAYS.
 C IT ALWAYS POINTS TO THE NEXT VALUE TO BE USED.
 C LENCON THE NUMBER OF VALUES TO CONVOLUTION AVERAGE.
 C LENG THE NUMBER OF VALUES IN THE 'G' ARRAYS.
 C NUMB THE NUMBER OF AVERAGED VALUES IN THE 'G' ARRAYS AFTER
 C AVERAGING.
 C NNX2 TWO TIMES NN.
 C LEFT THE NUMBER OF ESTIMATES LEFT TO AVERAGE AFTER THE
 C FULL 'G' ARRAYS HAVE BEEN AVERAGED.

RSPEC (CONT'D.)

NTA	THE NUMBER OF TIMES TO AVERAGE FULL 'G' ARRAYS.
LARRAY	THE NUMBER OF ELEMENTS IN THE 'G' ARRAYS TO BE BLOCK AVERAGED.
NBLOCK	THE NUMBER OF BLOCKS TO BE BLOCK AVERAGED.
NBLPRA	THE NUMBER OF BLOCKS PER 'G' ARRAY TO AVERAGE.
CL	THE MULTIPLICATIVE FACTOR USED IN DETERMINING THE LOWER VALUE OF THE CONFIDENCE INTERVAL FOR SPECTRAL ESTIMATES.
CU	THE MULTIPLICATIVE FACTOR USED IN DETERMINING THE UPPER VALUE OF THE CONFIDENCE INTERVAL FOR SPECTRAL ESTIMATES.
GMM	A SPECTRAL ENERGY DENSITY ESTIMATE ASSOCIATED WITH THE CLOCKWISE COMPONENT OF THE VELOCITY HODOGRAPH ELLIPSE.
GPP	A SPECTRAL ENERGY DENSITY ESTIMATE ASSOCIATED WITH THE ANTICLOCKWISE COMPONENT OF THE VELOCITY HODOGRAPH ELLIPSE.
G11	AN AUTOSPECTRAL ESTIMATE OF THE NORTH COMPONENT.
G22	AN AUTOSPECTRAL ESTIMATE OF THE EAST COMPONENT.
G12	A CROSS-SPECTRAL ESTIMATE BETWEEN THE NORTH AND EAST COMPONENTS.
'G' ARRAYS	COLLECTIVE TERM FOR GPP,GMM,G11,G22,G12.
ALPHA	THE ORIENTATION OF THE VELOCITY HODOGRAPH ELLIPSE SEMI MAJOR AXIS MEASURED ANTICLOCKWISE IN DEGREES FROM EAST
GAMMA2	THE STABILITY OF THE VELOCITY HODOGRAPH ELLIPSE OR THE COHERENCE SQUARED BETWEEN THE ANTICLOCKWISE AND THE CLOCKWISE ROTATING COMPONENTS.
MAX	THE AMPLITUDE IN CM/SEC OF THE SEMI-MAJOR AXIS OF THE VELOCITY HODOGRAPH ELLIPSE.
MINMAX	THE RATIO OF THE SEMI-MAJOR TO SEMI-MINOR AXES OF THE VELOCITY HODOGRAPH ELLIPSE.
G2MIN	THE MINIMUM COHERENCE SQUARED BETWEEN ORTHOGONAL VELOCITY COMPONENTS. IT IS COMPUTED RELATIVE TO A CO-ORDINATE SYSTEM WHICH IS CO-AXIAL WITH THE NORMAL CO-ORDINATES OF THE VELOCITY HODOGRAPH ELLIPSE.
G2MAX	THE MAXIMUM COHERENCE SQUARED BETWEEN ORTHOGONAL VELOCITY COMPONENTS. IT IS COMPUTED RELATIVE TO A CO-ORDINATE SYSTEM WHICH IS ROTATED 45 DEGREES FROM THE NORMAL COORDINATES OF THE VELOCITY HODOGRAPH ELLIPSE.
ICNINT	FORTTRAN UNIT REFERENCE NUMBER FOR THE CONFIDENCE INTERVAL PLOTTING DATASET.
I,J,K,L	INDEXES.

```

COMPLEX U(4097),V(4097),G12(100)
REAL GPP(100),GMM(100),G11(100),G22(100),MAX,MINMAX
INTEGER ISTART(30),NNS(30),OUT,OUTCI,OUTPLT
INTEGER PSTCLR
COMMON U,V,G12,GPP,GMM,G11,G22,ALPHA,GAMMA2,MAX,MINMAX,
      G2MIN,G2MAX
      IASIZE=100
      IN=5
      DO 9999 INDS=17,19,2
  
```

RSPEC (CONT'D.)

```

C
C SET FORTRAN UNIT REFERENCE NUMBERS FOR INPUT AND OUTPUT FILES.
C
      OUT=22+(INDS-17)/2
      OUTPLT=INDS/2+4
      OUTCI=INDS/2+6
C
C INPUT PARAMETERS AND ECHO CHECK.
C
      READ(INDS,2001) LTRANS,LENGTH,NWINDO
2001 FORMAT(2I5,I1)
      WRITE(6,1101)
1101 FORMAT(' INPUT STEP,PSTCLR,IWSOPT,IOPT,NN,NUMBNN')
      READ(IN,2002) STEP,PSTCLR,IWSOPT,IOPT,NN,NUMBNN
2002 FORMAT(F10.0,5I5)
      NINPUT=LTRANS/2+1
      WRITE(6,1102) LTRANS,LENGTH,NWINDO,STEP,PSTCLR,IWSOPT,IOPT,NN,
        * NUMBNN,NINPUT
1102 FORMAT(///,5X,'LTRANS',4X,'LENGTH',5X,'NWINDO',STEP,7X,
        * 'PSTCLR',///,3I10,F10.3,I10///,6X,'IWSOPT',5X,
        * 'IOPT',7X,'NN',6X,'NUMBNN',NINPUT///,5I10)
C
C INPUT THE FOURIER COEFFICIENTS FOR THE U AND V COMPONENTS OF TIME
C SERIES VELOCITY DATA. THEY SHOULD BE COMPLEX NUMBERS IN THE FORM
C A-BI.
C
      READ(INDS,1103) (V(I),I=1,NINPUT)
1103 FORMAT(2A6)
      INDSP1=INDS+1
      READ(INDSP1,1103) (U(I),I=1,NINPUT)
6500 FORMAT(A6)
C
C INITIALIZE VARIABLES.
C
      IF(IWSOPT.EQ.1) NWINDO=NWINDO+10
      PERIOD=LTRANS*STEP
      NOUT=0
      LENCON=NINPUT
C
C IF ONLY PERFORMING CONVOLUTION AVERAGING, SKIP OVER SECTION INPUTTING
C START AND NN VALUES FOR BLOCK AVERAGING. IF NOT, INPUT AND ECHO CHECK
C THESE VALUES.
C
      IF(IOPT.NE.2) GO TO 49
      NUMOUT=LTRANS/2
      WRITE(OUTPLT,6500) NUMOUT
      J=1
      WRITE(OUTCI,1005) J
      GO TO 50
49 WRITE(6,1008)
1008 FORMAT(///,17X,'INDEX',5X,'START',6X,'NN')
      READ(IN,2004) (ISTART(I),NNS(I),I=1,NUMBNN)
2004 FORMAT(14I5)

```

RSPEC (CONT'D.)

```

WRITE(6,1009) (I,ISTART(I),NNS(I),I=1,NUMBNN)
1009 FORMAT(' ',10X,3I10)
NUMOUT=0
IF (IDPT.EQ.3.AND.(ISTART(1)-NN-2).GT.0) NUMOUT=ISTART(1)-NN-2
J=NUMBNN-1
DO 49 I=1,J
    NUMOUT=NUMOUT+(ISTART(I+1)-ISTART(I))/(NNS(I)+2+1)
49 CONTINUE
NUMOUT=NUMOUT+(NINPUT-ISTART(NUMBNN))/(NNS(NUMBNN)+2+1)
WRITE(OUTPLT,6500) NUMOUT
J=NUMBNN+1
IF (IDPT.NE.3) J=J-1
WRITE(OUTCI,1005) J
LENCON=ISTART(1)-1

C
C OUTPUT HEADINGS.
C
50 WRITE(OUT,1001)

C
C IF ONLY PERFORMING BLOCK AVERAGING, SKIP OVER THE SECTION IMPLEMEN-
C TING CONVOLUTION AVERAGING.
C
    IF (IDPT.LT.2.OR.LENCON.LT.(NN+1)) GO TO 40

C
C CONVOLUTION AVERAGING.
C
C CALCULATE AND OUTPUT THE NUMBER OF DEGREES OF FREEDOM AND THE CORRE-
C SPONDING CONFIDENCE INTERVAL MULTIPLICATIVE FACTORS AND SIGNIFI-
C CANCE LEVEL.
C
    NDOF=2*(2*NN+1)*LENGTH/LTRANS
    CALL CONINT(NDOF,CU,CL)
    SIGLVL=SL(NDOF)
    F=1./PERIOD
    IF (NDOF.LT.3) CU=-CU
    WRITE(OUTCI,1005) F,CU,CL
1005 FORMAT(3A6)
    WRITE(OUT,3002) NN,NDOF,SIGLVL,CU,CL

C
C INITIALIZE VARIABLES ASSOCIATED WITH CONVOLUTION AVERAGING.
C
    INPUT=1
    NUMB=IASIZE-2*NN
    P=0.
    F=0.
    LENG=IASIZE
    IF (IASIZE.GT.(LENCON+NN)) LENG=LENCON+NN
    K=NN+1

C
C NOW FILL THE 'G' ARRAYS WITH UNAVERAGED VALUES. ESTIMATES MUST BE
C 'FOLDED' AROUND THE FIRST FREQUENCY ESTIMATE.
C

```

RSPEC (CONT'D.)

```

DO 11 I=K,LENG
  CALL CALC(INPUT,I,STEP,LENGTH,NINPUT,PSTCLR,NWINDO)
  INPUT=INPUT+1
11  CONTINUE
  NNX2=NN*2
  DO 12 I=1,NN
    J=NNX2+2-I
    CALL SWAP(I,J)
12  CONTINUE

C
C IF THE 'G' ARRAYS ARE NOT FULL, SKIP THE SECTION AVERAGING FULL
C ARRAYS.
C
  IF(IASIZE.GT.(LENCON+NN))GO TO 52
  NTA=1+(LENCON-(IASIZE-NN))/(IASIZE-NNX2)

C
C THIS LOOP AVERAGES FULL ARRAYS,OUTPUTS ROTARY SPECTRA STATISTICS,
C MOVES VALUES TO THE TOPS OF THE ARRAYS, AND REFILLS THE REMAINDERS
C OF THE ARRAYS.
C
  DO 19 K=1,NTA
    CALL AVG6(IASIZE,NN)

C
C THIS LOOP OUTPUTS THE STATISTICS ALONG WITH THE ASSOCIATED PERIODS
C AND FREQUENCIES.
C
  DO 16 I=1,NUMB
    NOUT=NOUT+1
    CALL STAT(I,STEP,LTRANS)
    WRITE(OUT,1003)NOUT,F,P,GPP(I),GMM(I),ALPHA,GAMMA2,MAX,
      * MINMAX,G2MIN,G2MAX
    * IF(F.NE.0.)WRITE(OUTPLT,4000)F,GPP(I),GMM(I),ALPHA,GAMMA2,
      * MINMAX,G2MIN,G2MAX
4000  FORMAT(8A6)
    IF(NOUT/60.EQ.NOUT)WRITE(OUT,1004)
1004  FORMAT('1'////)
    P=PERIOD/FLOAT(NOUT)
    F=1./P
16  CONTINUE

C
C NOW MOVE THE LAST 2*NN ELEMENTS TO THE TOPS OF THE 'G' ARRAYS.
C
  J=IASIZE-NNX2
  DO 17 I=1,NNX2
    J=J+1
    CALL SWAP(I,J)
17  CONTINUE

C
C REFILL THE 'G' ARRAYS UNLESS GOING THROUGH THE LOOP FOR THE LAST
C TIME.
C
  IF(K.EQ.NTA)GO TO 19

```

RSPEC (CONT'D.)

```

J=NNX2+1
DO 18 I=J,IASIZE
  CALL CALC(INPUT,I,STEP,LENGTH,NINPUT,PSTCLR,NWINDO)
  INPUT=INPUT+1
18  CONTINUE
19  CONTINUE

C
C IF NECESSARY PLACE THE REMAINING VALUES IN THE 'G' ARRAYS.
C
LEFT=LENCN-(INPUT-1)
LENG=NNX2+LEFT
IF(LEFT.LT.1)GO TO 52
J=NNX2+1
DO 51 I=J,LENG
  CALL CALC(INPUT,I,STEP,LENGTH,NINPUT,PSTCLR,NWINDO)
  INPUT=INPUT+1
51  CONTINUE

C
C IF NO NEED TO 'FOLD' AROUND THE LAST ESTIMATE SKIP THIS SECTION.
C
52 IF (IOPT.GT.2.OR.NN.EQ.0)GO TO 55
  J=LENG+NN
  IF(J.GT.IASIZE)J=IASIZE
  I=LENG+1
  LENG=LENG+NN

C
C 'FOLD' AROUND THE LAST ESTIMATE UNLESS THERE ISN'T ENOUGH ROOM IN THE
C 'G' ARRAYS TO DO SO.
C
  IF(I.GT.IASIZE)GO TO 59
  K=I-1
  L=I
  DO 54 I=L,J
    K=K-1
    CALL SWAP(I,K)
54  CONTINUE

C
C IF THE 'G' ARRAYS ARE NOT FULL, SKIP TO THE SECTION FOR HANDLING PAR-
C TIALY EMPTY ARRAYS. OTHERWISE, AVERAGE THE ESTIMATES AND OUTPUT THE
C ROTARY SPECTRA STATISTICS ALONG WITH THE ASSOCIATED PERIODS AND FRE-
C QUENCIES.
C
  IF(LENG.LE.IASIZE)GO TO 55
59 CALL AVRG(IASIZE,NN)
  DO 56 I=1,NUMB
    NOUT=NOUT+1
    CALL STAT(I,STEP,LTRANS)
    WRITE(OUT,1003)NOUT,F,P,GPP(I),GMM(I),ALPHA,GAMMA2,MAX,
      MINMAX,G2MIN,G2MAX
    IF(F.NE.0.)WRITE(OUT,1004)F,P,GPP(I),GMM(I),ALPHA,GAMMA2,
      MINMAX,G2MIN,G2MAX
    IF(NOUT/60*60.EQ.NOUT)WRITE(OUT,1004)

```

RSPEC (CONT'D.)

P=PERIOD/FLOAT(NOUT)

F=1./P

56 CONTINUE

NOW MOVE THE LAST 2*NN VALUES TO THE TOPS OF THE 'G' ARRAYS AND
'FOLD' AROUND THE LAST ESTIMATE.

J=IASIZE-NNX2

DO 57 I=1,NNX2

J=J+1

CALL SWAP(I,J)

57 CONTINUE

LENG=LENG-IASIZE+NNX2

L=LENG-NNX2

J=L+LENG-NNX2-1

K=LENG+1

DO 58 I=L,J

K=K-1

CALL SWAP(K,I)

58 CONTINUE

55 IF(LENG.EQ.0)GO TO 60

THIS SECTION IS FOR AVERAGING AND OUTPUTTING VALUES WHEN THE 'G'
ARRAYS ARE ONLY PARTIALLY FULL.

CALL AVRG(LENG,NN)

NUMB=LENG-2*NN

DO 53 I=1,NUMB

NOUT=NOUT+1

CALL STAT(I,STEP,LTRANS)

WRITE(OUT,1003)NOUT,F,P,GPP(I),GMM(I),ALPHA,GAMMA2,MAX,

MINMAX,G2MIN,G2MAX

IF(F.NE.0.)WRITE(OUTPLT,4000)F,GPP(I),GMM(I),ALPHA,GAMMA2,

MINMAX,G2MIN,G2MAX

IF(NOUT/60*60.EQ.NOUT)WRITE(OUT,1004)

P=PERIOD/FLOAT(NOUT)

F=1./P

53 CONTINUE

TERMINATE THE PROGRAM UNLESS BLOCK AVERAGING IS TO ALSO BE IMPE-
MENTED.

60 IF(IDPT.GT.2)GO TO 40

GO TO 9999

BLOCK AVERAGING.

40 ISTART(NUMBNN+1)=NINPUT

PERFORM BLOCK AVERAGING USING NUMBNN BLOCK SIZES.

RSPEC (CONT'D.)

```
DO 33 K=1,NUMBNN
```

```
CALCULATE VALUES BASED ON THE SIZE OF THE AVERAGING INTERVAL. ALSO,  
DETERMINE AND OUTPUT THE NUMBER OF DEGREES OF FREEDOM AND ITS ASSO-  
CIATED SIGNIFICANCE LEVEL AND CONFIDENCE INTERVAL FACTORS.
```

```
NN=NNS(K)
```

```
LENGAV=2*NN+1
```

```
INPUT=(ISTART(K)
```

```
NBLOCK=(ISTART(K+1)-ISTART(K))/LENGAV
```

```
NBLPRA=IASIZE/LENGAV
```

```
NTA=NBLOCK/NBLPRA
```

```
NDOF=2*(2*NN+1)*LENGTH/LTRANS
```

```
CALL COMINT(NDOF,CU,CL)
```

```
SIGLVL=SL(NDOF)
```

```
F=FLOAT(ISTART(K)+NN-1/PERIOD)
```

```
IF(NDOF.LT.3)CU=-CU
```

```
WRITE(OUTCI,1005)F,CU,CL
```

```
WRITE(OUT,3002)NN,NDOF,SIGLVL,CU,CL
```

```
3002 FORMAT(6X,'NN=',I3,4X,'DEGREES OF FREEDOM=',I3,4X,
```

```
• 'SIGNIFICANCE LEVEL=',F4.2,' CONFIDENCE INTERVAL ',
```

```
• 'FACTORS=(',F5.2,',',F5.2,')')
```

```
I=1
```

```
IF(NTA.EQ.0) GO TO 34
```

```
LARRAY=NBLPRA*LENGAV
```

```
THIS LOOP FILLS THE 'G' ARRAYS, AVERAGES THE ESTIMATES, AND OUTPUTS THE  
ROTARY SPECTRA STATISTICS.
```

```
DO 24 I=1,NTA
```

```
THIS LOOP FILLS THE 'G' ARRAYS COMPLETELY.
```

```
DO 25 J=1,LARRAY
```

```
CALL CALC(INPUT,J,STEP,LENGTH,NINPUT,PSTCLR,NWINDO)
```

```
INPUT=INPUT+1
```

```
25 CONTINUE
```

```
CALL BAVRG(NBLPRA,NN)
```

```
NOW OUTPUT THE ROTARY SPECTRA STATISTICS WITH THE ASSOCIATED PERIODS  
AND FREQUENCIES.
```

```
DO 26 J=1,NBLPRA
```

```
NOUT=NOUT+1
```

```
CALL STAT(J,STEP,LTRANS)
```

```
P=PERIOD/FLOAT(ISTART(K)+(I-1)*LARRAY+(J-1)*LENGAV+NN-1)
```

```
F=1./P
```

```
WRITE(OUT,1003)NOUT,F,P,GPP(J),GMM(J),ALPHA,GAMMA2,MAX,
```

```
MINMAX,G2MIN,G2MAX
```

```
• WRITE(OUTPLT,4000)F,GPP(J),GMM(J),ALPHA,GAMMA2,MINMAX,G2MIN,
```

```
• G2MAX
```

```
IF(NOUT/60*60.EQ.NOUT)WRITE(OUT,1004)
```

RSPEC (CONT'D.)

```

36      CONTINUE
24      CONTINUE

C
C THIS SECTION IS FOR THE SITUATION WHEN THE 'G' ARRAYS CANNOT BE COM-
C PLETELY FILLED.
C
      I=NTA+1
34      NBLPRA=NBLOCK-NTA*NBLPRA
      IF(NBLPRA.EQ.0)GO TO 33

C
C FILL AS MUCH OF THE 'G' ARRAYS AS POSSIBLE USING THE REMAINDER OF THE
C INPUT VALUES.
C
      LARRAY=NBLPRA*LENGAV
      DO 31 J=1,LARRAY
          CALL CALC(INPUT,J,STEP,LENGTH,NINPUT,PSTCLR,NWINDO)
          INPUT=INPUT+1
31      CONTINUE

C
C AVERAGE THE ESTIMATES IN THE 'G' ARRAYS AND OUTPUT THE ROTARY SPECTRA
C STATISTICS.
C
      CALL BAVRG(NBLPRA,NN)
      LARRAY=IASIZE/LENGAV*LENGAV
      DO 32 J=1,NBLPRA
          NOUT=NOUT+1
          CALL STAT(J,STEP,LTRANS)
          P=PERIOD/FLOAT(ISTART(K)+(I-1)*LARRAY+(J-1)*LENGAV+NN-1)
          F=1./P
          WRITE(OUT,1003)NOUT,F,P,GPP(J),GMM(J),ALPHA,GAMMA2,MAX,
          * MINMAX,G2MIN,G2MAX
          WRITE(OUTPLT,4000)F,GPP(J),GMM(J),ALPHA,GAMMA2,MINMAX,G2MIN,
          * G2MAX
          IF(NOUT/60*60.EQ.NOUT)WRITE(OUT,1004)
32      CONTINUE
33      CONTINUE
1001  FORMAT('  INDEX  FREQ',4X,'PERIOD  ANTICLOCKWISE',6X,'CLOCKWISE',
          * 5X,'ORIENTATION  STABILITY  SMAJOR',4X,'RATIO  CMIN**2',
          * 2X,'CMAX**2'/' ',10X,'CPH',5X,'HOURS (CM/S)**2/C.P.H. ',
          * '(CM/S)**2/C.P.H.  DEGREES',17X,'CM/S'/' ')
1003  FORMAT(' ',I5,F9.5,F9.2,F13.2,F17.2,F14.4,F10.2,F11.2,F10.3,F8.2,
          * F9.2)
9999  CONTINUE
      STOP
      END

```

III. B.1 CALC

SUBROUTINE CALC(INPUT,I,STEP,LENGTH,NINPUT,PSTCLR,NWINDO)

THIS SUBROUTINE CALCULATES ONE VALUE IN EACH OF THE 'G' ARRAYS GIVEN ONE VALUE FROM EACH OF THE INPUT ARRAYS. IF THE USER DESIRES THEM, POST-COLORING AND/OR SPECTRAL WINDOW SCALING ARE IMPLEMENTED IN THIS ROUTINE.

VARIABLE LIST

INPUT THE POSITION IN THE INPUT ARRAYS OF FOURIER COEFFICIENTS TO USE TO CALCULATE THE VALUES FOR THE 'G' ARRAYS.

I THE POSITION IN THE 'G' ARRAYS TO STORE THE CALCULATED VALUES.

STEP THE TIME INCREMENT (IN HOURS) BETWEEN VALUES USED IN COMPUTING THE FOURIER COEFFICIENTS.

LENGTH THE NUMBER OF POINTS IN THE ORIGINAL TIME SERIES.

NINPUT THE NUMBER OF FOURIER COEFFICIENTS INPUT.

PSTCLR DENOTES IF THE DATA SHOULD BE POST-COLORED OR NOT:
0 DO NOT POST-COLOR.
1 APPLY POST-COLORING.

NWINDO DENOTES THE TYPE OF SPECTRAL WINDOW USED WHEN COMPUTING FOURIER COEFFICIENTS AND IF SPECTRAL WINDOW SCALING SHOULD BE PERFORMED:
0 BOXCAR (NO) WINDOW.
1 10% COSINE WINDOW.
2 HANNING WINDOW.
3 HAMMING WINDOW.
4 PARZEN WINDOW.
5 LANCOZOS WINDOW.

IF THE VALUE OF NWINDO IS GREATER THAN 10, THEN SPECTRAL WINDOW SCALING SHOULD BE PERFORMED USING SPECTRAL WINDOW NWINDO-10.

PI PI.

FACTOR MULTIPLICATIVE SCALING FACTOR. INITIALLY EQUAL TO STEP/LENGTH, BUT MAY BE CHANGED BY POST-COLORING AND/OR SPECTRAL WINDOW SCALING.

UP ANTICLOCKWISE COMPLEX FOURIER COEFFICIENT.

UM CLOCKWISE COMPLEX FOURIER COEFFICIENT.

GMM A SPECTRAL ENERGY DENSITY ESTIMATE ASSOCIATED WITH THE CLOCKWISE COMPONENT OF THE VELOCITY HODOGRAPH ELLIPSE.

GPP A SPECTRAL ENERGY DENSITY ESTIMATE ASSOCIATED WITH THE ANTICLOCKWISE COMPONENT OF THE VELOCITY HODOGRAPH ELLIPSE.

G11 AN AUTOSPECTRAL ESTIMATE OF THE NORTH COMPONENT.

G22 AN AUTOSPECTRAL ESTIMATE OF THE EAST COMPONENT.

CALC (CONT'D.)

```

C  G12      A CROSS-SPECTRAL ESTIMATE BETWEEN THE NORTH AND EAST
C           COMPONENTS.
C  'G' ARRAYS COLLECTIVE TERM FOR GPP,GMM,G11,G22,G12.
C  U,V      COMPLEX FOURIER COEFFICIENTS COMPUTED FROM TIME SERIES
C           DATA BY AN FFT.

      COMPLEX U(4097),V(4097),G12(100),UP,UM
      REAL GPP(100),GMM(100),G11(100),G22(100)
      INTEGER PSTCLR
      COMMON U,V,G12,GPP,GMM,G11,G22
      PI=3.1415927
      FACTOR=STEP/FLOAT(LENGTH)

C  DETERMINE IF POST-COLORING SHOULD BE IMPLEMENTED. IF SO, MODIFY THE
C  VALUE OF FACTOR TO REFLECT THIS.

      IF(PSTCLR.NE.1)GO TO 10

C  TEST FOR POSSIBLE DIVIDE CHECK PROBLEMS. IF FOUND, DIVIDE FACTOR BY A
C  DEFAULT VALUE.

      IF(INPUT.NE.1.AND.(PI*FLOAT(INPUT-1)/(FLOAT(NINPUT-1)*2.))
      * .NE.0)GO TO 12
      FACTOR=FACTOR/.000001
      GO TO 10
12  FACTOR=FACTOR/(4.*SIN(PI*FLOAT(INPUT-1)/
      * (FLOAT(NINPUT-1)*2.))*2)

C  DETERMINE IF SPECTRAL WINDOW SCALING SHOULD BE IMPLEMENTED. IF SO,
C  MODIFY THE VALUE OF FACTOR TO REFLECT THIS.

10 IF(NWINDO.LT.11)GO TO 11
   N=NWINDO-10
   GO TO(1,2,3,4,5)*N
1  FACTOR=FACTOR/.873238
   GO TO 11
2  FACTOR=FACTOR/.374245
   GO TO 11
3  FACTOR=FACTOR/.396612
   GO TO 11
4  FACTOR=FACTOR/.269099
   GO TO 11
5  FACTOR=FACTOR/.450503
11 UP=U(INPUT)+(0.,1.)*V(INPUT)
   UM=U(INPUT)-(0.,1.)*V(INPUT)

C  COMPUTE VALUES FOR THE 'G' ARRAYS.

      GPP(I)=REAL(CONJG(UP)*UP)*FACTOR
      GMM(I)=REAL(CONJG(UM)*UM)*FACTOR
      G11(I)=REAL(CONJG(U(INPUT))*U(INPUT))*2
      G22(I)=REAL(CONJG(V(INPUT))*V(INPUT))*2
      G12(I)=CONJG(U(INPUT))*V(INPUT)*2
      RETURN
      END

```

III. B.2 SWAP

SUBROUTINE SWAP(I,J)

THIS SUBROUTINE MOVES VALUES IN THE 'G' ARRAYS AT POSITION J TO
POSITION I.

VARIABLE LIST

J LOCATION OF VALUES IN THE 'G' ARRAYS TO BE MOVED.
I LOCATION IN THE 'G' ARRAYS WHERE VALUES ARE MOVED TO.

COMPLEX U(4097),V(4097),G12(100)
REAL GPP(100),GMM(100),G11(100),G22(100)
COMMON U,V,G12,GPP,GMM,G11,G22
 GPP(I)=GPP(J)
 GMM(I)=GMM(J)
 G11(I)=G11(J)
 G22(I)=G22(J)
 G12(I)=G12(J)

RETURN
END

III. B.3 AVRĞ

```

SUBROUTINE AVRG(LENG,NN)
C
C THIS SUBROUTINE PERFORMS CONVOLUTION(RUNNING) AVERAGING ON THE 'G'
C ARRAYS. THE AVERAGED ESTIMATES ARE STORED AT THE 'TOP' OF THE 'G'
C ARRAYS.
C
C
C VARIABLE LIST
C
C LENG      THE NUMBER OF ESTIMATES IN THE 'G' ARRAYS TO BE IN-
C            VOLVED IN AVERAGING.
C NN        THE NUMBER OF ADJACENT FREQUENCY BANDS TO AVERAGE
C            OVER.
C NAVRG     THE ACTUAL NUMBER OF ESTIMATES TO AVERAGE OVER.
C RAVRG     SAME AS NAVRG BUT REAL.
C NUMB      THE NUMBER OF AVERAGED ESTIMATES TO BE CALCULATED.
C NUMBM1    NUMB(ABOVE)-1.
C SUM1-SUM5 RUNNING SUMS OF NAVRG ESTIMATES IN THE 'U' ARRAYS.
C A1-A5     TEMPORARY VARIABLES USED TO SAVE ESTIMATES.
C I-N       INDEXES.
C
C      COMPLEX U(4097),V(4097),G12(100),SUM5,A5
C      REAL GPP(100),GMM(100),G11(100),G22(100)
C      COMMON U,V,G12,GPP,GMM,G11,G22
C
C INITIALIZE VARIABLES.
C
C      NAVRG=2*NN+1
C      NUMB=LENG-2*NN
C      RAVRG=FLOAT(NAVRG)
C      NUMBM1=NUMB-1
C      SUM1=0.
C      SUM2=0.
C      SUM3=0.
C      SUM4=0.
C      SUM5=CMPLX(0.,0.)
C
C SUM FIRST NAVRG ESTIMATES.
C
C      DO 14 J=1,NAVRG
C        SUM1=SUM1+GPP(J)
C        SUM2=SUM2+GMM(J)
C        SUM3=SUM3+G11(J)
C        SUM4=SUM4+G22(J)
C        SUM5=SUM5+G12(J)
C 14    CONTINUE
C      IF(NUMBM1.LT.1)GO TO 16

```

AD-A082 082

JAYCOR ALEXANDRIA VA

F/B 9/2

USER'S MANUAL FOR COMPUTER PROGRAMS TO PERFORM OCEANOGRAPHIC VE--ETC(U)

JAN 80

N00014-78-C-0879

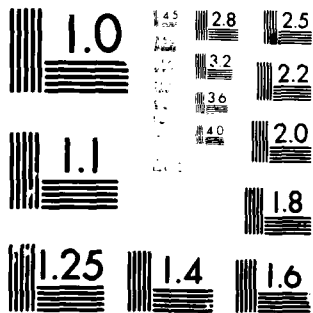
NL

UNCLASSIFIED

2 OF 2

ALL
PAGE(S)

END
DATE
FILMED
4-80
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

AVRG (CONT'D.)

```
C
C CALCULATE NUMBM1 AVERAGED ESTIMATES.
```

```
C DO 15, I=1, NUMBM1
```

```
C SAVE ESTIMATES FOR LATER.
```

```
C A1=GPP(I)
```

```
C A2=GMM(I)
```

```
C A3=G11(I)
```

```
C A4=G22(I)
```

```
C A5=G12(I)
```

```
C
C DIVIDE SUMS BY RAVRG TO CALCULATE AVERAGES AND STORE IN THE 'G'
C ARRAYS.
```

```
C GPP(I)=SUM1/RAVRG
```

```
C GMM(I)=SUM2/RAVRG
```

```
C G11(I)=SUM3/RAVRG
```

```
C G22(I)=SUM4/RAVRG
```

```
C G12(I)=SUM5/RAVRG
```

```
C
C DETERMINE SUMS FOR NEXT ESTIMATES.
```

```
C SUM1=SUM1-A1+GPP(I+NAVRG)
```

```
C SUM2=SUM2-A2+GMM(I+NAVRG)
```

```
C SUM3=SUM3-A3+G11(I+NAVRG)
```

```
C SUM4=SUM4-A4+G22(I+NAVRG)
```

```
C SUM5=SUM5-A5+G12(I+NAVRG)
```

```
C 15 CONTINUE
```

```
C
C DIVIDE SUMS BY RAVRG TO CALCULATE AVERAGES FOR FINAL ESTIMATES IN
C THE 'G' ARRAYS.
```

```
C 16 GPP(NUMB)=SUM1/RAVRG
```

```
C GMM(NUMB)=SUM2/RAVRG
```

```
C G11(NUMB)=SUM3/RAVRG
```

```
C G22(NUMB)=SUM4/RAVRG
```

```
C G12(NUMB)=SUM5/RAVRG
```

```
C RETURN
```

```
C END
```

III. B.4 BAVRG

SUBROUTINE BAVRG(NUMBL,NN)

THIS SUBROUTINE PERFORMS BLOCK AVERAGING ON THE 'G' ARRAYS. AS EACH BLOCK OF 2*NN+1 ELEMENTS IS AVERAGED, THE NEW VALUES ARE MOVED TO THE TOP OF THE 'G' ARRAYS.

VARIABLE LIST

NUMBL THE NUMBER OF BLOCKS TO BE AVERAGED.
 NN THE NUMBER OF ADJACENT FREQUENCY BANDS TO AVERAGE OVER.
 NAVRG THE ACTUAL NUMBER OF POINTS PER BLOCK TO AVERAGE.
 AVRG SAME AS NAVRG BUT REAL.
 SUM1-SUM5 RUNNING SUMS OF THE ESTIMATES PER BLOCK OF THE 'G' ARRAYS.
 I,J,K INDEXES.
 AU,AV,BU,BV INPUT ARRAYS (ALSO USED TO STORE CO- AND QUAD- SPECTRAL ESTIMATES).

COMPLEX U(4097),V(4097),G12(100),SUM5
 REAL GPP(100),GMM(100),G11(100),G22(100)
 COMMON U,V,G12,GPP,GMM,G11,G22
 NAVRG=2*NN+1
 AVRG=FLOAT(NAVRG)

AVERAGE NUMBL BLOCKS.

DO 10 I=1,NUMBL

INITIALIZE SUMS TO ZERO.

SUM1=0.
 SUM2=0.
 SUM3=0.
 SUM4=0.
 SUM5=CMPLX(0.,0.)

SUM NAVRG ESTIMATES.

DO 11 J=1,NAVRG
 K=(I-1)*NAVRG+J
 SUM1=SUM1+GPP(K)
 SUM2=SUM2+GMM(K)
 SUM3=SUM3+G11(K)
 SUM4=SUM4+G22(K)
 SUM5=SUM5+G12(K)

11 CONTINUE

BAVRG (CONT'D.)

```
C
C
C  DIVIDE BY AVRG TO CALCULATE AVERAGES AND STORE IN 'G' ARRAYS.
      GPP(I)=SUM1/AVRG
      GMM(I)=SUM2/AVRG
      G11(I)=SUM3/AVRG
      G22(I)=SUM4/AVRG
      G12(I)=SUM5/AVRG
D  10  CONTINUE
      RETURN
      END
```

III. B.5 STAT

SUBROUTINE STAT(I,STEP,LTRANS)

THIS SUBROUTINE CALCULATES ROTARY SPECTRAL ANALYSIS STATISTICS FROM THE 'G' ARRAYS.

VARIABLE LIST

I	THE POSITION OF THE VALUES IN THE 'G' ARRAYS FROM WHICH THE STATISTICS ARE COMPUTED.
STEP	THE TIME INCREMENT (IN HOURS) BETWEEN THE TIME SERIES ESTIMATES.
PERIOD	THE FIRST (AND LARGEST) PERIOD, IN HOURS, ASSOCIATED WITH FOURIER COEFFICIENTS.
R2	COMMON FACTOR USED IN COMPUTING THE MINIMUM AND MAXIMUM COHERENCIES.
ALPHA	THE ORIENTATION OF THE VELOCITY HODOGRAPH ELLIPSE SEMI-MAJOR AXIS MEASURED ANTICLOCKWISE IN DEGREES FROM EAST.
GAMMA2	THE STABILITY OF THE VELOCITY HODOGRAPH ELLIPSE OR THE COHERENCE SQUARED BETWEEN THE ANTICLOCKWISE AND THE CLOCKWISE ROTATING COMPONENTS.
MAX	THE AMPLITUDE IN CM/SEC OF THE SEMI-MAJOR AXIS OF THE VELOCITY HODOGRAPH ELLIPSE.
MINMAX	THE RATIO OF THE SEMI-MAJOR TO SEMI-MINOR AXES OF THE VELOCITY HODOGRAPH ELLIPSE.
G2MIN	THE MINIMUM COHERENCE SQUARED BETWEEN ORTHOGONAL VELOCITY COMPONENTS. IT IS COMPUTED RELATIVE TO A CO-ORDINATE SYSTEM WHICH IS CO-AXIAL WITH THE NORMAL CO-ORDINATES OF THE VELOCITY HODOGRAPH ELLIPSE.
G2MAX	THE MAXIMUM COHERENCE SQUARED BETWEEN ORTHOGONAL VELOCITY COMPONENTS. IT IS COMPUTED RELATIVE TO A CO-ORDINATE SYSTEM WHICH IS ROTATED 45 DEGREES FROM THE NORMAL COORDINATES OF THE VELOCITY HODOGRAPH ELLIPSE.
GMM	A SPECTRAL ENERGY DENSITY ESTIMATE ASSOCIATED WITH THE CLOCKWISE COMPONENT OF THE VELOCITY HODOGRAPH ELLIPSE.
GPP	A SPECTRAL ENERGY DENSITY ESTIMATE ASSOCIATED WITH THE ANTICLOCKWISE COMPONENT OF THE VELOCITY HODOGRAPH ELLIPSE.
G11	AN AUTOSPECTRAL ESTIMATE OF THE NORTH COMPONENT.
G22	AN AUTOSPECTRAL ESTIMATE OF THE EAST COMPONENT.
G12	A CROSS-SPECTRAL ESTIMATE BETWEEN THE NORTH AND THE EAST COMPONENTS.
'G' ARRAYS	COLLECTIVE TERM FOR GPP, GMM, G11, G22, G12.

STAT (CONT'D.)

```

COMPLEX U(4097),V(4097),G12(100)
REAL GPP(100),GMM(100),G11(100),G22(100),MAX,MINMAX
COMMON U,V,G12,GPP,GMM,G11,G22,ALPHA,GAMMA2,MAX,MINMAX,
      G2MIN,G2MAX
RAD=180./3.14159265
IF (G11(I).NE.G22(I)) GO TO 10
  ALPHA=45.
  GO TO 11
10 ALPHA=ATAN2(2.*REAL(G12(I)),G11(I)-G22(I))/2.*RAD
11 GAMMA2=((G11(I)-G22(I))*2+4.*REAL(G12(I))*2)/((G11(I)+G22(I))*2
      -4.*AIMAG(G12(I))*2)
MAX=SQRT(1./STEP/LTRANS)*(SQRT(GPP(I))+SQRT(GMM(I)))
MINMAX=(SQRT(GPP(I))-SQRT(GMM(I)))/(SQRT(GPP(I))+SQRT(GMM(I)))
R2=1./4.*(G11(I)+G22(I))*2-G11(I)*G22(I)+REAL(G12(I))*2
G2MIN=AIMAG(G12(I))*2/(1./4.*(G11(I)+G22(I))*2-R2)
G2MAX=(R2+(AIMAG(G12(I))*2/(1./4.*(G11(I)+G22(I))*2)
RETURN
END

```

III. B.6 SL

FUNCTION SL(NDOF)

THIS FUNCTION DETERMINES A 90% SIGNIFICANCE LEVEL FOR COHERENCIES GIVEN THE NUMBER OF DEGREES OF FREEDOM. THESE VALUES WERE CALCULATED FROM 'TABLE OF THE DISTRIBUTION OF THE COEFFICIENT OF COHERENCE FOR STATIONARY BIVARIATE GAUSSIAN PROCESSES' BY AMOS AND KOOPMANS, 1963. THE RELATIONSHIP FOR VALUES ASSOCIATED WITH NDOF ≥ 12 WAS FOUND TO BE WITHIN 3% ACCURACY.

VARIABLE LIST

NDOF	THE NUMBER OF DEGREES OF FREEDOM.
SL	SIGNIFICANCE LEVEL.
N	POINTER.

```
IF(NDOF.GT.11)GO TO 200
N=NDOF+1
GO TO(2,2,2,3,4,5,6,7,8,9,10,11),N
2 SL=1.
RETURN
3 SL=.9900
RETURN
4 SL=.9000
RETURN
5 SL=.7845
RETURN
6 SL=.6838
RETURN
7 SL=.6019
RETURN
8 SL=.5359
RETURN
9 SL=.4822
RETURN
10 SL=.4377
RETURN
11 SL=.4005
RETURN
200 SL=4.5578/LOAT(NDOF)
RETURN
END
```

III. B.7 CONINT

SUBROUTINE CONINT(NDOF,CUPPER,CLOWER)

THIS SUBROUTINE CALCULATES A 90% CONFIDENCE INTERVAL FOR CLOCKWISE AND ANTICLOCKWISE SPECTRAL ESTIMATES GIVEN THE NUMBER OF DEGREES OF FREEDOM. THE MULTIPLICATIVE CONFIDENCE INTERVAL FACTORS ARE CALCULATED FROM VALUES OF THE CHI SQUARE DISTRIBUTION.

VARIABLE LIST

NDOF	THE NUMBER OF DEGREES OF FREEDOM.
CUPPER	THE MULTIPLICATIVE FACTOR USED IN DETERMINING THE UPPER VALUE OF THE CONFIDENCE INTERVAL.
CLOWER	THE MULTIPLICATIVE FACTOR USED IN DETERMINING THE LOWER VALUE OF THE CONFIDENCE INTERVAL.
RNDOF	SAME AS NDOF BUT REAL.
CP05	ARRAY CONTAINING VALUES OF THE CHI SQUARE DISTRIBUTION USED IN CALCULATING THE LOWER VALUE OF THE CONFIDENCE INTERVAL.
CP95	ARRAY CONTAINING VALUES OF THE CHI SQUARE DISTRIBUTION USED IN CALCULATING THE UPPER VALUE OF THE CONFIDENCE INTERVAL.

REAL CP05(39),CP95(39)

DATA CP95/.004,.103,.352,.711,1.15,1.64,2.17,2.73,3.33,3.94,4.57,
 5.23,5.89,6.57,7.26,7.96,8.67,9.39,10.1,10.9,11.6,12.3,
 13.1,13.8,14.6,15.4,16.2,16.9,17.7,18.5,26.5,34.8,43.2,
 51.7,60.4,69.1,77.9,86.8,95.7/
 DATA CP05/3.84,5.99,7.81,9.49,11.1,12.6,14.1,15.6,16.9,18.3,19.7,
 21.0,22.4,23.7,25.,26.3,27.6,28.9,30.1,31.4,32.7,33.9,
 35.2,36.4,37.7,38.9,40.1,41.3,42.6,43.8,55.8,67.5,79.1,
 90.5,101.9,113.1,124.3,135.5,146.6/

IF LESS THAN 3 DEGREES OF FREEDOM SET UPPER AND LOWER FACTORS TO
 DEFAULT VALUES SIGNIFYING THERE IS NO CONFIDENCE INTERVAL.

IF(NDOF.GT.2)GO TO 50

CUPPER=99.99

CLOWER=00.00

RETURN

IF LESS THAN 31 DEGREES OF FREEDOM(BUT MORE THAN 2),SET UPPER AND
 LOWER CONFIDENCE INTERVAL FACTORS USING VALUES OF THE CHI SQUARE DIS-
 TRIBUTION.

CONINT (CONT'D.)

```
50 RND0F=FLOAT(NDOF)
   IF(NDOF.GT.30)GO TO 100
   CUPPER=RND0F/CP95(NDOF)
   CLOWER=RND0F/CP05(NDOF)
   RETURN
```

```
C IF LESS THAN 121 DEGREES OF FREEDOM(BUT MORE THAN 30),SET UPPER AND
C LOWER CONFIDENCE INTERVAL FACTORS BY INTERPOLATING VALUES OF THE CHI
C SQUARE DISTRIBUTION.
```

```
100 IF(NDOF.GT.120) GO TO 200
    J=30+NDOF/10-3
    CUPPER=RND0F/(CP95(J)+AMOD(RND0F,10.)/10.+(CP95(J+1)-CP95(J)))
    CLOWER=RND0F/(CP05(J)+AMOD(RND0F,10.)/10.+(CP05(J+1)-CP05(J)))
    RETURN
```

```
C IF GREATER THAN 120 DEGREES OF FREEDOM,SIMPLY SET THE UPPER AND
C LOWER CONFIDENCE INTERVAL FACTORS TO THE VALUES FOR 120 DEGREES OF
C FREEDOM. THIS SHOULD RARELY HAPPEN.
```

```
200 CUPPER=1.25
    CLOWER=.82
    RETURN
    END
```

IV. A. RCSPEC

PROGRAM- RCSPEC
PROGRAMMER- JACK HICKMAN
DATE WRITTEN- MARCH, 1979

THIS PROGRAM PERFORMS A ROTARY CROSS SPECTRAL ANALYSIS OF TIME SERIES DATA. INPUT CONSISTS OF FOUR COMPLEX FOURIER COEFFICIENT DATASETS (A U AND A V COMPONENT FROM EACH OF TWO TIME SERIES), VARIABLES DESCRIBING THE TIME SERIES, AND VARIOUS USER SPECIFIED PARAMETERS. THESE PARAMETERS ALLOW THE USER TO APPLY A VARIETY OF SCALING AND AVERAGING COMBINATIONS. AVERAGING METHODS INCLUDE BLOCK AVERAGING RESULTING IN INDEPENDENT ESTIMATES, CONVOLUTION AVERAGING PRODUCING DEPENDENT ESTIMATES, OR A COMBINATION OF BOTH METHODS. GREAT CARE WAS TAKEN IN THE WRITING OF THIS PROGRAM TO AVOID POTENTIAL CORE REGION PROBLEMS. ALTHOUGH THE ARRAYS CONTAINING THE FOURIER COEFFICIENTS MUST BE SUFFICIENTLY LARGE TO ACCOMMODATE ALL OF THE INPUT DATA, THE 'U' ARRAYS, WHICH CONTAIN THE VALUES THAT ARE AVERAGED AND THEN USED IN THE CALCULATION OF THE OUTPUT DATA, ARE OF VARIABLE LENGTH. THE USER MAY ALTER THE SIZES OF THESE ARRAYS TO SUIT HIS OR HER PARTICULAR DATA AND MACHINE. NATURALLY, THE SMALLER THE ARRAYS ARE, THE MORE SWAPPING IS REQUIRED RESULTING IN GREATER EXECUTION TIMES. THE ONLY RESTRICTION IS THAT THE SIZE OF THE 'U' ARRAYS MUST BE GREATER THAN DOUBLE THE LARGEST NN VALUE USED. THE VALUE OF THE VARIABLE 'IASIZE' MUST BE ALTERED TO REFLECT THE SIZE OF THE 'U' ARRAYS.

SUBROUTINES REQUIRED

CALC
SWAP
AVRG
BAYRG
PHACDH
SL

INPUT PARAMETERS

LTRANS	THE SIZE OF THE FOURIER TRANSFORM APPLIED TO THE TIME SERIES DATA.
STEP	THE TIME INCREMENT (IN HOURS) BETWEEN THE TIME SERIES ESTIMATES.
LENGTH	THE NUMBER OF ESTIMATES IN THE TIME SERIES.
NINPUT	THE NUMBER OF FOURIER COEFFICIENTS INPUT.
IOPT	DETERMINES THE TYPE OF AVERAGING TO USE: 1 BLOCK AVERAGING.

RCSPEC (CONT'D.)

C 2 CONVOLUTION AVERAGING.
 C 3 COMBINATION OF BLOCK AND CONVOLUTION AVERAGING.
 C NN THE NUMBER OF ADJACENT FREQUENCY BANDS TO AVERAGE OVER.
 C NUMBNM THE NUMBER OF NN VALUES TO USE IF BLOCK AVERAGING.
 C NIN1,NIN2 THE FORTRAN UNIT REFERENCE NUMBERS FOR THE FOURIER
 C EIN1,EIN2 COEFFICIENT DATASETS TO INPUT.
 C NWINDO DENOTES THE TYPE OF SPECTRAL WINDOW USED WHEN COMPUTING
 C FOURIER COEFFICIENTS:
 C 0 BOXCAR (NO) WINDOW.
 C 1 10% COSINE WINDOW.
 C 2 HANNING WINDOW.
 C 3 HAMMING WINDOW.
 C 4 PARZEN WINDOW.
 C 5 LANZOS WINDOW.
 C IWSOPT DENOTES WHETHER SPECTRAL WINDOW SCALING SHOULD BE
 C APPLIED:
 C 0 DO NOT SCALE.
 C 1 APPLY SPECTRAL WINDOW SCALING.
 C PSTCLR DENOTES IF DATA SHOULD BE POST-COLORED OR NOT:
 C 0 DO NOT APPLY POST-COLORING.
 C 1 APPLY POST-COLORING.
 C NFMT1,NFMT2 THE FORMATS OF THE COMPLEX FOURIER COEFFICIENT
 C EFMT1,EFMT2 DATASETS TO INPUT.
 C AU,AV,BU,BV ARRAYS FOR THE INPUT FOURIER COEFFICIENTS. ALSO
 C USED TO STORE CO- AND QUAD- SPECTRAL ESTIMATES.
 C ISTART ARRAY CONTAINING THE POSITIONS IN THE FOURIER COEFFICIENT
 C ARRAYS WHERE DIFFERENT NN VALUES ARE USED DURING
 C BLOCK AVERAGING.
 C MNS ARRAY OF DIFFERENT NN VALUES FOR USE WHEN BLOCK AVERAGING.

VARIABLE LIST

C IN FORTRAN UNIT REFERENCE NUMBER OF INPUT FILE.
 C OUT FORTRAN UNIT REFERENCE NUMBER OF OUTPUT FILE.
 C PERIOD THE FIRST (AND LARGEST) PERIOD, IN HOURS, ASSOCIATED WITH
 C THE FOURIER COEFFICIENTS.
 C IASIZE THE LENGTH OF THE 'U' ARRAYS. THIS NUMBER, ALONG WITH
 C THE SIZES OF THE 'U' ARRAYS SHOULD BE ADJUSTED TO
 C COMPENSATE FOR CORE REGION AND TURN-AROUND TIME.
 C P A PERIOD (IN HOURS) ASSOCIATED WITH OUTPUT VALUES.
 C F A FREQUENCY (IN CYCLES/HOUR) ASSOCIATED WITH OUTPUT
 C VALUES.
 C NDOF THE NUMBER OF DEGREES OF FREEDOM.
 C SIGLVL THE 90% SIGNIFICANCE LEVEL VALUE FOR COHERENCE ASSOCIATED
 C WITH A GIVEN NDOF.
 C NOUT A COUNTER FOR THE NUMBER OF OUTPUT LINES.
 C INPUT THE INDEX FOR THE INPUT (FOURIER COEFFICIENTS) ARRAYS.
 C IT ALWAYS POINTS TO THE NEXT VALUE TO BE USED.
 C NSTORE ANOTHER INDEX FOR THE INPUT ARRAYS POINTING TO THE NEXT
 C LOCATION AVAILABLE FOR STORING CO- AND QUAD- SPECTRAL
 C ESTIMATES.

RCSPEC (CONT'D.)

```

C  MINPUT      THE NUMBER OF PAIRS OF FOURIER COEFFICIENTS GENERATED
C              BY THE FFT.
C  LENCON      THE NUMBER OF VALUES TO CONVOLUTION AVERAGE.
C  LENG        THE NUMBER OF VALUES IN THE 'U' ARRAYS.
C  NUMB        THE NUMBER OF AVERAGED VALUES IN THE 'U' ARRAYS AFTER
C              AVERAGING.
C  NNx2        TWO TIMES NN.
C  LEFT        THE NUMBER OF ESTIMATES LEFT TO AVERAGE AFTER THE
C              FULL 'U' ARRAYS HAVE BEEN AVERAGED.
C  COABNN,COABNP, THE SQUARED COHERENCIES BETWEEN THE PAIRS OF POLARIZED
C  COABPN,COABPP, CONSTITUENTS AT LOCATIONS A AND B.
C  PHABNN,PHABNP, THE PHASES FOR THE PAIRS OF POLARIZED CONSTITUENTS AT
C  PHABPN,PHABPP, LOCATIONS A AND B.
C  UACNAN,...., PRODUCTS OF ANTICLOCKWISE AND/OR CLOCKWISE COMPLEX
C  UBCPBP      FOURIER COEFFICIENTS AT LOCATIONS A AND/OR B. ALSO
C              KNOWN AS THE 'U' ARRAYS.
C  NTA         THE NUMBER OF TIMES TO AVERAGE FULL 'U' ARRAYS.
C  LARRAY      THE NUMBER OF ELEMENTS IN THE 'U' ARRAYS TO BE BLOCK
C              AVERAGED.
C  NBLOCK      THE NUMBER OF BLOCKS TO BE BLOCK AVERAGED.
C  NBLPRA      THE NUMBER OF BLOCKS PER 'U' ARRAY TO AVERAGE.
C  I,J,K,L     INDEXES.

C  COMPLEX AU(4097),AV(4097),BU(4097),BV(4097),
C  *          UACBPB(50),UACNBN(50),UACPNB(50),UACNBP(50),
C  *          UACPAP(50),UBCPBP(50),UACNAN(50),UBCNBN(50)
C  *          INTEGER ISTART(30),NNS(30),OUT,OUTPLT,PSTCLR
C  *          COMMON AU,AV,BU,BV,UACBPB,UACNBN,UACPNB,UACNBP,UACPAP,UBCPBP,
C  *          UACNAN,UBCNBN,
C  *          PHABPP,PHABNN,PHABPN,PHABNP,COABPP,COABNN,COABPN,COABNP
C  *          IASIZE=50

C  SET FORTRAN REFERENCE NUMBERS FOR INPUT AND OUTPUT FILES.
C
C  IN=5
C  OUT=21
C  OUTPLT=11

C  INPUT PARAMETERS AND ECHO CHECK.
C
C  READ(17,2001)LTRANS,LENGTH,NWINDO
2001 FORMAT(2I5,I1)
C  WRITE(6,2003)
2003 FORMAT(' INPUT STEP,PSTCLR,IWSOPT,IOPT,NN,NUMBNN')
C  READ(IN,2002)STEP,PSTCLR,IWSOPT,IOPT,NN,NUMBNN
2002 FORMAT(F10.0,6I5)
C  NINPUT=LTRANS/2+1
C  WRITE(6,1101)LTRANS,LENGTH,NWINDO,STEP,PSTCLR,IWSOPT,IOPT,NN,
C  *          NUMBNN,NINPUT
1101 FORMAT('///',5X,'LTRANS',4X,'LENGTH',5X,'NWINDO',STEP,7X,
C  *          'PSTCLR',///,3I10,F10.3,I10///,6X,'IWSOPT',5X,
C  *          'IOPT',7X,'NN',6X,'NUMBNN',NINPUT///,5I10)

```

RCSPEC (CONT'D.)

```

C
C INPUT FOUR SERIES OF FOURIER COEFFICIENTS: A U AND V COMPONENT
C FROM EACH OF TWO TIME SERIES.
C
      READ(17,1103) (AV(I), I=1, NINPUT)
1103  FORMAT(2A6)
      READ(18,2001) LTRANS, LENGTH, NWINDO
      READ(18,1103) (AU(I), I=1, NINPUT)
      READ(19,2001) LTRANS, LENGTH, NWINDO
      READ(19,1103) (BV(I), I=1, NINPUT)
      READ(20,2001) LTRANS, LENGTH, NWINDO
      READ(20,1103) (BU(I), I=1, NINPUT)
1001  FORMAT(' INDEX FREQ PERIOD', 5X, 'COH**2 PHASE(DEG)', 2X,
*          'COH**2 PHASE(DEG) COH**2 PHASE(DEG) COH**2', 2X,
*          'PHASE(DEG)///', 10X, 'CPH', 5X, 'HOURS', 7X, '++', 8X, '++', 8X,
*          '--', 8X, '--', 8X, '+-', 8X, '+-', 8X, '+-', 8X, '-+//')
C
C INITIALIZE VARIABLES.
C
      IF(IWSOPT.EQ.1) NWINDO=NWINDO+10
      PERIOD=LTRANS*STEP
      NSTORE=1
      NOUT=0
      LENCON=NINPUT
C
C IF ONLY PERFORMING CONVOLUTION AVERAGING, SKIP OVER SECTION INPUT-
C TING START AND NN VALUES FOR BLOCK AVERAGING. IF NOT, INPUT AND
C ECHO CHECK THESE VALUES.
C
      IF(IDPT.NE.2) GO TO 49
      NUMOUT=LTRANS/2
      WRITE(OUTPLT, 6500) NUMOUT
6500  FORMAT(A6)
      GO TO 50
C
49  WRITE(6,1008)
1008  FORMAT('///', 17X, 'INDEX', 5X, 'START', 6X, 'NN//')
      READ(IN, 2004) (ISTART(I), NNS(I), I=1, NUMBNN)
2004  FORMAT(14I5)
      WRITE(6,1009) (I, ISTART(I), NNS(I), I=1, NUMBNN)
1009  FORMAT(' ', 10X, 3I10)
      NUMOUT=0
      IF(IDPT.EQ.3.AND. (ISTART(1)-NN-2).GT.0) NUMOUT=ISTART(1)-NN-2
      J=NUMBNN-1
      DO 48 I=1, J
          NUMOUT=NUMOUT+(ISTART(I+1)-ISTART(I))/(NNS(I)+2+1)
48  CONTINUE
      NUMOUT=NUMOUT+(NINPUT-ISTART(NUMBNN))/(NNS(NUMBNN)+2+1)
      WRITE(OUTPLT, 6500) NUMOUT
      LENCON=ISTART(1)-1
C
C OUTPUT HEADINGS.
C
50  WRITE(OUT, 1001)

```

RCSPEC (CONT'D.)

IF ONLY PERFORMING BLOCK AVERAGING, SKIP OVER SECTION IMPLEMENTING
CONVOLUTION AVERAGING.

IF (IDPT.LT.2.OR.LENCON.LT.(NN+1)) GO TO 40

CONVOLUTION AVERAGING.

CALCULATE AND OUTPUT THE NUMBER OF DEGREES OF FREEDOM AND THE COR-
RESPONDING SIGNIFICANCE LEVEL.

NDOF=2*(2*NN+1)*LENGTH/LTRANS
SIGLVL=SL(NDOF)
WRITE(OUT,3001)NN,NDOF,SIGLVL

INITIALIZE VARIABLES ASSOCIATED WITH CONVOLUTION AVERAGING.

INPUT=1
NUMB=IASIZE-2*NN
P=0.
F=0.
LENG=IASIZE
IF (IASIZE.GT.(LENCON+NN)) LENG=LENCON+NN
K=NN+1

NOW FILL THE 'U' ARRAYS WITH UNAVERAGED VALUES. ESTIMATES MUST BE
'FOLDED' AROUND THE ZERO FREQUENCY ESTIMATE.

DO 11 I=K,LENG
CALL CALC(INPUT,I,STEP,LENGTH,NINPUT,PSTCLR,NWINDO)
INPUT=INPUT+1
11 CONTINUE
NNX2=NN*2
DO 12 I=1,NN
J=NNX2+2-I
CALL SWAP(I,J)
12 CONTINUE

IF THE 'U' ARRAYS ARE NOT FULL SKIP THE SECTION AVERAGING FULL
ARRAYS.

IF (IASIZE.GT.(LENCON+NN)) GO TO 52
NTA=1+(LENCON-(IASIZE-NN))/(IASIZE-NNX2)

THIS LOOP AVERAGES FULL ARRAYS, OUTPUTS PHASE AND COHERENCE DATA,
MOVES VALUES TO THE TOPS OF ARRAYS, AND REFILLS THE REMAINDERS OF
THE ARRAYS.

DO 19 K=1,NTA
CALL AVRG(IASIZE,NN,NSTORE)

RCSPEC (CONT'D.)

```

C
C THIS LOOP OUTPUTS THE AVERAGED PHASE AND COHERENCE VALUES ALONG
C WITH THE ASSOCIATED PERIODS AND FREQUENCIES.
C
      DO 16 I=1,NUMB
        NOUT=NOUT+1
        CALL PHACOH(I)
        WRITE(OUT,1003)NOUT,F,P,COABPP,PHABPP,COABNN,PHABNN,COABPN,
          PHABPN,COABNP,PHABNP
          IF(F.NE.0.)WRITE(OUTPLT,7001)F,COABPP,PHABPP,COABNN,PHABNN,
            COABPN,PHABPN,COABNP,PHABNP
7001      FORMAT(9A6)
1003      FORMAT(' ',I5,F9.5,F10.2,4(F9.2,F11.4))
          IF(NOUT/60*60.EQ.NOUT)WRITE(OUT,1004)
1004      FORMAT('1'///)
          P=PERIOD/FLOAT(NOUT)
          F=1./P
      16  CONTINUE

C
C NOW MOVE THE LAST 2*NN ELEMENTS TO THE TOPS OF THE U ARRAYS.
C
      J=IASIZE-NNX2
      DO 17 I=1,NNX2
        J=J+1
        CALL SWAP(I,J)
      17  CONTINUE

C
C REFILL THE 'U' ARRAYS UNLESS GOING THROUGH THE LOOP FOR THE LAST TIME
C
      IF(K.EQ.NTA)GO TO 19
      L=NNX2+1
      DO 18 I=L,IASIZE
        CALL CALC(INPUT,I,STEP,LENGTH,NINPUT,PSTCLR,NWINDO)
        INPUT=INPUT+1
      18  CONTINUE
      19 CONTINUE

C
C IF NECESSARY, PLACE REMAINING VALUES IN THE 'U' ARRAYS.
C
      LEFT=LENCON-(INPUT-1)
      LENG=NNX2+LEFT
      IF(LEFT.LT.1)GO TO 52
      J=NNX2+1
      DO 51 I=J,LENG
        CALL CALC(INPUT,I,STEP,LENGTH,NINPUT,PSTCLR,NWINDO)
        INPUT=INPUT+1
      51  CONTINUE

C
C IF NO NEED TO 'FOLD' AROUND THE LAST ESTIMATE SKIP THE NEXT SECTION.
C
      52 IF(IDPT.GT.2.OR.NN.EQ.0)GO TO 55
      J=LENG+NN
      IF(J.GT.IASIZE)J=IASIZE

```

RCSPEC (CONT'D.)

```

I=LENG+1
LENG=LENG+NN

```

```

C 'FOLD' AROUND THE LAST ESTIMATE UNLESS THERE ISN'T ENOUGH ROOM IN
C THE 'U' ARRAYS TO DO SO.
C

```

```

IF(I.GT.IASIZE)GO TO 59

```

```

L=I

```

```

K=I-1

```

```

DO 54 I=L,J

```

```

K=K-1

```

```

CALL SWAP(I,K)

```

```

54 CONTINUE

```

```

C IF THE 'U' ARRAYS ARE NOT FULL SKIP TO SECTION FOR HANDLING THEM.
C OTHERWISE, AVERAGE THE ESTIMATES AND OUTPUT THE PHASE AND COHE-
C RENCE DATA ALONG WITH THE ASSOCIATED PERIODS AND FREQUENCIES.
C

```

```

IF(LENG.LE.IASIZE)GO TO 55

```

```

59 CALL AVRG(IASIZE,NN,NSTORE)

```

```

DO 56 I=1,NUMB

```

```

NOUT=NOUT+1

```

```

CALL PHACOH(I)

```

```

WRITE(OUT,1003)NOUT,F,P,COABPP,PHABPP,COABNN,PHABNN,COABPN,
PHABPN,COABNP,PHABNP

```

```

IF(F.NE.0.)WRITE(OUTPLT,7001)F,COABPP,PHABPP,COABNN,PHABNN,
COABPN,PHABPN,COABNP,PHABNP

```

```

IF(NOUT/60*60.EQ.NOUT)WRITE(OUT,1004)

```

```

P=PERIOD/FLOAT(NOUT)

```

```

F=1./P

```

```

56 CONTINUE

```

```

C NOW MOVE THE LAST 2*NN VALUES TO THE TOP OF THE 'U' ARRAYS AND
C 'FOLD' AROUND THE LAST ESTIMATE.
C

```

```

J=IASIZE-NNX2

```

```

DO 57 I=1,NNX2

```

```

J=J+1

```

```

CALL SWAP(I,J)

```

```

57 CONTINUE

```

```

LENG=LENG-IASIZE+NNX2

```

```

L=LENG-NNX2

```

```

J=L+LENG-NNX2-1

```

```

K=LENG+1

```

```

DO 58 I=L,J

```

```

K=K-1

```

```

CALL SWAP(K,I)

```

```

58 CONTINUE

```

```

C THIS SECTION IS FOR HANDLING SITUATIONS WHEN THE 'U' ARRAYS ARE
C NOT FULL. FIRST,THE ESTIMATES ARE AVERAGED. THEN THE AVERAGED
C PHASE AND COHERENCE VALUES ARE OUTPUT.
C

```

RCSPEC (CONT'D.)

```

55 IF (LENG.EQ.0)GO TO 60
   CALL AVRG (LENG*NN,NSTORE)
   NUMB=LENG-2*NN
   DO 53 I=1,NUMB
     NOUT=NOUT+1
     CALL PHACOH(I)
     WRITE (OUT,1003) NOUT,F,P,COABPP,PHABPP,COABNN,PHABNN,COABPN,
      PHABPN,COABNP,PHABNP
     IF (F.NE.0.)WRITE (OUTPLT,7001) F,COABPP,PHABPP,COABNN,PHABNN,
      COABPN,PHABPN,COABNP,PHABNP
     IF (NOUT/60*60.EQ.NOUT)WRITE (OUT,1004)
     P=PERIOD/FLOAT (NOUT)
     F=1./P
53  CONTINUE

```

UNLESS BLOCK AVERAGING IS TO BE PERFORMED, OUTPUT CO- AND QUAD-
SPECTRAL ESTIMATES WHICH ARE NOW STORED IN THE INPUT ARRAYS.

```

60 IF(IOPT.GT.2)GO TO 40
   WRITE(OUT,1007)
1007 FORMAT('1',2X,'INDEX  FREQ',4X,'PERIOD',2X,4(3X,'COSPEC',3X,
  ◆      'QUADSPEC')//',10X,'CPH',5X,'HOURS',7X,'++',8X,'++',8X,
  ◆      '--',8X,'--',8X,'+-',8X,'+-',8X,'→',8X,'→')
   WRITE(OUT,3002)NN,NDOF
   P=0.
   F=0.
   DO 20 I=1,LENCON
     WRITE(OUT,1005)I,F,P,AU(I),AV(I),BU(I),BV(I)
1005   FORMAT(' ',I5,F9.5,F9.2,2X,8F10.2)
     IF(I/60#.EQ.I)WRITE(OUT,1004)
     P=PERIOD/FLOAT(I)
     F=1./P
   20 CONTINUE
   STOP

```

BLOCK AVERAGING.

```
40 LENCON=LENCON-NN
   ISTART (NUMBN+1)=NINPUT
```

PERFORM BLOCK AVERAGING USING NUMBNN BLOCK SIZES.

DO 3.3 K=1 • NUMBNN

CALCULATE VALUES BASED ON THE SIZE OF THE AVERAGING INTERVAL. ALSO, DETERMINE THE NUMBER OF DEGREES OF FREEDOM AND ITS ASSOCIATED SIGNIFICANCE LEVEL AND OUTPUT THEM.

```

NN=NNS (K)
LENGAY=2*NN+1
INPUT=I START (K)

```

RCSPEC (CONT'D.)

```

NBLOCK=(ISTART(K+1)-ISTART(K))/LENGAV
NBLPRA=IASIZE/LENGAV
NTA=NBLOCK/NBLPRA
NDOF=2*(2*NN+1)*LENGTH/LTRANS
SIGLYL=SL(NDOF)
WRITE(OUT,3001)NN,NDOF,SIGLYL
3001 FORMAT(' ',6X,'***** NN=',I2,' *****',6X,'DEGREES OF FREEDOM=',I3,
*        ' ',6X,'SIGNIFICANCE LEVEL=',F6.2)
      I=1
      IF(NTA.EQ.0) GO TO 34
      LARRAY=NBLPRA*LENGAV
C
C THIS LOOP FILLS THE 'U' ARRAYS, AVERAGES THE ESTIMATES, AND OUTPUTS
C THE PHASE AND COHERENCE DATA.
C
      DO 24 I=1,NTA
C
C THIS LOOP FILLS THE 'U' ARRAYS COMPLETELY.
C
          DO 25 J=1,LARRAY
              CALL CALC(INPUT,J,STEP,LENGTH,NINPUT,PSTCLR,NWINDO)
              INPUT=INPUT+1
          25      CONTINUE
              CALL BAVRG(NBLPRA,NN,NSTORE)
C
C NOW OUTPUT THE PHASE AND COHERENCE VALUES WITH THE ASSOCIATED PERIODS
C AND FREQUENCIES.
C
          DO 26 J=1,NBLPRA
              NOUT=NOUT+1
              CALL PHACOH(J)
              P=PERIOD/FLOAT(ISTART(K)+(I-1)*LARRAY+(J-1)*LENGAV+NN-1)
              F=1./P
              WRITE(OUT,1003)NOUT,F,P,COABPP,PHABPP,COABNN,PHABNN,COABPN,
*              PHABPN,COABNP,PHABNP
              WRITE(OUTPLT,7001)F,COABPP,PHABPP,COABNN,PHABNN,COABPN,
*              PHABPN,COABNP,PHABNP
              IF(NOUT/60*60.EQ.NOUT)WRITE(OUT,1004)
          26      CONTINUE
      24      CONTINUE
C
C THIS SECTION IS FOR THE SITUATION WHEN THE 'U' ARRAYS CANNOT BE
C COMPLETELY FILLED.
C
      I=NTA+1
      34 NBLPRA=NBLOCK-NTA*NBLPRA
      IF(NBLPRA.EQ.0)GO TO 33
C
C FILL AS MUCH OF THE 'U' ARRAYS AS POSSIBLE USING THE REMAINDER OF
C THE INPUT VALUES.
C
      LARRAY=NBLPRA*LENGAV

```

RCSPEC (CONT'D.)

```

DO 31 J=1,LARRAY
  CALL CALC(INPUT,J,STEP,LENGTH,NINPUT,PSTCLR,NWINDO)
  INPUT=INPUT+1

```

```

31 CONTINUE

```

AVERAGE THE ESTIMATES IN THE 'U' ARRAYS AND OUTPUT THE PHASE AND COHERENCE VALUES.

```

CALL BAVRG(NBLPRA,NN,NSTORE)

```

```

LARRAY=IASIZE/LENGAV*LENGAV

```

```

DO 32 J=1,NBLPRA

```

```

  NOUT=NOUT+1

```

```

  CALL PHACOH(J)

```

```

  P=PERIOD/FLOAT(ISTART(K)+(I-1)*LARRAY+(J-1)*LENGAV+NN-1)

```

```

  F=1./P

```

```

  WRITE(OUT,1003)NOUT,F,P,COABPP,PHABPP,COABNN,PHABNN,COABPN,
    PHABPN,COABNP,PHABNP

```

```

  WRITE(OUTPLT,7001)F,COABPP,PHABPP,COABNN,PHABNN,COABPN,PHABPN,
    COABNP,PHABNP

```

```

  IF(NOUT/60*60.EQ.NOUT)WRITE(OUT,1004)

```

```

32 CONTINUE

```

```

33 CONTINUE

```

THIS SECTION OUTPUTS THE CO- AND QUAD- SPECTRAL ESTIMATES. FIRST PRINT THE HEADING.

```

WRITE(OUT,1007)

```

```

NOUT=0

```

IF CONVOLUTION AVERAGING WAS NOT PERFORMED, SKIP THIS SECTION. OTHERWISE, CALCULATE AND OUTPUT THE NUMBER OF DEGREES OF FREEDOM AND OUTPUT THE CO- AND QUAD- SPECTRAL ESTIMATES.

```

IF(IOPT.LT.2.OR.LENCON.LT.1)GO TO 36

```

```

NDOF=2*(NMX2+1)*LENGTH/LTRANS

```

```

WRITE(OUT,3002)NN,NDOF

```

```

P=0.0

```

```

F=0.0

```

```

DO 37 I=1,LENCON

```

```

  WRITE(OUT,1005)I,F,P,AU(I),AV(I),BU(I),BV(I)

```

```

  IF(I/60*60.EQ.I)WRITE(OUT,1004)

```

```

  P=PERIOD/FLOAT(I)

```

```

  F=1./P

```

```

37 CONTINUE

```

```

NOUT=LENCON

```

THIS LOOP OUTPUTS CO- AND QUAD- SPECTRAL ESTIMATES FOR VALUES THAT HAVE BEEN BLOCK AVERAGED. FIRST, THE NUMBER OF DEGREES OF FREEDOM CORRESPONDING TO EACH NN VALUE IS CALCULATED AND OUTPUT. THEN, THE SPECTRAL ESTIMATES ARE PRINTED.

RCSPEC (CONT'D.)

```

36 DO 35 K=1,NUMBNN
    NN=NNS(K)
    LENGAV=2*NN+1
    NDOF=2*LENGAV*LENGTH/LTRANS
    WRITE(OUT,3002)NN,NDOF
    NBLOCK=(ISTART(K+1)-ISTART(K))/LENGAV
    DO 30 I=1,NBLOCK
        NOUT=NOUT+1
        P=PERIOD/FLOAT(ISTART(K)+(I-1)*LENGAV+NN-1)
        F=1./P
        WRITE(OUT,1005)NOUT,F,P,AU(NOUT),AV(NOUT),BU(NOUT),BV(NOUT)
        IF(NOUT/60*60.EQ.NOUT)WRITE(OUT,1004)
    30    CONTINUE
    35    CONTINUE
    STOP
3002 FORMAT(6X,'◆◆◆NN=',I3,' ◆◆◆',6X,'DEGREES OF FREEDOM=',I3)
END

```

IV. B.1 CALC

SUBROUTINE CALC(INPUT,I,STEP,LENGTH,NINPUT,PSTCLR,NWINDO)

THIS SUBROUTINE CALCULATES ONE VALUE IN EACH OF THE 'U' ARRAYS GIVEN ONE VALUE FROM EACH OF THE INPUT ARRAYS. THE 'U' ARRAYS ARE PRODUCTS OF CLOCKWISE AND/OR ANTICLOCKWISE COMPLEX FOURIER COEFFICIENTS AT LOCATIONS A AND/OR B. IF THE USER DESIRES THEM, POST-COLORING AND/OR SPECTRAL WINDOW SCALING ARE IMPLEMENTED IN THIS ROUTINE.

VARIABLE LIST

INPUT	THE POSITION IN THE INPUT ARRAYS OF THE FOURIER COEFFICIENTS TO USE TO CALCULATE THE VALUES FOR THE 'U' ARRAYS.
I	THE POSITION IN THE 'U' ARRAYS TO STORE THE CALCULATED VALUES.
STEP	THE TIME INCREMENT (IN HOURS) BETWEEN THE TIME SERIES ESTIMATES.
LENGTH	THE NUMBER OF POINTS IN THE ORIGINAL TIME SERIES.
NINPUT	THE NUMBER OF FOURIER COEFFICIENTS INPUT.
PSTCLR	DETERMINES IF DATA SHOULD BE POST-COLORED OR NOT: 0 DO NOT POST-COLOR. 1 POST COLOR.
NWINDO	DENOTES THE TYPE OF SPECTRAL WINDOW USED WHEN COMPUTING FOURIER COEFFICIENTS AND IF SPECTRAL WINDOW SCALING SHOULD BE PERFORMED. 0 BOXCAR (NO) WINDOW. 1 10% COSINE WINDOW. 2 HANNING WINDOW. 3 HAMMING WINDOW. 4 PARZEN WINDOW. 5 LANZOS WINDOW. IF THE VALUE OF NWINDO IS GREATER THAN 10, THEN SPECTRAL WINDOW SCALING SHOULD BE PERFORMED USING SPECTRAL WINDOW NWINDO-10.
PI	SELF EXPLANATORY.
FACTOR	MULTIPLICATIVE SCALING FACTOR. INITIALLY EQUAL TO STEP/LENGTH BUT MAY BE CHANGED BY POST-COLORING AND/OR SPECTRAL WINDOW SCALING.
UAPOS	ANTICLOCKWISE COMPLEX FOURIER COEFFICIENT AT LOCATION A. UANEG, UBPOS, UBNEG, UAPOSC (CONJUGATE OF UAPOS), UANEGC ARE OTHER VARIABLES LIKE THIS AT DIFFERING LOCATIONS AND/OR ROTATIONS.
'U' ARRAYS	PRODUCTS OF ANTICLOCKWISE AND/OR CLOCKWISE COMPLEX FOURIER COEFFICIENTS AT LOCATIONS A AND/OR B. ALL CALCULATED VALUES FROM THIS PROGRAM ARE DERIVED FROM THESE ARRAYS.
N	POINTER.

CALC (CONT'D.)

```

C AU,AV,BU,BV  ARRAYS CONTAINING COMPLEX FOURIER COEFFICIENTS.
C
C   COMPLEX AU(4097),AV(4097),BU(4097),BV(4097),
C   *   UACBPB(50),UACNBN(50),UACPNB(50),UACNBP(50),
C   *   UACPAP(50),UBCPBP(50),UACNAN(50),UBCNBN(50)
C   COMPLEX UAPDS,UANEG,UBPDS,UBNEG,UAPDSC,UANEGC
C   INTEGER PSTCLR
C   COMMON AU,AV,BU,BV,UACBPB,UACNBN,UACPNB,UACNBP,UACPAP,UBCPBP,
C   *   UACNAN,UBCNBN
C   PI=3.1415927
C   FACTOR=STEP/FLOAT(LENGTH)
C
C DETERMINE IF POST-COLORING SHOULD BE IMPLEMENTED. IF SO,MODIFY THE
C VALUE OF FACTOR TO REFLECT THIS.
C
C   IF(PSTCLR.NE.1)GO TO 10
C
C TEST FOR POSSIBLE DIVIDE CHECK PROBLEMS. IF FOUND,DIVIDE FACTOR
C BY A DEFAULT VALUE.
C
C   IF(INPUT.NE.1.AND.(PI*FLOAT(INPUT-1)/(FLOAT(NINPUT-1)*2.))
C   *   .NE.0)GO TO 12
C   FACTOR=FACTOR/.000001
C   GO TO 10
12  FACTOR=FACTOR/(4.*SIN(PI*FLOAT(INPUT-1)/
C   *   (FLOAT(NINPUT-1)*2.))*2)
C
C DETERMINE IF SPECTRAL WINDOW SCALING SHOULD BE IMPLEMENTED. IF SO,
C MODIFY THE VALUE OF FACTOR TO REFLECT THIS.
C
10 IF(NWINDO.LT.11)GO TO 11
   N=NWINDO-10
   GO TO (1,2,3,4,5),N
1   FACTOR=FACTOR/.873239
   GO TO 11
2   FACTOR=FACTOR/.374245
   GO TO 11
3   FACTOR=FACTOR/.396612
   GO TO 11
4   FACTOR=FACTOR/.269099
   GO TO 11
5   FACTOR=FACTOR/.450503
C
C COMPUTE THE ANTICLOCKWISE AND CLOCKWISE FOURIER COEFFICIENTS AT
C LOCATIONS A AND B AND THEIR CONJUGATES.
C
11 UAPDS=AU(INPUT)+(0.,1.)*AV(INPUT)
   UANEG=AU(INPUT)-(0.,1.)*AV(INPUT)
   UBPDS=BU(INPUT)+(0.,1.)*BV(INPUT)
   UBNEG=BU(INPUT)-(0.,1.)*BV(INPUT)
   UAPDSC=CONJG(UAPDS)
   UANEGC=CONJG(UANEG)

```

CALC (CONT'D.)

C
C COMPUTE VALUES FOR THE 'U' ARRAYS.
C

UACBPB(I)=UAPOSC*UBPOS*FACTOR
UACNBN(I)=UANEGC*UBNEG*FACTOR
UACPBN(I)=UAPOSC*UBNEG*FACTOR
UACNBP(I)=UANEGC*UBPOS*FACTOR
UACPAP(I)=UAPOSC*UAPOS*FACTOR
UBCPBP(I)=CONJG(UBPOS)*UBPOS*FACTOR
UACNAN(I)=UANEGC*UANEG*FACTOR
UBCNBN(I)=CONJG(UBNEG)*UBNEG*FACTOR
RETURN
END

IV. B.2 SWAP

SUBROUTINE SWAP(I,J)

THIS SUBROUTINE MOVES VALUES IN THE 'U' ARRAYS AT POSITION J TO
POSITION I.

VARIABLE LIST

J LOCATION OF VALUES IN THE 'U' ARRAYS TO BE MOVED.
I LOCATION IN THE 'U' ARRAYS WHERE VALUES ARE MOVED TO.

COMPLEX AU(4097),AV(4097),BU(4097),BV(4097),
UACBPB(50),UACNBN(50),UACPBN(50),UACNBP(50),
UACPAP(50),UBCPBP(50),UACNAN(50),UBCNBN(50)
COMMON AU,AV,BU,BV,UACBPB,UACNBN,UACPBN,UACNBP,UACPAP,UBCPBP,
UACNAN,UBCNBN
UACBPB(I)=UACBPB(J)
UACNBN(I)=UACNBN(J)
UACPBN(I)=UACPBN(J)
UACNBP(I)=UACNBP(J)
UACPAP(I)=UACPAP(J)
UBCPBP(I)=UBCPBP(J)
UACNAN(I)=UACNAN(J)
UBCNBN(I)=UBCNBN(J)

RETURN
END

IV. B.3 AVRG

SUBROUTINE AVRG(LENG,NN,NSTORE)

THIS SUBROUTINE PERFORMS CONVOLUTION(RUNNING) AVERAGING ON THE 'U' ARRAYS. THE AVERAGED ESTIMATES ARE STORED AT THE 'TOP' OF THE 'U' ARRAYS. THE CO- AND QUAD- SPECTRAL ESTIMATES ARE STORED IN THE INPUT ARRAYS(AU,AV,BU,BV).

VARIABLE LIST

LENG THE NUMBER OF ESTIMATES IN THE 'U' ARRAYS TO BE INVOLVED IN AVERAGING.
 NN THE NUMBER OF ADJACENT FREQUENCY BANDS TO AVERAGE OVER.
 NSTORE THE NEXT POSITION IN THE INPUT ARRAYS TO BE FILLED WITH CO- AND QUAD- SPECTRAL ESTIMATES.
 NAVRG THE ACTUAL NUMBER OF ESTIMATES TO AVERAGE OVER.
 RAVRG SAME AS NAVRG BUT REAL.
 NUMB THE NUMBER OF AVERAGED ESTIMATES TO BE CALCULATED.
 NUMB1 NUMB(ABOVE)-1.
 SUM1-SUM8 RUNNING SUMS OF NAVRG ESTIMATES IN THE 'U' ARRAYS.
 A1-A8 TEMPORARY VARIABLES USED TO SAVE ESTIMATES.
 AU,AV,BU,BV INPUT ARRAYS(ALSO USED TO STORE CO- AND QUAD- SPECTRAL ESTIMATES).
 I,N INDEXES.
 'U' ARRAYS PRODUCTS OF ANTICLOCKWISE AND/OR CLOCKWISE COMPLEX FOURIER COEFFICIENTS AT LOCATIONS A AND/OR B. THESE ARE THE VALUES THAT ARE AVERAGED IN THIS SUBROUTINE.

COMPLEX AU(4097),AV(4097),BU(4097),BV(4097),
 UACBPB(50),UACNBN(50),UACPBN(50),UACNBP(50),
 UACPAP(50),UBCPBP(50),UACNAN(50),UBCNBN(50),
 SUM1,SUM2,SUM3,SUM4,SUM5,SUM6,SUM7,SUM8,
 A1,A2,A3,A4,A5,A6,A7,A8
 COMMON AU,AV,BU,BV,UACBPB,UACNBN,UACPBN,UACNBP,UACPAP,UBCPBP,
 UACNAN,UBCNBN

INITIALIZE VARIABLES.

NAVRG=2*NN+1
 NUMB=LENG-2*NN
 RAVRG=FLOAT(NAVRG)
 NUMB1=NUMB-1
 SUM1=CMPLX(0.,0.)
 SUM2=CMPLX(0.,0.)
 SUM3=CMPLX(0.,0.)
 SUM4=CMPLX(0.,0.)
 SUM5=CMPLX(0.,0.)

AVRG (CONT'D.)

```
SUM6=CMPLX(0.,0.)
SUM7=CMPLX(0.,0.)
SUM8=CMPLX(0.,0.)
```

```
C
C SUM FIRST NAVRG ESTIMATES.
C
```

```
DO 14 J=1,NAVRG
  SUM1=SUM1+UACBPB(J)
  SUM2=SUM2+UACNBN(J)
  SUM3=SUM3+UACPBN(J)
  SUM4=SUM4+UACNBP(J)
  SUM5=SUM5+UACPAP(J)
  SUM6=SUM6+UBCPBP(J)
  SUM7=SUM7+UACNAN(J)
  SUM8=SUM8+UBCNBN(J)
```

```
14 CONTINUE
  IF (NUMB1.LT.1) GO TO 16
```

```
C
C CALCULATE NUMB1 AVERAGED ESTIMATES.
C
```

```
DO 15 I=1,NUMB1
```

```
C
C SAVE ESTIMATES FOR LATER.
C
```

```
  A1=UACBPB(I)
  A2=UACNBN(I)
  A3=UACPBN(I)
  A4=UACNBP(I)
  A5=UACPAP(I)
  A6=UBCPBP(I)
  A7=UACNAN(I)
  A8=UBCNBN(I)
```

```
C
C DIVIDE SUMS BY NAVRG TO CALCULATE AVERAGES AND STORE IN THE 'U'
C ARRAYS.
C
```

```
  UACBPB(I)=SUM1/NAVRG
  UACNBN(I)=SUM2/NAVRG
  UACPBN(I)=SUM3/NAVRG
  UACNBP(I)=SUM4/NAVRG
  UACPAP(I)=SUM5/NAVRG
  UBCPBP(I)=SUM6/NAVRG
  UACNAN(I)=SUM7/NAVRG
  UBCNBN(I)=SUM8/NAVRG
```

```
C
C STORE CO- AND QUAD- SPECTRAL ESTIMATES IN THE INPUT ARRAYS.
C
```

```
  AU(NSTORE)=UACBPB(I)
  AV(NSTORE)=UACNBN(I)
  BU(NSTORE)=UACPBN(I)
  BV(NSTORE)=UACNBP(I)
  NSTORE=NSTORE+1
```

AVRG (CONT'D.)

DETERMINE SUMS FOR NEXT ESTIMATES.

```
SUM1=SUM1-A1+UACBPB(I+NAVRS)
SUM2=SUM2-A2+UACNBN(I+NAVRS)
SUM3=SUM3-A3+UACPBN(I+NAVRS)
SUM4=SUM4-A4+UACNBP(I+NAVRS)
SUM5=SUM5-A5+UACPAP(I+NAVRS)
SUM6=SUM6-A6+UBCPBP(I+NAVRS)
SUM7=SUM7-A7+UACNAN(I+NAVRS)
SUM8=SUM8-A8+UBCNBN(I+NAVRS)
```

15 CONTINUE

DIVIDE SUMS BY NAVRS TO CALCULATE AVERAGES FOR FINAL ESTIMATES IN THE 'U' ARRAYS.

```
16 UACBPB(NUMB)=SUM1/NAVRS
UACNBN(NUMB)=SUM2/NAVRS
UACPBN(NUMB)=SUM3/NAVRS
UACNBP(NUMB)=SUM4/NAVRS
UACPAP(NUMB)=SUM5/NAVRS
UBCPBP(NUMB)=SUM6/NAVRS
UACNAN(NUMB)=SUM7/NAVRS
UBCNBN(NUMB)=SUM8/NAVRS
```

STORE CO- AND QUAD- SPECTRAL ESTIMATES IN THE 'U' ARRAYS.

```
AU(NSTORE)=UACBPB(NUMB)
AV(NSTORE)=UACNBN(NUMB)
BU(NSTORE)=UACPBN(NUMB)
BV(NSTORE)=UACNBP(NUMB)
NSTORE=NSTORE+1
RETURN
END
```

IV. B.4 BAVRG

SUBROUTINE BAVRG(NUMBL,NN,NSTORE)

THIS SUBROUTINE PERFORMS BLOCK AVERAGING ON THE 'U' ARRAYS. AS EACH BLOCK OF $2 \times NN + 1$ ELEMENTS IS AVERAGED, THE NEW VALUES ARE MOVED TO THE TOP OF THE 'U' ARRAYS. THE CO- AND QUAD- SPECTRAL ESTIMATES ARE STORED IN THE INPUT ARRAYS(AU,AV,BU,BV).

VARIABLE LIST

NUMBL THE NUMBER OF BLOCKS TO BE AVERAGED.
 NN THE NUMBER OF ADJACENT FREQUENCY BANDS TO AVERAGE OVER.
 NSTORE THE NEXT POSITION IN THE INPUT ARRAYS TO BE FILLED WITH CO- AND QUAD- SPECTRAL ESTIMATES.
 NAVRG THE ACTUAL NUMBER OF POINTS PER BLOCK TO AVERAGE.
 AVRG SAME AS NAVRG BUT REAL.
 SUM1-SUM8 RUNNING SUMS OF THE ESTIMATES PER BLOCK OF THE 'U' ARRAYS.
 I,J,K INDEXES.
 AU,AV,BU,BV INPUT ARRAYS(ALSO USED TO STORE CO- AND QUAD- SPECTRAL ESTIMATES).
 'U' ARRAYS PRODUCTS OF ANTICLOCKWISE AND/OR CLOCKWISE COMPLEX FOURIER COEFFICIENTS AT LOCATIONS A AND/OR B. THESE ARE THE VALUES THAT ARE AVERAGED IN THIS SUBROUTINE.

COMPLEX AU(4097),AV(4097),BU(4097),BV(4097),
 UACBPB(50),UACNBN(50),UACPNB(50),UACNBP(50),
 UACPAP(50),UBCPBP(50),UACNAN(50),UBCNBN(50),
 SUM1,SUM2,SUM3,SUM4,SUM5,SUM6,SUM7,SUM8
 COMMON AU,AV,BU,BV,UACBPB,UACNBN,UACPNB,UACNBP,UACPAP,UBCPBP,
 UACNAN,UBCNBN
 NAVRG=2*NN+1
 AVRG=FLOAT(NAVRG)

AVERAGE NUMBL BLOCKS.

DO 10 I=1,NUMBL

INITIALIZE SUMS TO ZERO.

SUM1=CMPLX(0.,0.)
 SUM2=CMPLX(0.,0.)
 SUM3=CMPLX(0.,0.)
 SUM4=CMPLX(0.,0.)
 SUM5=CMPLX(0.,0.)
 SUM6=CMPLX(0.,0.)
 SUM7=CMPLX(0.,0.)
 SUM8=CMPLX(0.,0.)

SUM NAVRG ESTIMATES.

DO 11 J=1,NAVRG
 K=(I-1)*NAVRG+J

BAVRG (CONT'D.)

```

SUM1=SUM1+UACPBP(K)
SUM2=SUM2+UACNBN(K)
SUM3=SUM3+UACPBN(K)
SUM4=SUM4+UACNBP(K)
SUM5=SUM5+UACPAP(K)
SUM6=SUM6+UBCPBP(K)
SUM7=SUM7+UACNAN(K)
SUM8=SUM8+UBCNBN(K)
11
C
C DIVIDE BY AVRG TO CALCULATE AVERAGES AND STORE IN 'U' ARRAYS.
C
UACPBP(I)=SUM1/AVRG
UACNBN(I)=SUM2/AVRG
UACPBN(I)=SUM3/AVRG
UACNBP(I)=SUM4/AVRG
UACPAP(I)=SUM5/AVRG
UBCPBP(I)=SUM6/AVRG
UACNAN(I)=SUM7/AVRG
UBCNBN(I)=SUM8/AVRG
C
C STORE CO- AND QUAD- SPECTRAL ESTIMATES IN THE INPUT ARRAYS.
C
AU(NSTORE)=UACPBP(I)
AV(NSTORE)=UACNBN(I)
BU(NSTORE)=UACPBN(I)
BV(NSTORE)=UACNBP(I)
10 NSTORE=NSTORE+1
RETURN
END

```

IV. B.5 PHACOH

SUBROUTINE PHACOH(I)

THIS SUBROUTINE CALCULATES THE SQUARED COHERENCIES AND PHASES BETWEEN THE PAIRS OF POLARIZED CONSTITUENTS AT LOCATIONS A+B GIVEN A PARTICULAR LOCATION IN THE 'U' ARRAYS. FIRST, COMPLEX CROSS SPECTRAL ESTIMATES FOR THE PAIRS OF POLARIZED CONSTITUENTS AT DIFFERENT LOCATIONS ARE CALCULATED. THEN, THE SQUARED COHERENCIES AND PHASES ARE EASILY CALCULATED FROM THE REAL AND IMAGINARY COMPONENTS OF THESE ESTIMATES.

VARIABLE LIST

I THE POSITION OF THE VALUES IN THE 'U' ARRAYS FOR WHICH SQUARED COHERENCIES AND PHASES ARE CALCULATED.
 GABPP, GABNN, CROSS SPECTRAL ESTIMATES FOR PAIRS OF POLARIZED CON-
 GABPN, GABNP STITUENTS AT LOCATIONS A AND B.
 RABPP, RABNN, THE REAL COMPONENTS OF THE COMPLEX CROSS SPECTRAL
 RABNP, RABPN ESTIMATES.
 AIABPP, AIABNN, THE IMAGINARY COMPONENTS OF THE COMPLEX CROSS SPECTRAL
 AIABNP, AIABPN ESTIMATES.
 COABPP, COABNN, THE SQUARED COHERENCIES BETWEEN THE PAIRS OF POLA-
 COABNP, COABPN RIZED CONSTITUENTS AT LOCATIONS A AND B.
 PHABPP, PHABNN, THE PHASES FOR THE PAIRS OF POLARIZED CONSTITUENTS
 PHABNP, PHABPN AT LOCATIONS A AND B.
 'U' ARRAYS THE PRODUCTS OF ANTICLOCKWISE AND/OR CLOCKWISE COMPLEX
 FOURIER COEFFICIENTS AT LOCATIONS A AND/OR B.

COMPLEX AU(4097), AV(4097), BU(4097), BV(4097),
 UACPBP(50), UACNBN(50), UACPNB(50), UACNBP(50),
 UACPAP(50), UBCPBP(50), UACNAN(50), UBCNBN(50),
 GABPP, GABNN, GABPN, GABNP
 COMMON AU, AV, BU, BV, UACPBP, UACNBN, UACPNB, UACNBP, UACPAP, UBCPBP,
 UACNAN, UBCNBN,
 PHABPP, PHABNN, PHABPN, PHABNP, COABPP, COABNN, COABPN, COABNP

CALCULATE THE CROSS SPECTRAL ESTIMATES FOR THE PAIRS OF POLARIZED CONSTITUENTS AT LOCATIONS A AND B.

RAD=180./3.1415927
 GABPP=UACPBP(I)/CSQRT(UACPAP(I)+UBCPBP(I))
 GABNN=UACNBN(I)/CSQRT(UACNAN(I)+UBCNBN(I))
 GABPN=UACPNB(I)/CSQRT(UACPAP(I)+UBCNBN(I))
 GABNP=UACNBP(I)/CSQRT(UBCPBP(I)+UACNAN(I))

BREAK THE COMPLEX CROSS SPECTRAL ESTIMATES INTO THEIR REAL AND IMAGINARY COMPONENTS.

PHACOH (CONT'D.)

```

RABPP=REAL (GABPP)
AIABPP=AIMAG (GABPP)
RABNN=REAL (GABNN)
AIABNN=AIMAG (GABNN)
RABPN=REAL (GABPN)
AIABPN=AIMAG (GABPN)
RABNP=REAL (GABNP)
AIABNP=AIMAG (GABNP)

```

CALCULATE THE SQUARED COHERENCIES AND PHASES.

```
PHABPP=ATAN2 (AIABPP,RABPP) ◀RAD
COABPP=RABPP◀◀2+AIABPP◀◀2
PHABNN=ATAN2 (AIABNN,RABNN) ◀RAD
COABNN=RABNN◀◀2+AIABNN◀◀2
PHABPN=ATAN2 (AIABPN,RABPN) ◀RAD
COABPN=RABPN◀◀2+AIABPN◀◀2
PHABNP=ATAN2 (AIABNP,RABNP) ◀RAD
COABNP=RABNP◀◀2+AIABNP◀◀2
RETURN
END
```

IV. B.6 SL

FUNCTION SL(NDOF)

THIS FUNCTION DETERMINES A 90% SIGNIFICANCE LEVEL FOR COHERENCIES GIVEN THE NUMBER OF DEGREES OF FREEDOM. THESE VALUES WERE CALCULATED FROM 'TABLE OF THE DISTRIBUTION OF THE COEFFICIENT OF COHERENCE FOR STATIONARY BIVARIATE GAUSSIAN PROCESSES' BY AMOS AND KOOPMANS, 1963. THE RELATIONSHIP FOR VALUES ASSOCIATED WITH NDOF ≥ 12 WAS FOUND TO BE WITHIN 3% ACCURACY.

VARIABLE LIST

NDOF	THE NUMBER OF DEGREES OF FREEDOM.
SL	SIGNIFICANCE LEVEL.
N	POINTER.

```
IF(NDOF.GT.11)GO TO 200
N=NDOF+1
GO TO(2,2,2,3,4,5,6,7,8,9,10,11),N
2 SL=1.
  RETURN
3 SL=.9900
  RETURN
4 SL=.9000
  RETURN
5 SL=.7945
  RETURN
6 SL=.6838
  RETURN
7 SL=.6019
  RETURN
8 SL=.5359
  RETURN
9 SL=.4822
  RETURN
10 SL=.4377
  RETURN
11 SL=.4005
  RETURN
200 SL=4.5578/FLOAT(NDOF)
  RETURN
  END
```

APPENDIX B

SOURCE LISTING - 'JAY.RUNALL'

APPENDIX B - 'JAY. RUNALL' LISTING

@FREE 7.
@FREE 8.
@FREE 9.
@FREE 10.
@FREE 11.
@FREE 12.
@FREE 13.
@FREE 14.
@FREE 15.
@FREE 17.
@FREE 18.
@FREE 19.
@FREE 20.
@FREE 21.
@FREE 22.
@FREE 23.
@DELETE 7.
@DELETE 8.
@DELETE 9.
@DELETE 10.
@DELETE 11.
@DELETE 12.
@DELETE 13.
@DELETE 14.
@DELETE 15.
@DELETE 17.
@DELETE 18.
@DELETE 19.
@DELETE 20.
@DELETE 21.
@DELETE 22.
@DELETE 23.
@ASG.CP 11..F40/0//400
@ASG.CP 12..F40/0//400
@ASG.CP 13..F40/0//400
@ASG.CP 14..F40/0//10
@ASG.CP 15..F40/0//10
@ASG.CP 17..F40/0//200
@ASG.CP 18..F40/0//200
@ASG.CP 19..F40/0//200
@ASG.CP 20..F40/0//200
@ASG.CP 21..F40/0//500
@ASG.CP 22..F40/0//400
@ASG.CP 23..F40/0//400

'JAY.RUNALL' (CONT'D.)

```
@USE 4.FEBFILE.
@XQT JAY.GET
VACM      1      4    500 1000      1      0
          2      2      7      1      8
@XQT JAY.GET
VACM      2      5    300 1000      1      0
          2      2      9      1     10
@XQT JAY.FOURCO
          1024      1      0
@XQT JAY.RCSPEC
          1.      0      0      2      3      0
@XQT JAY.RSPEC
          1.      0      0      1      3      5
          1      1    100      5    900      1 1500      3 2000      2
          1.      0      0      2      4      0
@ASG.CP PLOT1.F40/0/300
@ASG.T 16
@DATA.I 16
@ADD.D JAY.ICAT
@END
@ASG.T 24
@DATA.I 24
@ADD.D JAY.USER
@END
@ASG.T 25
@DATA.I 25
@ADD.D JAY.ICAT
@END
@XQT JAY.PLOT1/RUN
@COPY PSLSTS.PLOT1
```

APPENDIX C

SOURCE LISTING - 'MAP' DATASET

APPENDIX C - 'MAP' DATASET LISTINGS

JAY.GETMAP

```
IN JAY.GET  
LIB SYSTEMS+SUBLIB.  
END
```

JAY.FOURCMAP

```
IN JAY.FOURCD  
IN JAY.FFT  
IN JAY.FFTWIND  
IN JAY.FFTPWHITE
```

JAY.RSMAP

```
IN JAY.RSMAIN  
IN JAY.RASCALC  
IN JAY.SWAP  
IN JAY.RASAVRG  
IN JAY.RASBAVRG  
IN JAY.RASSTAT  
IN JAY.RSL  
IN JAY.CONINT
```

JAY.RCSMAP

```
IN JAY.RXMAIN  
IN JAY.RCSCALC  
IN JAY.RCSSWAP  
IN JAY.RCSAVRG  
IN JAY.RCSBAVRG  
IN JAY.RCSPHCD  
IN JAY.RSL
```

APPENDIX D

SOURCE LISTING - 'JAY. USER'

APPENDIX D - 'JAY.USER' LISTING

0													
3	3	1	1	0	0	0	0	0	0	0	0	0	0
1	500	0	0	0	0	0	0	0	0	0	0	0	0
0	500	0	0	0	0	0	0	0	0	0	0	0	0
1	500	0	0	0	0	0	0	0	0	0	0	0	0
1	500	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

APPENDIX E - SOURCE LISTINGS

PLOTTING PROGRAM

- I. MAIN PROGRAM - BIGPLT
- II. SUBPROGRAMS
 - A. ABUTMS
 - B. ALFSET
 - C. BOTTOM
 - D. CONINT
 - E. DECADE
 - F. DOWN
 - G. LABEL
 - H. MINMAX
 - I. MNPL
 - J. SETMES
 - K. SETUP
 - L. TOP
 - M. UP

I. BIGPLT

USER CONTROL:

USER CONTROL OF THE GRAPHICS OUTPUT IS ACCOMPLISHED VIA TWO DATA SETS WHOSE FORTRAN REFERENCE NUMBERS ARE CALLED KU AND KD IN THE MAIN PROGRAM. KU CONTAINS CONTROL DATA AND KD CONTAINS USER SUPPLIED LABELS.

KU (ALL RECORDS ARE 1515 FORMAT)

FIRST CARD - ONE FIELD READ (ENTERS AS KLAB)

FIELD CONTAINS

1	=>	USER SUPPLIED LABELS ARE TO BE READ INTO ARRAY ICAT FROM KD
0	=>	NO USER LABELS SUPPLIED

SECOND CARD - FOUR FIELDS READ (ENTER AS NPL(1)-(4))

THE FOUR FIELDS CORRESPOND IN ORDER TO THE FOUR PLOTS

FIELD CONTAINS:

0	=>	CORRESPONDING PLOT NOT DRAWN
NEG. NUMBER	=>	CATALOG OF USER SUPPLIED LABELS TO BE USED FOR THE CORRESPONDING PLOT
POS. NUMBER	=>	DEFAULT CATALOG TO BE USED FOR THE CORRESPONDING PLOT
NUMBER WHOSE ABSOLUTE VALUE IS		
1	=>	X-AXIS LOG Y-AXIS LINEAR
2	=>	X-AXIS LINEAR Y-AXIS LINEAR
3	=>	X-AXIS LOG Y-AXIS LOG

REMAINING CARDS - NINE FIELDS READ (ENTERED AS NBOT,NPTS, AND IT(1)-(7))

THE REMAINING CARDS CORRESPOND IN ORDER TO THE PLOTS WHICH ARE TO BE DRAWN

FIELD #1 CONTAINS:

0	=>	MOORING DATA NOT INCLUDED ON PLOT
1	=>	MOORING DATA APPEARS ON PLOT

FIELD #2 CONTAINS THE NUMBER OF DATA POINTS WHICH THE

BIGPLT (CONT'D.)

PLOT WILL CONTAIN. IF THIS FIELD IS ZERO OR BLANK, THE PROGRAM WILL ASSUME ALL POINTS ARE TO BE INCLUDED.

FIELDS #3 - 9 WILL ONLY BE READ IF USER SUPPLIED LABELS ARE TO BE USED IN THE CORRESPONDING PLOT. THE PROGRAM READS THESE VALUES INTO ARRAY IT. SEE DOCUMENTATION OF ROUTINES TOP AND SETUP FOR FURTHER EXPLANATION.

KD (ALL RECORDS ARE 12A6 FORMAT)

EACH RECORD SHOULD CONTAIN A CHARACTER STRING NOT TO EXCEED 71 CHARACTERS FOLLOWED BY "\$". THESE STRINGS WILL BE WRITTEN USING THE ALPHABET CONVENTIONS OF SUBROUTINE ALFSET. I.E., INSTRUCTIONAL STRING OPTIONS ARE AVAILABLE.

NOTE: KD SHOULD CONTAIN NO MORE THAN 15 RECORDS

EXAMPLE:

SUPPOSE DATA SET KU CONTAINS THE FOLLOWING CARDS

```

1
3      0      0      -1
1 3500
0 4098      1      .2      3      4      2      5

```

THE PROGRAM WOULD:

- READ THE LABELS FROM KD INTO ICAT
- DRAW THE FIRST PLOT USING DEFAULT TITLE AND LABELS ON A LOG-LOG AXIS.
- USE ONLY THE FIRST 3500 FREQUENCY VALUES AND INCLUDE MOORING DATA AT THE BOTTOM OF THE FIRST PLOT
- DRAW THE FOURTH PLOT ON A LOG-LINEAR AXIS USING THE FIRST THREE RECORDS FROM KD AS THE LINES OF TITLE, THE FOURTH RECORD AS X-AXIS LABEL, AND THE SECOND AND FIFTH RECORDS AS Y-AXIS LABELS
- THE FOURTH PLOT WOULD USE 4098 POINTS AND WOULD NOT CONTAIN ANY MOORING INFORMATION

```

REAL X(4100),Y(4100),ZX(2),ZY(2)
INTEGER LAB(12),ICAT(12,30,2),IPAK(120),NPL(10),IT(15)
COMMON LAB,ICAT,IPAK,X,Y

```

SIZING PARAMETERS (ALL DISTANCES ARE IN INCHES)

- P TOTAL HORIZONTAL SIZE OF THE SPACE WHICH WILL BE DEDICATED TO AXIS FRAMES.
- Q TOTAL VERTICAL SIZE OF THE SPACE WHICH WILL BE DEDICATED TO AXIS FRAME(S). THIS DOES NOT INCLUDE SPACE FOR TITLE OR STORY.

BIGPLT (CONT'D.)

[illegible]

P=6.
D=6.5
G=.5
XPOS=1.5
YPOS=3.0

[illegible]

KR=12
KRX=11
KCI=14
KF=7
KF1=KF
KU=24
KD=25
KDD=16
KTEST=0

```

CC      LOAD DEFAULT LABELS INTO ARRAY ICAT
CC
CC      DO 40 J=1,30
CC      READ(KDD,55) (ICAT(I,J,1),I=1,12)
40      CONTINUE
C
C      DO 30 J=16,30
C      READ(KDD,55) (ICAT(I,J,2),I=1,12)
30      CONTINUE
CC
CC      DOES USER WISH TO SUPPLY LABELS ???
CC
C      READ(KU,99) KLAB
C      IF(KLAB.EQ.0) GO TO 60

```

•

[illegible]

BIGPLT (CONT'D.)

First Plot - flow chart

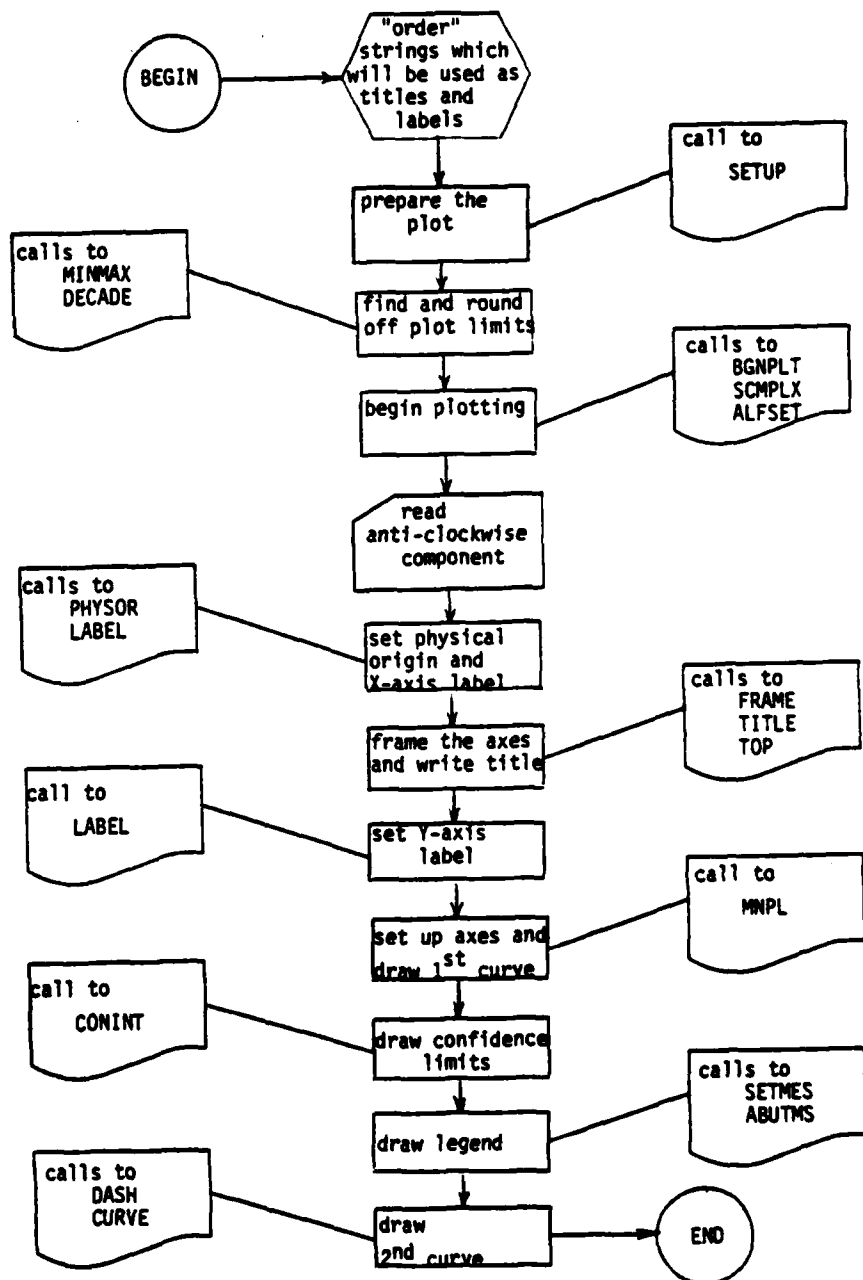


Figure E-1

BIGPLT (CONT'D.)

```

READ(KR,90) (X(J),J=1,NPTS)
REWIND KR
READ(KR,90) DUM
READ(KR,91) (Y(J),J=1,NPTS)
XMN=X(1)
XMX=X(NPTS)
CALL MINMAX(TMN,TMX,Y(1),Y(1),NPTS)
REWIND KR
READ(KR,90) DUM
READ(KR,92) (Y(J),J=1,NPTS)
CALL MINMAX(YMN,YMX,TMN,TMX,NPTS)
CALL DECADE(YMN,YMX)
YMN=YMN/100.
DX=YMN*10.

```

```

C
C BEGIN PLOTTING
C

```

```

CALL BGNPL(0)
CALL SCMPX
CALL ALFSET

```

```

C
C READ ANTI-CLOCKWISE COMPONENT
C

```

```

REWIND KR
READ(KR,90) DUM
READ(KR,91) (Y(J),J=1,NPTS)

```

```

C
C SET PHYSICAL ORIGIN AND X-AXIS LABEL
C

```

```

CALL PHYSOR(XPOS,YPOS)
CALL LABEL(NCAT,IT(4))

```

```

C
C FRAME THE AXES AND WRITE TITLE
C

```

```

CALL TITLE(1H,1,LAB,100,0,0,P,Q)
CALL FRAME
CALL TOP(NCAT,IT)

```

```

C
C SET Y-AXIS LABEL
C

```

```

CALL LABEL(NCAT,IT(5))
STP=(YMX-YMN)/5.

```

```

C
C SET UP AXES AND DRAW FIRST CURVE
C

```

```

CALL MNPL(NOPT,XMN,XMX,YMN,YMX,STP,NPTS,P,QQ)

```

```

C
C DRAW CONFIDENCE LIMITS
C

```

```

CALL CONINT(KCI,X(NPTS),DX)
CALL BOTTOM(NBOT,0.0,KF)
CALL UP(NBOT,YPOS,Q)

```

BIGPLT (CONT'D.)

DRAW LEGEND

```

QM=QM-.15
CALL SETMES(4,3.,QM)
CALL ABUTMS(19)
QM=QM-.25
CALL SETMES(5,3.,QM)
CALL ABUTMS(17)

```

DRAW SECOND CURVE

```
REWIND KR
READ(KR,90) DUM
READ(KR,92) (Y(J),J=1,NPTS)
CALL DASH
CALL CURVE(X,Y,NPTS,0)
D=1.
CALL ENDPL(1)
```

SECOND PLOT (see figure E-2 on next page for flow chart)

THIS PROGRAM SEGMENT PLOTS THE SAME INFORMATION AS THE FIRST PLOT. THE FORMAT IS CHANGED, HOWEVER, SO THAT THE CLOCKWISE AND ANTI-CLOCKWISE COMPONENTS ARE PLOTTED ON SEPARATE COORDINATE FRAMES.

```
200 IF (NPL(2).EQ.0) GO TO 300
```

ORDER STRINGS WHICH WILL BE USED AS TITLE AND LABELS

```
SET IT
IT(1)=6
IT(2)=3
IT(3)=0
IT(4)=1
IT(5)=2
IT(6)=2
```

```
REWIND KR
READ(KR,90) NPTS
NPTS=NPTS-1
CALL SETUP(NPL(2),KU,NCAT,NBOT,NOPT,NPTS,IT)
```

CALL DOWN (NBOT, YPOS, Q)

$$DD = (D - 5) / 2.$$

$$DDD = DD + 5$$

FIND AND ROUND OFF PLOT LIMITS

BIGPLT (CONT'D.)

Second Plot flow chart

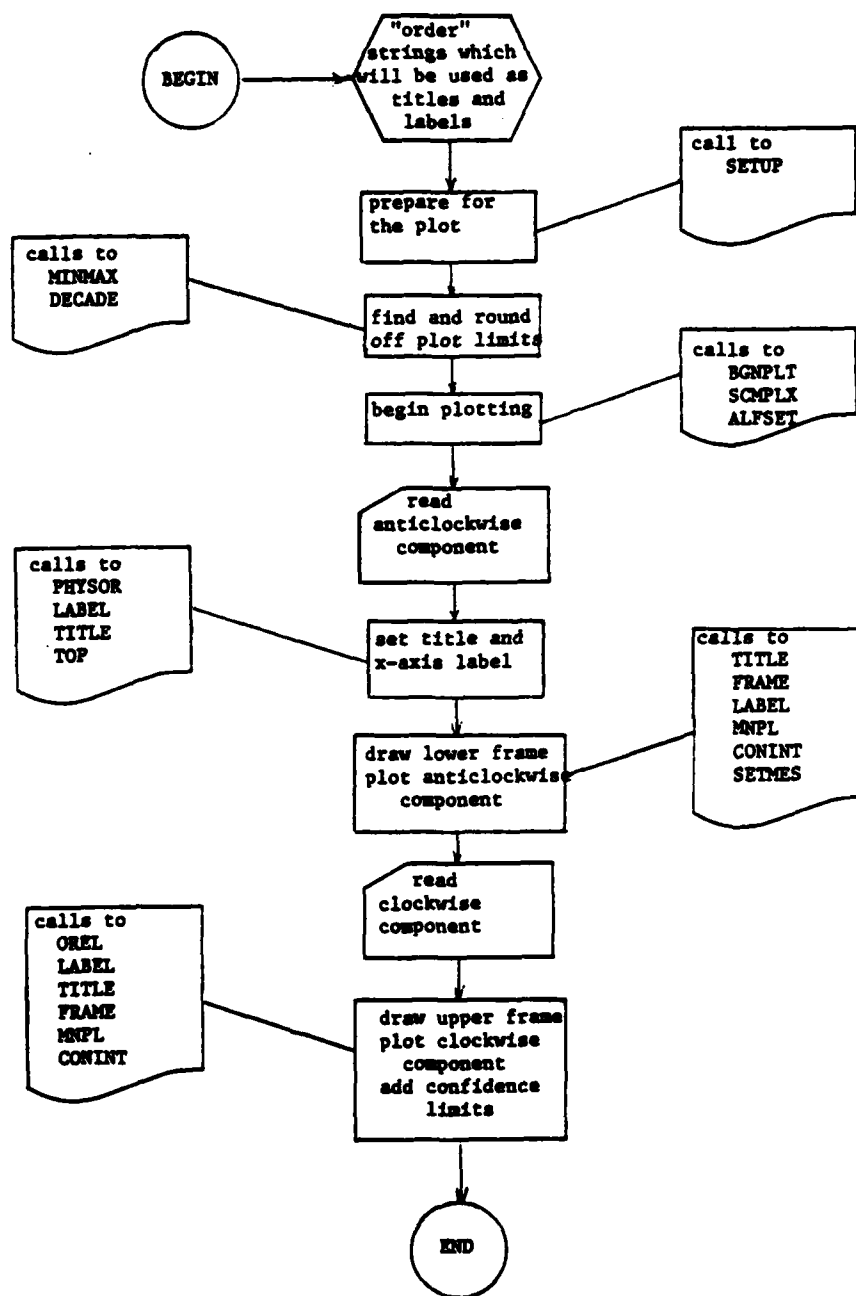


Figure E-2

BIGPLT (CONT'D.)

```

READ(KR,90) (X(J),J=1,NPTS)
REWIND KR
READ(KR,90) DUM
READ(KR,91) (Y(J),J=1,NPTS)
XMN=X(1)
XMX=X(NPTS)
CALL MINMAX(TMN,TMX,Y(1),Y(1),NPTS)
REWIND KR
READ(KR,90) DUM
READ(KR,92) (Y(J),J=1,NPTS)
CALL MINMAX(YMN,YMX,TMN,TMX,NPTS)
CALL DECADE(YMN,YMX)
DX=YMN*10.

```

BEGIN PLOTTING

```

CALL BGNPL(0)
CALL SCMPX
CALL ALFSET

```

READ ANTI-CLOCKWISE COMPONENT

```

REWIND KR
READ(KR,90) DUM
READ(KR,91) (Y(J),J=1,NPTS)

```

DRAW LOWER FRAME AND PLOT ANTICLOCKWISE COMPONENT

```

CALL PHYSOP(XPOS,YPOS)
CALL TITLE(1H ,1,0,0,0,0,P,Q)
CALL TOP(NCAT,IT)
CALL ENDGR(2)

```

```

CALL LABEL(NCAT,IT(4))
CALL TITLE(1H ,1,LAB,100,0,0,P,QQ)
CALL FRAME
CALL LABEL(NCAT,IT(5))
NST=(YMX-YMN)/8
YST=NST

```

```

CALL MNPL(NOPT,XMN,XMX,YMN,YMX,YST,NPTS,P,QQ)
CALL BOTTOM(NBOT,0,0,KF)
CALL UP(NBOT,YPOS,Q)
CALL CONINT(KCI,X(NPTS),DX)

```

```

CALL SETMES(5,.5,.1)
CALL ENDGR(2)

```

READ CLOCKWISE COMPONENT

```

REWIND KR
READ(KR,90) DUM
READ(KR,92) (Y(J),J=1,NPTS)

```

BIGPLT (CONT'D.)

CALL DOWN (NBOT.YPOS.Q)

BIGPLT (CONT'D.)

Third Plot flow chart

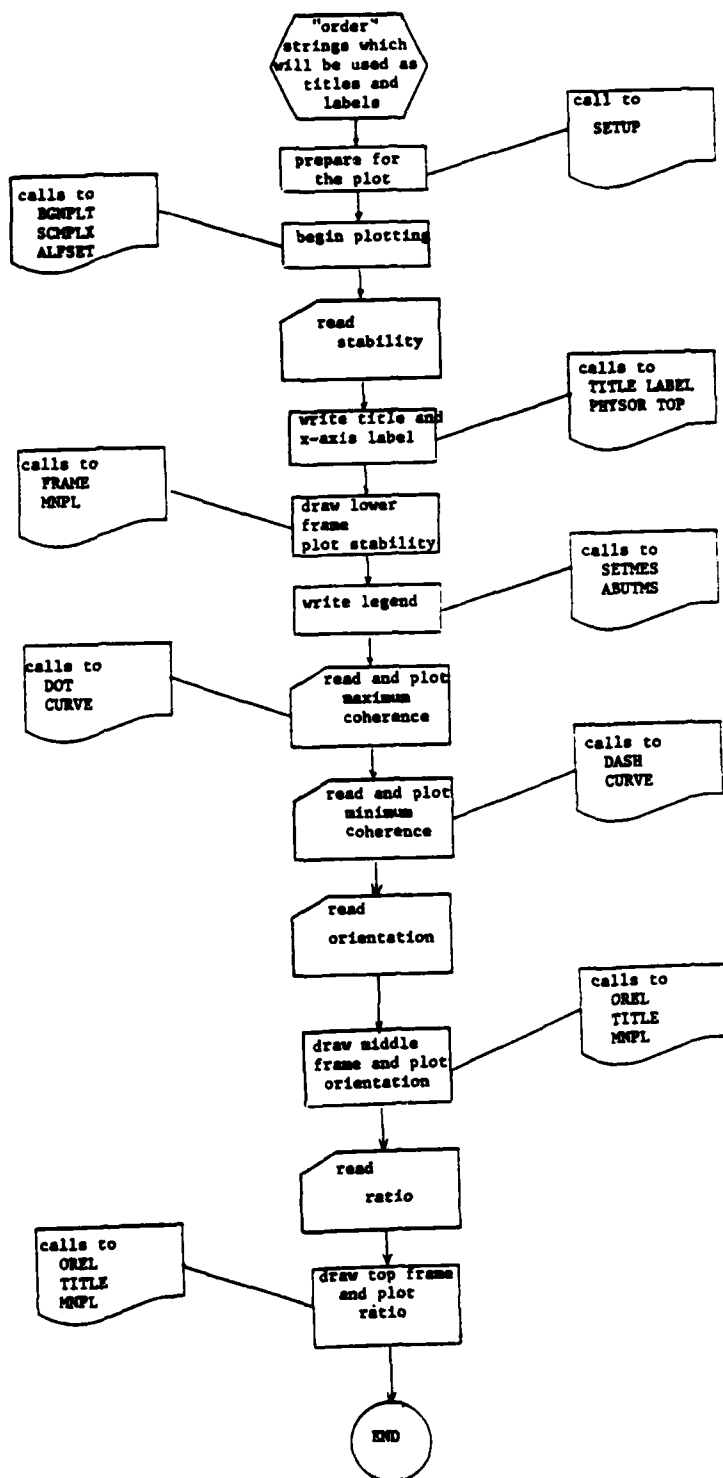


Figure E-3

BIGPLT (CONT'D.)

```

00=(0-2.5)/3.
000=00+5
C
READ(KR,90) (X(J),J=1,NPTS)
XMN=X(1)
XMX=X(NPTS)
C
ZX(1)=XMN
ZX(2)=XMX
ZY(1)=0.
ZY(2)=0.
CC
CC BEGIN PLOTTING
CC
CALL BGNPL(0)
CALL SCMPX
CALL ALFSET
CC
CC READ STABILITY
CC
C STABILITY
REWIND KR
READ(KR,90) DUM
READ(KR,94) (Y(J),J=1,NPTS)
CC
CC WRITE TITLE AND X-AXIS LABEL
CC
CALL PHYSOR(XPOS,YPOS)
CALL TITLE(1H,1,0,0,0,0,P,0)
CALL TOP(NCAT,IT)
CALL ENDGR(3)
CC
CC DRAW LOWER FRAME AND PLOT STABILITY
CC
CALL LABEL(NCAT,IT(4))
CALL TITLE(1H,1,LAB,100,0,0,P,00)
CALL YINTX
CALL FRAME
CALL LABEL(NCAT,IT(5))
CALL MNPL(NOPT,XMN,XMX,0.,1.,.25,NPTS,P,00)
CALL BOTTOM(NBOT,0.0,KF)
CALL UP(NBOT,YPOS,0)
CC
CC WRITE THE LEGEND
CC
XP=.5
YP=00-.25
CALL SETMES(12,XP,YP)
CALL ABUTMS(17)
CALL ABUTMS(16)
CALL ABUTMS(21)
CALL ABUTMS(11)

```

BIGPLT (CONT'D.)

```
CALL ABUTMS(19)
CALL ABUTMS(16)
CALL ABUTMS(21)
CALL ABUTMS(10)
CALL ABUTMS(18)
```

```
CC
CC READ AND PLOT MAXIMUM COHERENCE
CC
```

```
C   MAX COHERENCE
    REWIND KR
    READ(KR,90) DUM
    READ(KR,97) (Y(J),J=1,NPTS)
    CALL DOT
    CALL CURVE(X,Y,NPTS,0)
```

```
CC
CC READ AND PLOT MINIMUM COHERENCE
CC
```

```
C   MIN COHERENCE
    REWIND KR
    READ(KR,90) DUM
    READ(KR,96) (Y(J),J=1,NPTS)
    CALL DASH
    CALL CURVE(X,Y,NPTS,0)
    CALL RESET('DASH')
    CALL ENDGR(3)
```

```
CC
CC READ ORIENTATION
CC
```

```
C   ORIENTATION
    REWIND KR
    READ(KR,90) DUM
    READ(KR,93) (Y(J),J=1,NPTS)
```

```
CC
CC DRAW MIDDLE FRAME AND PLOT ORIENTATION
CC
```

```
CALL DREL(0.,999)
CALL LABEL(1,16)
CALL TITLE(1H,1,LAB,100,0,0,P,99)
CALL FRAME
CALL LABEL(NCAT,IT(6))
CALL MNPL(NOPT,XMN,XX,-180.,180.,90.,NPTS,P,99)
CALL CURVE(ZX,ZY,2,0)
CALL SETMES(14,XP,YP)
CALL ENDGR(3)
```

```
CC
CC READ RATIO
CC
```

```
C   RATIO
    REWIND KR
    READ(KR,90) DUM
    READ(KR,95) (Y(J),J=1,NPTS)
```

BIGPLT (CONT'D.)

```
CC DRAW TOP FRAME AND PLOT RATIO
CC CALL DREL(0.,QQQ)
CC CALL LABEL(1,16)
CC CALL TITLE(1H ,1,LAB,100,0,0,P,QQ)
CC CALL FRAME
CC CALL LABEL(NCAT,IT(7))
C CALL MNPL(NOPT,XMN,XXM,-1.,1.,.5,NPTS,P,QQ)
CC CALL CURVE(ZX,ZY,2,0)
CC CALL SETMES(27,XP,YP)
CC CALL ENDGR(3)
CC CALL ENDDL(3)
C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><>
C FOURTH PLOT      (see figure E-4 on next page for flow chart)
C
C THIS PROGRAM SEGMENT PLOTS ROTARY CROSS SPECTRAL DATA
C ON TWO COORDINATE FRAMES:
C - THE UPPER FRAME CONTAINS PLOTS OF THE VARIOUS COHE-
C RENCIES (SQUARED) OF CLOCKWISE AND ANTI-CLOCKWISE
C COMPONENTS AT THE TWO MOORINGS AS FUNCTIONS OF
C FREQUENCY.
C - THE LOWER FRAME CONTAINS PLOTS OF THE CORRESPONDING
C PHASES.
C
C NOTATIONAL CONVENTIONS:
C (+ + ) =>          CLOCKWISE AT A           CLOCKWISE AT B
C (+ - ) =>          CLOCKWISE AT A   ANTICLOCKWISE AT B
C (- - ) =>          ANTICLOCKWISE AT A    ANTICLOCKWISE AT B
C (- + ) =>          ANTICLOCKWISE AT A     CLOCKWISE AT B
C<><><><><><><><><><><><><><><><><><><><><><><><><><><><><>
400 IF(KTEST.EQ.1) GO TO 401
KTEST=1
KR=13
KCI=15
KF=9
KF2=KF
GO TO 100
401 IF(NPL(4).EQ.0) GO TO 500
CC ORDER STRINGS WHICH WILL BE USED AS TITLE AND LABELS
CC
C SET IT
IT(1)=13
IT(2)=15
IT(3)=28
IT(4)=1
IT(5)=28
IT(6)=15
CC PREPARE FOR PLOT
```

BIGPLT (CONT'D.)

Fourth Plot flow chart

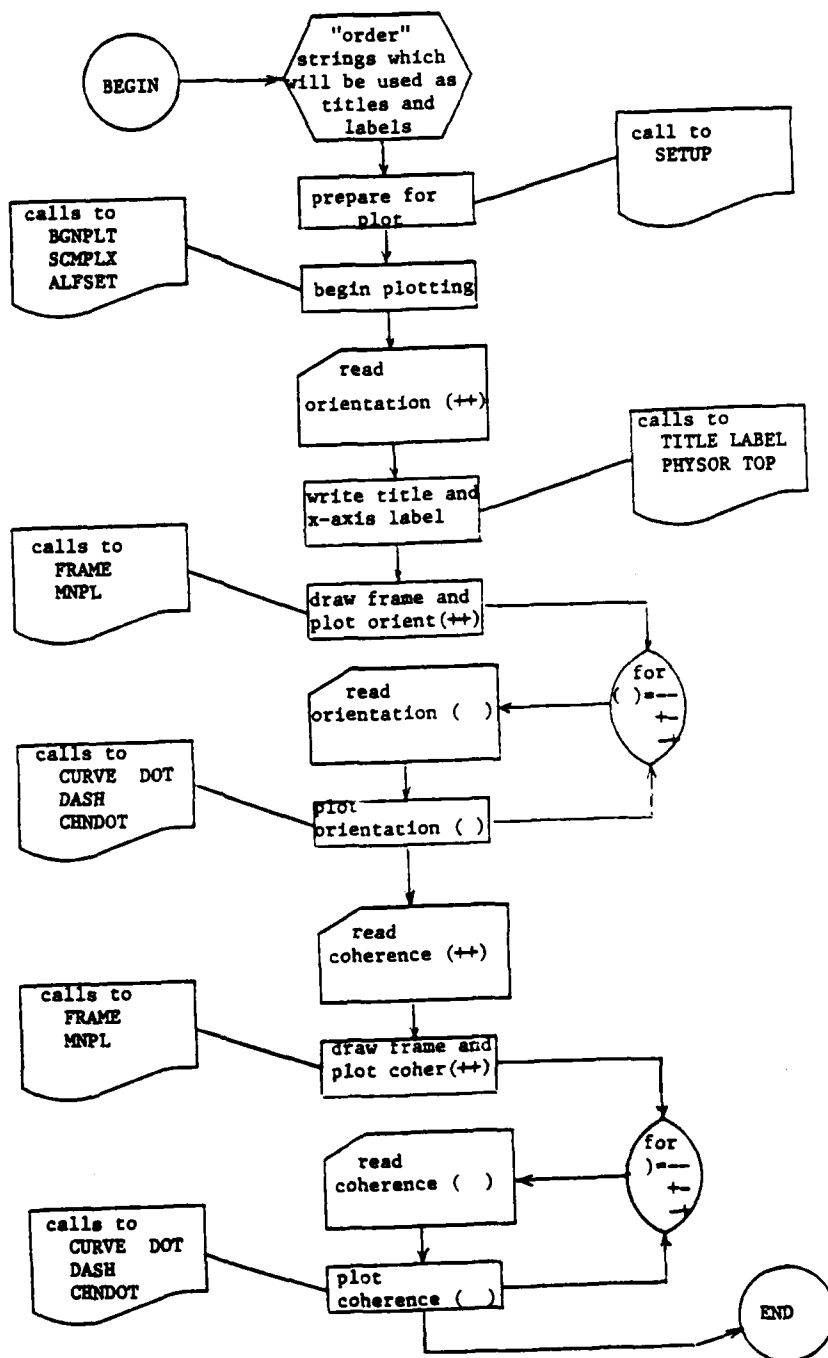


Figure E-4

BIGPLT (CONT'D)

```
REWIND KRX
READ(KRX,90) NPTS
NPTS=NPTS-1
CALL SETUP(NPL(4),KU,NCAT,NBOT,NOPT,NPTS,IT)
```

```
CALL DOWN(NBOT,YPOS,0)
```

```
READ(KRX,90) (X(J),J=1,NPTS)
XMN=X(1)
XMX=X(NPTS)
```

```
ZX(1)=XMN
ZX(2)=XMX
ZY(1)=0.
ZY(2)=0.
QQ=(Q-6)/2.
QQQ=QQ+6
```

```
BEGIN PLOTTING
```

```
CALL BGNPL(0)
CALL SCMPX
CALL ALFSET
```

```
READ ORIENTATION(++)
```

```
ORIENTATION (++)
REWIND KRX
READ(KRX,90) DUM
READ(KRX,92) (Y(J),J=1,NPTS)
```

```
WRITE TITLE AND X-AXIS LABEL
```

```
CALL PHYSQR(XPOS,YPOS)
CALL YINTX
CALL TITLE(1H,1,0,0,0,0,P,Q)
CALL TOP(NCAT,IT)
CALL ENDGR(4)
```

```
DRAW FRAME AND PLOT ORIENTATION (++)
```

```
CALL LABEL(NCAT,IT(4))
CALL TITLE(1H,1,LAB,100,0,0,P,QQ)
CALL FRAME
CALL LABEL(NCAT,IT(5))
CALL MNPL(NOPT,XMN,XMX,-180.,180.,90.,NPTS,P,QQ)
CALL CURVE(ZX,ZY,2,0)
CALL HEIGHT(.07)
CALL MESSAG(9HMODRING A,9,-.3,-.67)
CALL RESET('HEIGHT')
CALL BOTTOM(NBOT,0,0,KF1)
```

BIGPLT (CONT'D.)

```
CALL HEIGHT(.07)
CALL MESSAGE(9HMOORING B.9.-.3.-1.87)
CALL RESET('HEIGHT')
CALL BOTTOM(NBOT,-1.2,KF2)
CALL UP(NBOT,YPOS.9)
```

```
CC
CC WRITE LEGEND
CC
```

```
YP=QQ-.25
CALL SETMES(22,.5*YP)
CALL ABUTMS(23)
CALL ABUTMS(17)
CALL SETMES(22,2.5*YP)
CALL ABUTMS(24)
CALL ABUTMS(18)
YP=YP-.25
CALL SETMES(22,.5*YP)
CALL ABUTMS(25)
CALL ABUTMS(19)
CALL SETMES(22,2.5*YP)
CALL ABUTMS(26)
CALL ABUTMS(20)
```

```
CC
CC READ AND PLOT REMAINING ORIENTATION - (==), (==), (==)
CC
```

```
ORIENTATION (==)
REWIND KRX
READ(KRX,90) DUM
READ(KRX,94) (Y(J),J=1,NPTS)
CALL DOT
CALL CURVE(X,Y,NPTS,0)
```

```
ORIENTATION (==)
REWIND KRX
READ(KRX,90) DUM
READ(KRX,96) (Y(J),J=1,NPTS)
CALL DASH
CALL CURVE(X,Y,NPTS,0)
```

```
ORIENTATION (==)
REWIND KRX
READ(KRX,90) DUM
READ(KRX,98) (Y(J),J=1,NPTS)
CALL CHNDOT
CALL CURVE(X,Y,NPTS,0)
CALL RESET('CHNDOT')
CALL ENDGR(4)
```

```
CC
CC READ COHERENCE (==)
CC
```

```
COHERENCE (==)
REWIND KRX
```

BIGPLT (CONT'D.)

```

READ(KRX,90) DUM
READ(KRX,91) (Y(J),J=1,NPTS)

```

```

CC
CC DRAW FRAME AND PLOT COHERENCE(++)
CC

```

```

CALL DREL(0.,QQQ)
CALL LABEL(1,16)
CALL TITLE(1H,1,LAB,100,0,0,P,QQ)
CALL FRAME
CALL LABEL(NCAT,IT(6))
CALL MNPL(NOPT,XMN,XX,0.,1.,.25,NPTS,P,QQ)

```

```

CC
CC WRITE LEGEND
CC

```

```

YP=YP+.25
CALL SETMES(21,.5,YP)
CALL ABUTMS(23)
CALL ABUTMS(17)
CALL SETMES(21,2.5,YP)
CALL ABUTMS(24)
CALL ABUTMS(18)
YP=YP-.25
CALL SETMES(21,.5,YP)
CALL ABUTMS(25)
CALL ABUTMS(19)
CALL SETMES(21,2.5,YP)
CALL ABUTMS(26)
CALL ABUTMS(20)

```

```

CC
CC PLOT REMAINING COHERENCE
CC

```

```

COHERENCE (--)
REWIND KRX
READ(KRX,90) DUM
READ(KRX,93) (Y(J),J=1,NPTS)
CALL DOT
CALL CURVE(X,Y,NPTS,0)

```

```

COHERENCE (+-)
REWIND KRX
READ(KRX,90) DUM
READ(KRX,95) (Y(J),J=1,NPTS)
CALL DASH
CALL CURVE(X,Y,NPTS,0)

```

```

COHERENCE (->)
REWIND KRX
READ(KRX,90) DUM
READ(KRX,97) (Y(J),J=1,NPTS)
CALL CHNDOT
CALL CURVE(X,Y,NPTS,0)

```

BIGPLT (CONT'D.)

C
CALL ENDGR(4)
CALL ENDPL(4)

C
90 FORMAT(A6)
91 FORMAT(6X,A6)
92 FORMAT(12X,A6)
93 FORMAT(18X,A6)
94 FORMAT(24X,A6)
95 FORMAT(30X,A6)
96 FORMAT(36X,A6)
97 FORMAT(42X,A6)
98 FORMAT(48X,A6)
99 FORMAT(15I5)

C
500 CALL DONEPL

C
STOP
END

ABUTMS

```
SUBROUTINE ABUTMS(ITEM)
  INTEGER LAB(12),ICAT(12,30,2)
  COMMON LAB,ICAT
```

```

SUBROUTINE ABUTMS TAKES A SELECTED LABEL FROM THE DEFAULT
CATALOG AND CONCATENATES IT WITH THE LABEL WHICH WAS
PLACED ON THE PLOT BY THE IMMEDIATELY PRECEDING CALL TO
SETMS OR ABUTMS.

```

ON ENTRY:

ITEM IS THE CATALOG NUMBER OF THE SELECTED LABEL

```
CALL HEIGHT(.1)
CALL LABEL(1,ITEM)
CALL MESSAGE(LAB,100,'ABUT','ABUT')
CALL RESET('HEIGHT')
RETURN
END
```

EDF: 20

Q: >

ALFSET

SUBROUTINE ALFSET

[illegible]

ALPHABET

ESCAPE CHARACTER

BASALF - L/CSTD

—

MIXALF - STAND

“ ”

MX3ALF - L/CGR

19 03

MX4ALF - INSTR

18

CALL BASALF('L/CSTD')

CALL MIXALF ('STAND')

CALL MX3ALF ('L/CGR', '!')

CALL MX4ALF('INSTR','♦')

CALL FOR
RETURN

END

EDF: 20

08:25

```
SUBROUTINE BOTTOM(NBOT,Y,KF)
INTEGER LAB(12),ICAT(12,30,2),IPAK(120),NDOC(3),L(12)
COMMON LAB,ICAT,IPAK
```

24:3

BOTTOM (cont)

IF (NBOT.EQ.0) RETURN

REWIND KF

READ (KF,77) (IPAK(J),J=1,7)

READ (KF,78) (IPAK(J),J=8,15)

READ (KF,79) IPAK(16)

READ (KF,76) (IPAK(J),J=17,19)

FORMAT (1X,3A6)

FORMAT (19X,7A6)

FORMAT (1X,2A6,12X,6A6)

FORMAT (55X,A6)

L(1)=0

L(2)=2

L(3)=4

L(5)=6

L(4)=5

L(5)=6

L(6)=7

L(7)=8

L(8)=9

L(9)=12

L(10)=15

L(11)=18

L(12)=19

XX=.5

YY=Y-.5

CALL HEIGHT(.07)

DO 7 J=1,11

KL=L(J)

K=L(J+1)-L(J)

DO 6 M=1,K

NDOC(M)=IPAK(KL+M)

CONTINUE

IF (J.NE.7) GO TO 5

XX=3.5

YY=Y-.5

YY=YY-.17

LL=15+J

KK=K*6

CALL LABEL(2,LL)

CALL MESSAGE(LAB,100,XX,YY)

CALL MESSAGE(NDOC,KK,'ABUT','ABUT')

CONTINUE

CALL RESET('HEIGHT')

RETURN

END

CONINT

SUBROUTINE CONINT(KCI,XMAX,DX)

REAL X(3),U(3),L(3)

SUBROUTINE CONINT(KCI,XMAX,DX) MAKES USE OF DISSPLA
ROUTINE CURVE TO DRAW CONFIDENCE LIMITS ON THE PLOTS.

EACH INTERVAL HAS ITS ENDPOINTS STORED IN THE ARRAY X WITH THE CORRESPONDING UPPER AND LOWER 90% CONFIDENCE LIMITS STORED IN ARRAYS U AND L. CURVE IS CALLED TO DRAW THESE ON THE PLOT USING THE LOWEST AVAILABLE DEC-
ADE AS A REFERENCE BASE.

INPUT PARAMETERS:

KCI FORTRAN REFERENCE NUMBER FOR THE DATA SET
CONTAINING THE CONFIDENCE LIMIT DATA.

XMAX LARGEST X-VALUE (FREQUENCY) IN THE ENERGY
DENSITY PLOT.

DX VALUE OF THE LOWEST DECADE ON THE Y-AXIS
 WHICH IS ABOVE THE PLOT ORIGIN.

2483

(see figure E-5 on next page for flow chart)

CONINT (CONT'D.)

Subroutine CONINT Flow Chart

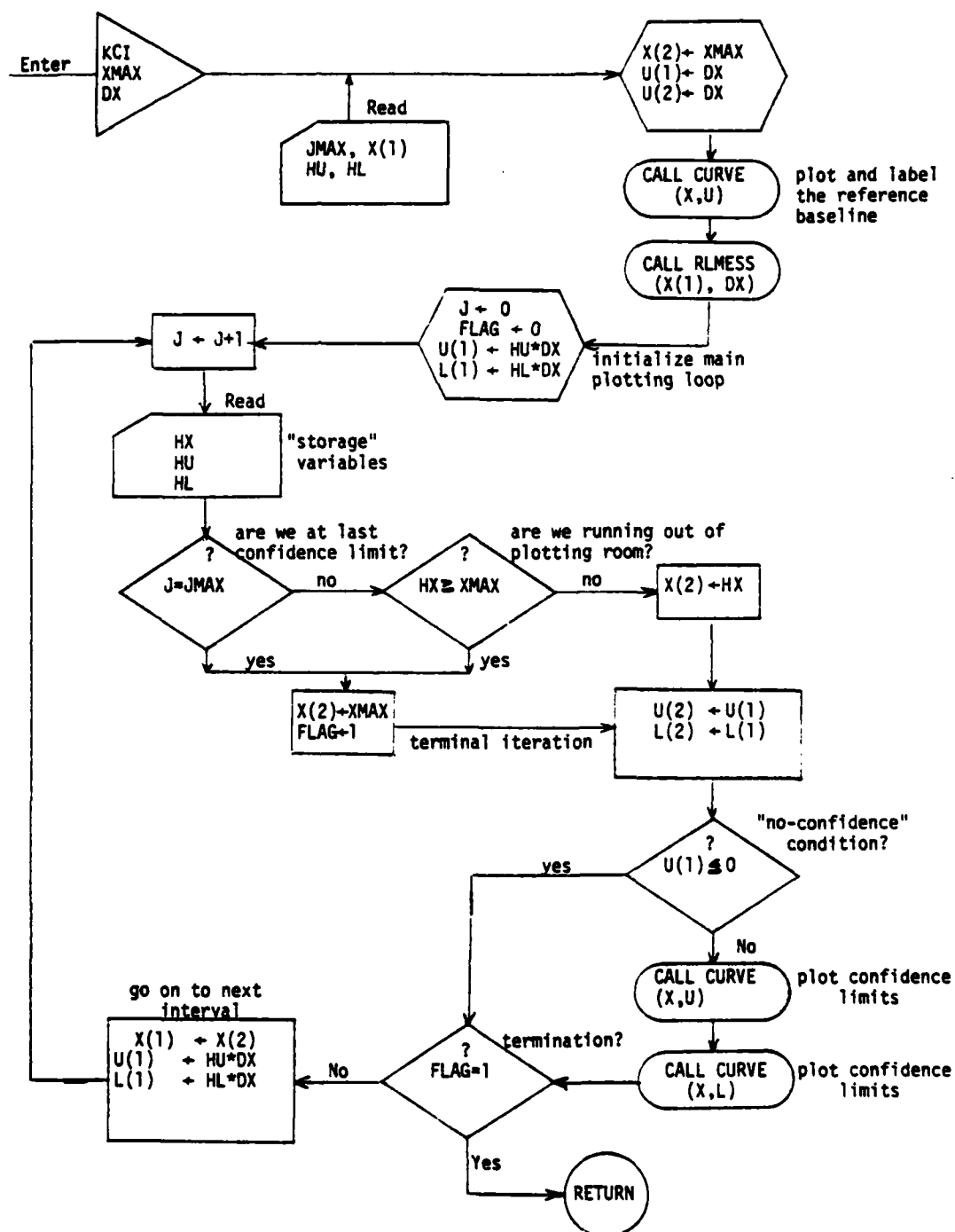


Figure E-5

E-26

DECADE

SUBROUTINE DECADE(XMN,XMX)

SUBROUTINE DECADE ROUNDS XMN DOWN TO THE NEXT LOWEST
POWER OF 10 AND ROUNDS XMX UP TO THE NEXT HIGHEST
POWER OF 10.

ON ENTRY XMN AND XMX ARE ASSUMED TO BE REAL
NUMBERS GREATER THAN $10^{(-6)}$.

ON EXIT XMN AND XMX ARE THE ROUNDED VALUES.

NOTE IF XMN OR XMX WERE INPUT LESS THAN OR EQUAL TO 10 $\diamond$ (-6), THEN THE OUTPUTS ARE 10 $\diamond$ (-6) AND 10 $\diamond$ (-5) RESPECTIVELY.

(see figure E-6 on next page for flow chart)

$R = .000001$

$$\mathbb{C} = \mathbb{D}$$

```
1  C=10.↵C
   IF (XMN.GE.C) GO TO 1
   XMN=C/10.
```

```

2  D=10.4D
   IF (XMX.GT.D) GO TO 2
   XMX=D

```

RETURN
END

EDF: 29

08 >

DECADE (CONT'D.)

Subroutine DECADE Flow Chart

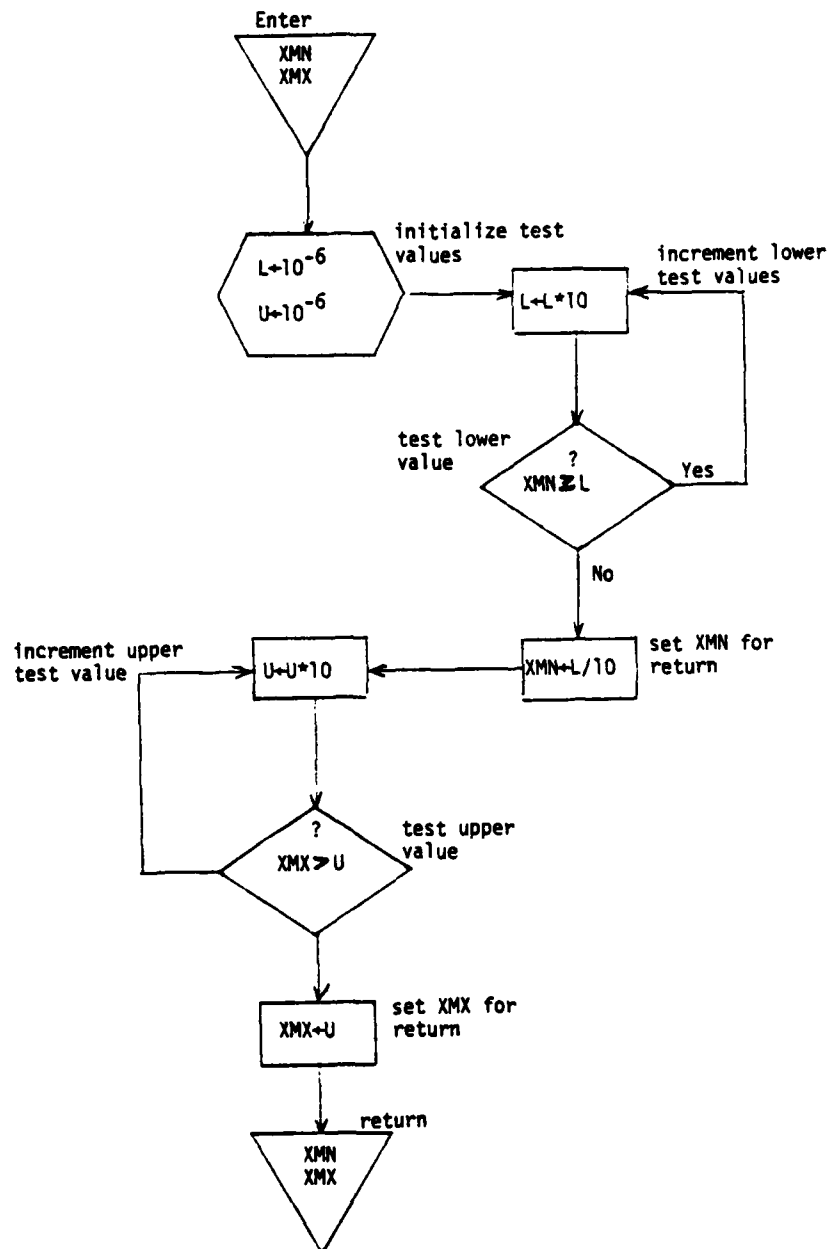


Figure E-6
E-28

DOWN

```

SUBROUTINE DOWN(NBOT,YPOS,Q)
C SUBROUTINE DOWN FURNISHES AN EFFECTIVE 'INVERSE' FOR
C SUBROUTINE UP, I.E., IT MOVES THE PHYSICAL ORIGIN DOWN
C AND INCREASES THE LENGTH OF THE Y-AXIS.
C
C ON ENTRY:
C
C NBOT          ESCAPE PARAMETER
C               NBOT=0      RETURN WITHOUT EXECUTION
C               NBOT=1      ROUTINE EXECUTED
C
C YPOS          LENGTH IN INCHES FROM BOTTOM OF PAGE
C               TO PHYSICAL ORIGIN
C
C ON EXIT:
C
C NBOT          ENTRY VALUE MINUS 1.5
C
C YPOS          ENTRY VALUE PLUS 1.5
C
C IF(NBOT.NE.0) RETURN
C YPOS=YPOS-1.5
C Q=Q+1.5
C RETURN
C END

```

●

10

金

MINMAX

```
SUBROUTINE MINMAX(MIN,MAX,TMN,TMX,NPTS)
REAL MIN,MAX,X(4100),Y(4100)
INTEGER LAB(12),ICAT(12,30,2),IPAK(120)
COMMON LAB,ICAT,IPAK,X,Y
```

```

SUBROUTINE MINMAX SCANS AN ARRAY V OF LENGTH NPTS AND RETURNS
ITS MAXIMUM AND MINIMUM VALUES - MAX/MIN - PROVIDED THEY ARE
GREATER THAN/LESS THAN THRESHOLD LEVEL INPUTS - TMX/TMN .

```

ON ENTRY:

5

IS IN COMMON BLOCK RESERVED FOR ARRAY
CONTAINING Y-COORDINATES

TMN

IS MINIMUM THRESHOLD VALUE

TMX

IS MAXIMUM THRESHOLD VALUE

ON EXIT:

V. TMN. TMX

ARE UNCHANGED

MIN

IS LESSER OF TMN AND MIN(V(J) J=1,..NPTS)

MAX

IS GREATER OF TMX AND MAX(V(J) J=1,...NPTS)

$$\text{MIN} = \text{TMN}$$
$$MAX = TMX$$

```
DO 1 J=1,NPTS
```

```
IF (Y(J).LT.MIN) MIN=Y(J)
```

```
IF (V(J).GT.MAX) MAX=V(J)
```

CONTINUE

RETURN

END

EDF: 36

083

MNPL

```

SUBROUTINE MNPL(NOPT,XMN,XXM,YMN,XXY,YSTP,NPTS,P,QQ)
INTEGER LAB(12),ICAT(12,30,2),IPAK(120)
REAL XMN,XXM,YMN,XXY,YSTP,X(4100),Y(4100),QQ
COMMON LAB,ICAT,IPAK,X,Y

```

SUBROUTINE MNPL CALLS THE DISSPLA ROUTINES WHICH ARE APPROPRIATE FOR SETTING UP THE GRAPH AXES AS SPECIFIED BY THE USER.

ON ENTRY:

AXIS SELECTION PARAMETER			
NOPT	NOPT	X-AXIS	Y-AXIS
	0	LOG	LIN
	1	LIN	LIN
	2	LOG	LOG

XMN, XMX ARE THE MINIMUM AND MAXIMUM VALUES WHICH
APPEAR ON THE X-AXIS.

YMN, YMX ARE THE MINIMUM AND MAXIMUM VALUES WHICH
APPEAR ON THE Y-AXIS.

YSTEP THE DISTANCE IN AXIS UNITS BETWEEN
Y-AXIS TICK MARKS.

P,QQ LENGTHS OF X AND Y AXES IN INCHES.

X,Y ARRAYS WHICH HOLD X AND Y COORDINATES OF
POINTS TO BE PLOTTED.

LAB ARRAY WHICH HOLDS THE Y-AXIS LABEL.

NPTS **NUMBER OF POINTS TO BE PLOTTED.**

(see figure E-7 on next page for flow chart)

```
IF (NOPT.EQ.1) GO TO 5
```

```
CALL ALGPLT(XMN, XMX, P, XOR, XCYLE)
CALL XLOG(XOR, XCYLE, 0., 1.)
```

```
IF (NOPT.EQ.2) GO TO 15
GO TO 10
```

```
5 CALL GRAF(XMN,'SCALE',XMX,YMN,YSTP,YMX)
```

```
10 CALL YGRAXS(YMN,YSTP,YSX,YSY,LAB,100,0.,0.)  
GO TO 20
```

```
15 CALL ALGPLT(YMN, YMX, QQ, YOR, YCYCLE)
   CALL YLGAXS(YOR, YCYCLE, QQ, LAB, 100, 0., 0.)
```

```

20 CALL CURVE(X,Y,NPTS,0)
   D=1
   RETURN
END

```

MNPL (CONT'D.)

Subroutine MNPL Flow Chart

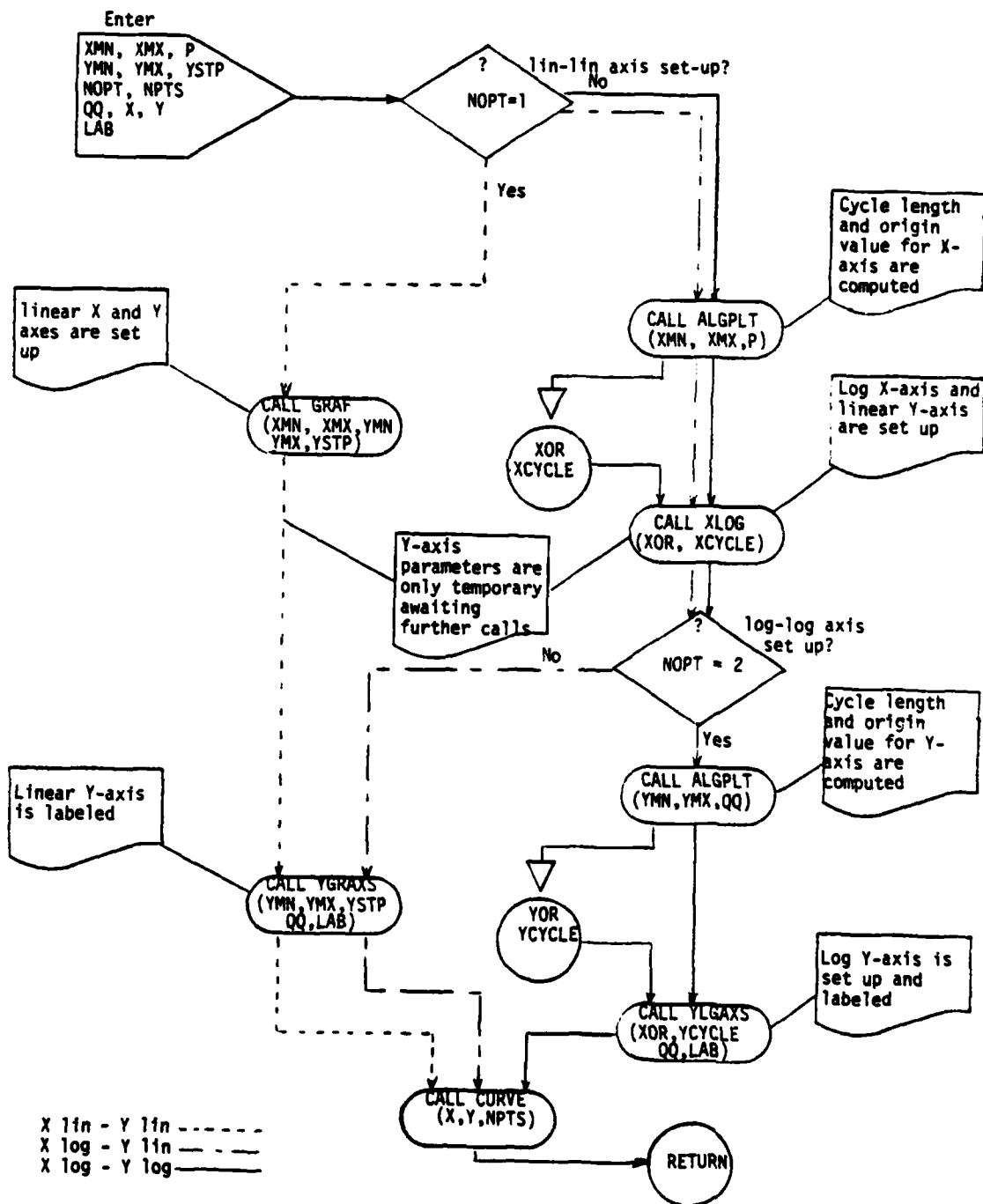


Figure E-7
E-33

SETMES

```
SUBROUTINE SETMES (ITEM,XP,YP)
  INTEGER LAB(12),ICAT(12,30,2)
  COMMON LAB,ICAT
```

SUBROUTINE SETMES TAKES A SELECTED LABEL FROM THE DEFAULT CATALOG CONTAINED IN ARRAY ICAT AND WRITES IT AT A SPECIFIED LOCATION ON THE PLOT.

ON ENTRY:

ITEM IS THE CATALOG NUMBER OF THE SELECTED LABEL

XP,YP ARE THE HORIZONTAL AND VERTICAL DISTANCES IN INCHES FROM THE CURRENT PLOT ORIGIN TO THE LOCATION WHERE THE MESSAGE STARTS.

```
CALL HEIGHT(.1)
CALL LABEL(1,ITEM)
CALL MESSAGE(LAB,100,XP,YP)
CALL RESET('HEIGHT')
RETURN
END
```

EOF:22
0:>

SETUP

SUBROUTINE SETUP(K,KU,NCAT,NBOT,NOPT,NPTS,IT)

SUBROUTINE SETUP READS USER INSTRUCTION DATA AND USES IT TO
SET THE PROGRAM CONTROL PARAMETERS NECESSARY FOR CARRYING
OUT THE ENCODED INSTRUCTIONS.

ON ENTRY:

K IS PARAMETER FOR SELECTING AXIS TYPE
AND LABEL CATALOG
=0 => NO PLOT WANTED - RETURN
>0 => TITLES AND LABELS SELECTED FROM
DEFAULT CATALOG
<0 => TITLES AND LABELS SELECTED FROM
USER SUPPLIED CATALOG
ABS(K)=1 => X-LOG Y-LIN
=2 => X-LIN Y-LIN
=3 => X-LOG Y-LOG

KU FORTRAN UNIT NUMBER FOR THE DATA SET
CONTAINING USER CONTROL RECORDS

ON EXIT:

NCAT IS CATALOG SELECTION PARAMETER
=1 => DEFAULT CATALOG (K>0)
=2 => USER SUPPLIED CATALOG (K<0)
NBOT
=0 => CURRENT METER DATA NOT WANTED
=1 => CURRENT METER DATA TO BE WRITTEN AT
BOTTOM OF PLOT

NOPT AXIS SELECTION PARAMETER
=0 => X-LOG Y-LIN
=1 => X-LIN Y-LIN
=2 => X-LOG Y-LOG

NPTS NUMBER OF POINTS TO BE PLOTTED

IT(J) ARRAY CONTAINING CATALOG NUMBERS FOR
TITLES AND LABELS
J=1,2,3 => IT(J) CONTAINS CATALOG NUMBER FOR
LINES OF TITLE
=4 => IT(J) CONTAINS CATALOG NUMBER FOR
X-AXIS LABEL
=5,6,7 => IT(J) CONTAINS CATALOG NUMBER FOR
Y-AXIS LABELS

50:>

SETUP (cont)

```
INTEGER IT(13),LT(13)
READ(KU,99) NBOT,MPTS,(LT(J),J=1,13)
IF(MPTS.NE.0.AND.MPTS.LE.NPTS) NPTS=MPTS
NCAT=1
IF(K.GT.0) GO TO 1
```

```
0      DO 2 L=1,13
        IT(L)=LT(L)
2      CONTINUE
NCAT=2
K=-K
```

```
0      NOPT=K-1
1      FORMAT(15I5)
99     RETURN
END
```

```
EDF:66 SCAN:15
0:>
```

TOP

```
SUBROUTINE TOP(NCAT,IT)
  INTEGER IT(20),LAB(12),ICAT(12,30,2)
  COMMON LAB,ICAT
```

SUBROUTINE 'TOP' MAKES USE OF DISSPLA ROUTINE 'HEADIN'
TO PLACE A TITLE ON THE PLOT.

THE LINES IN THE TITLE ARE SELECTED ONE AT A TIME FROM THE CATALOG OF LABELS CONTAINED IN ARRAY ICAT AND STORED IN ARRAY LAB FOR INPUT TO 'HEADIN'.

ON ENTRY:

```

CATALOG SELECTION PARAMETER
1 - DEFAULT CATALOG
2 - USER SUPPLIED LABELS

```

```

IT      ARRAY CONTAINING THE CATALOG NUMBERS FOR
        THE SELECTED LINES.
IT(J)=  CATALOG NUMBER FOR J-TH LINE OF TITLE
        (J=1,2,3.)
IT(J)=0 SIGNALS THAT THERE ARE ONLY J-1 LINES
        IN THE TITLE.

```

DETERMINE NUMBER OF LINES IN TITLE C

```
IF (IT(1).EQ.0) RETURN
```

NLINES=1

DO 2 L=2,3

```
IF(IT(L).EQ.0) GO TO 3
```

NLINES=NLINES+1

CONTINUE

WRITE THE LINES OF TITLE

DO 4 J=1,NLINES

ITEM=IT(J)

CALL LABEL (NCAT, ITEM)

NHGT=3

CALL HEADIN(LAB,100,NHGT,NLINES)

CONTINUE

RETURN

END

EDF: 47

0:3 >

UP

SUBROUTINE UP(NBOT,YPOS,Q)

SUBROUTINE UP IS USED TO MAKE ROOM AT THE BOTTOM OF THE
PLOT FOR WRITING CURRENT METER INFORMATION. THIS IS DONE BY
MOVING THE PHYSICAL ORIGIN UP AND REDUCING THE VERTICAL
AXIS LENGTH.

ON ENTRY:

NBOT	ESCAPE PARAMETER
	NBOT=0 RETURN WITHOUT EXECUTION
	NBOT=1 ROUTINE IS EXECUTED
YPOS	LENGTH IN INCHES FROM BOTTOM OF PAGE TO PHYSICAL ORIGIN
Q	LENGTH IN INCHES OF Y-AXIS

ON EXIT:

NBOT	ENTRY VALUE PLUS 1.5
YPOS	ENTRY VALUE MINUS 1.5

IF(NBOT.NE.0) RETURN
YPOS=YPOS+1.5
Q=Q-1.5
RETURN
END

EOF:30

0:>

REFERENCES

REFERENCES

- Amos, D. E. and L. H. Koopmans (1963). Table of the distribution of the coefficient of coherence for stationary bivariate Gaussian processes. National Technical Information Service, U.S. Dept. of Commerce.
- Blackman, R. B. and J. W. Tukey (1958). The Measurement of Power Spectra, Dover Publications, New York, 190pp.
- Fofonoff, N. P. (1969). Spectral characteristics of internal waves in the ocean, Deep-Sea Res. 16, 59-71.
- Gonella, J. (1972). A rotary-component method for analyzing meteorological and oceanographic vector time series, Deep-Sea Res. 19, 833-846.
- Groves, G. W. and E. J. Hannan (1963). Time series regression of sea level on weather, Rev. Geophys., 6, 129-174.
- Hayashi, Y. (1979). Space-time spectral analysis of rotary vector series, Jour. Atm. Sci., 36, 757-766.
- Mooers, C. N. K. (1973). A technique for the cross spectrum analysis of pairs of complex-valued time series, with emphasis on properties of polarized components and rotational invariants, Deep-Sea Res., 20, 1129-1141.
- Muller, P. and G. Siedler (1976). Consistency relations for internal waves, Deep-Sea Res., 2, 613-628.
- Muller, P., D. J. Olbers, and J. Willebrand (1978). The IWEX spectrum, Jour. Geophys. Res., 83, 479-500.
- Singleton, R.C., (1969). An Algorithm for Computing the Mixed Radix Fast Fourier Transform, IEEE trans. on Audio and Electroacoustics, Vol. AU-17, No. 2, June, 1969, pp. 93-103.
- Weisberg, C., (1972). Bermuda sea level in relation to tides weather, and baroclinic fluctuations, Rev. Geophys. Space Phys., 10, 1-49.
- Wunsch, C., (1972). Bermuda sea level in relation to tides, weather, and baroclinic fluctuations, Rev. Geophys. Space Phys., 10, 1-49.

JAYCOR

27 February 1980

Dr. E. Michael Stanley
Naval Ocean Research and Development Activity
Environmental Measurements Program (Code 500)
NSTL Station, Mississippi 39529

Dear Dr. Stanley:

Enclosed is the final version of the "User's Manual for Computer Programs to Perform Oceanographic Vector Times Series Data Analysis and Related Graphics," which represents a deliverable under contract N00014-78-C-0879. The document is divided into two parts: the introductory material and instructions to users, and the program source listing and test data sets.

This software represents the state-of-the-art in vector time series analysis, and will be both durable and versatile, being able to accommodate a number of types of time series of oceanographic or meteorologic vectors. Due to the complex nature of vector time series analysis, it is strongly recommended that any user read the manual very carefully, beginning with the preface, before trying to use the software. We have found in the testing phase that many problems that were encountered were the result of casual reading of the manual and not in the software itself. Also, in this way, the user can take full advantage of the wide range of options available in both the analysis portion of the programs as well as in the graphics.

We regret the delay in submitting this final document, but acknowledge, as you do, the multitude of difficulties that were overcome to reach this point.

We would like to recognize the valuable assistance provided us by Dr. Kim Saunders and Mr. Mark Bergin. Without their aid in working with the NAVOCEAN computer system, this task would have been extremely difficult to complete. We have enjoyed working with the NORDA staff on this project and look forward to future projects for you and the environmental measurements program.

Sincerely yours,



Francis C. Monastero
Director
Ocean and Environmental Sciences Division

Distribution of Final Report Under Contract No. N00014-78-C-0879

<u>Addressee</u>	<u>DODAAD Code</u>	<u>Number of Copies</u>
Scientific Officer	N68462	1
Administrative Contracting Officer	S0514A	1
Director, Naval Research Laboratory, Attn: Code 2627 Washington, DC 20375	N00173	6
Defense Documentation Center Bldg. 5, Cameron Station Alexandria, VA 22314	S47031	12
Office of Naval Research Branch Office Pasadena 1030 East Green Street Pasadena, CA 91106	N62887	1

FILMED
—8