See 1473

2

20086-6013-RU-00

SUBMARINE EFFECTS ENGINEERING CODE

4. Operating Instructions

by

David J. Henryson

July 31, 1975

Sponsored by
Advanced Research Projects Agency
ARPA Order Number 1910

Contract No. N00014-72-C-0074 -- Program Code 3E20

Scientific Officer · Ralph D. Cooper, Program Director
Fluid Dynamics
Office of Naval Research

TRW SYSTEMS GROUP
One Space Park
Redondo Beach
California 90278

20086-6013-RU-00

SUBMARINE EFFECTS ENGINEERING CODE

4. Operating Instructions

by

David J. Henryson

July 31, 1975

DDC

C

Sponsored by
Advanced Research Projects Agency
ARPA Order Number 1910

Contract No. N00014-72-C-0074 -- Program Code 3E20

Scientific Officer· Ralph D. Cooper, Program Director
Fluid Dynamics
Office of Naval Research

TRW SYSTEMS GROUP
One Space Park
Redondo Beach
California 90278

CONTENTS

LIST OF ILLUSTRATIONS

1. INTRODUCTION

The disturbance generated by a moving submerged body in a stratified ocean can be separated, under realistic ocean conditions, according to the distinct physical phenomena responsible for propagating the disturbance. Because, at best, the ocean is weakly stratified, there is a region near the body in which inertia and pressure dominate over buoyant restoring forces at all but very low body velocities. This can be calculated without considering the (weak in this region) effect of density stratification, and will be termed the potential disturbance. Sufficiently far from the body, the inertia and pressure disturbances become small and buoyant restoring forces become significant, so that the disturbance takes the form, predominantly, of internal waves in the stratified ocean. These internal waves can be excited by the displacement effect of the body (which, of course, also drives the less persistent potential disturbance), but in addition, they can be excited by the "collapse" of the turbulent wake of the body. The dynamics of this wake are dominated, close to the body, by inertia and diffusion. Farther away, inertia, pressure and buoyant restoring forces dominate. The region in which diffusion becomes unimportant can be termed the start of collapse, since buoyant restoring forces will then start to flatten the wake, whose density distribution has been changed by turbulent mixing, toward a shape consistent with static density equilibrium, generating internal waves in the process.

The purpose of this study has been to develop a code capable of computing these disturbances with emphasis on optimizing the speed, simplicity and versatility of the code to make it suitable for engineering studies involving large numbers of individual calculations. The present report describes the operation of the resulting code, which is indeed quite versatile. The cost of this versatility is, as always, an unavoidable complexity. The complete capabilities of the code are described in the following sections, but it is not reasonable to expect a new user, immediately on reading this description, to generate the data set required to run a specific calculation.

It is recommended, rather, that the new user start by becoming thoroughly familiar with volumes 1-3 of this series, which describe the analytical basis for the code before reading further in the present report. The first few cases run should follow the format of one of the sample calculations presented here, before the user attempts to invoke one of the almost limitless variations thereof.

2. PROGRAM CAPABILITIES

SEEK is a Fortran IV program which runs on the CDC 6000 series computers. It simulates the internal waves generated by various sources moving horizontally through a vertically stratified ocean. A detailed description of the problem and the technical approach is contained in references 1-3. The capabilities of the program are briefly outlined below.

Ocean description:

- up to 400 points in thermocline table
- variable spacing in thermocline table
- table covers only thermocline, not unstratified regions
- up to 80 modes

Source models:

- Rankine or dipole body
- oval or circular cross-section superstructure
- wake collapse

Disturbance calculation:

- select from 10 variables (cross-track velocity, vertical displacement, etc.)
- potential flow solutions for Rankine body, oval superstructure
- Fourier transforms by FFT algorithm for near field
  - select x-y or z-y grid pattern
  - up to 256 point transform (unaliased)
  - mode range of up to 40
- Fourier transforms by stationary phase approximation for far field
  - select x-y, z-y or z-x grid pattern
  - up to 800 points per cut
  - mode range of up to 41

Input

- data library capability
- save and use processed ocean data (compute eigenvalues only once)
- free form input (Namelist)

Output:

- print
- plot
- variable format
- save output data and reformat

Multi-case

- input only changes from previous case
- redundant dispersion relation calculations eliminated

3. OVERVIEW OF PROGRAM OPERATION

The SEEK program consists of five major modules as illustrated in Figure 1. Figure 2 diagrams the program files which are manageable by the user.

At the start of execution, the program transfers control to the input processor which reads the first card of the data deck from the INPUT file. This card is an "input processor control card" (IPCC). An IPCC (depending on its type) may direct the input processor to

1. read data from the data deck (file INPUT),
2. read data from the data library (file TAPE1),
3. interrupt the input sequence and execute the data read in,
4. terminate the program.

Typically, several sets of data are read from files INPUT and TAPE1 before the input sequence is interrupted. When an IPCC of type 3 is read, the input processor examines selected variables, noting which of them were changed during the input sequence. The input processor then returns control to the main program.

The dispersion table generator performs its function based on the options selected and available data. The first case of a run presents two possibilities.

1. The dispersion tables are constructed entirely from data read by the input processor.
2. The dispersion tables are constructed using both input data and "processed ocean" data from file TAPE2.

Every time item 1 holds, the program writes the new processed ocean data on file TAPE2. If this file is saved at the end of the run, it can be restored and used in a subsequent run. Substantial savings can be realized by using processed ocean data.

Processed ocean data are used only for the first case of a run. It is assumed (but not required) that each subsequent case is a more or less minor variation of the preceding case. Accordingly, the dispersion table generator uses results from the preceding case whenever possible. The

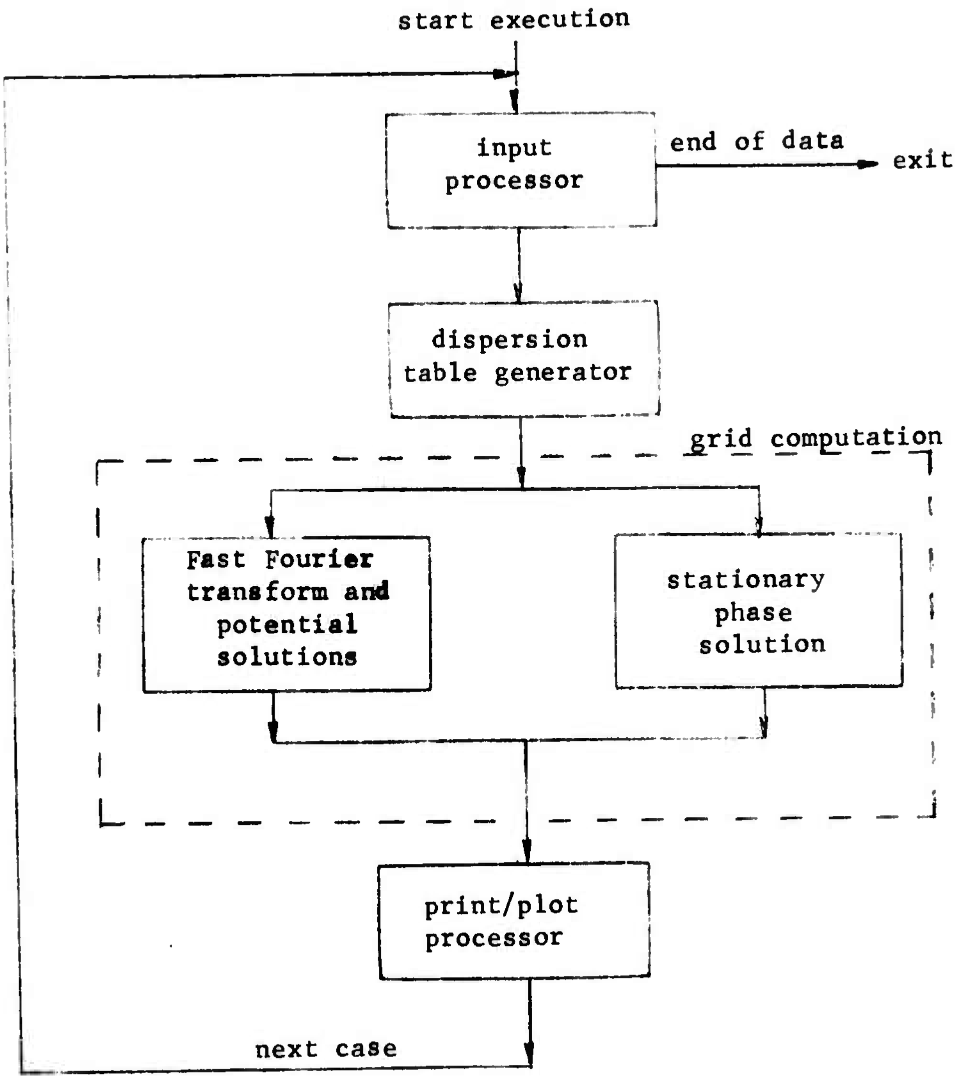


Figure 1. Program Organization

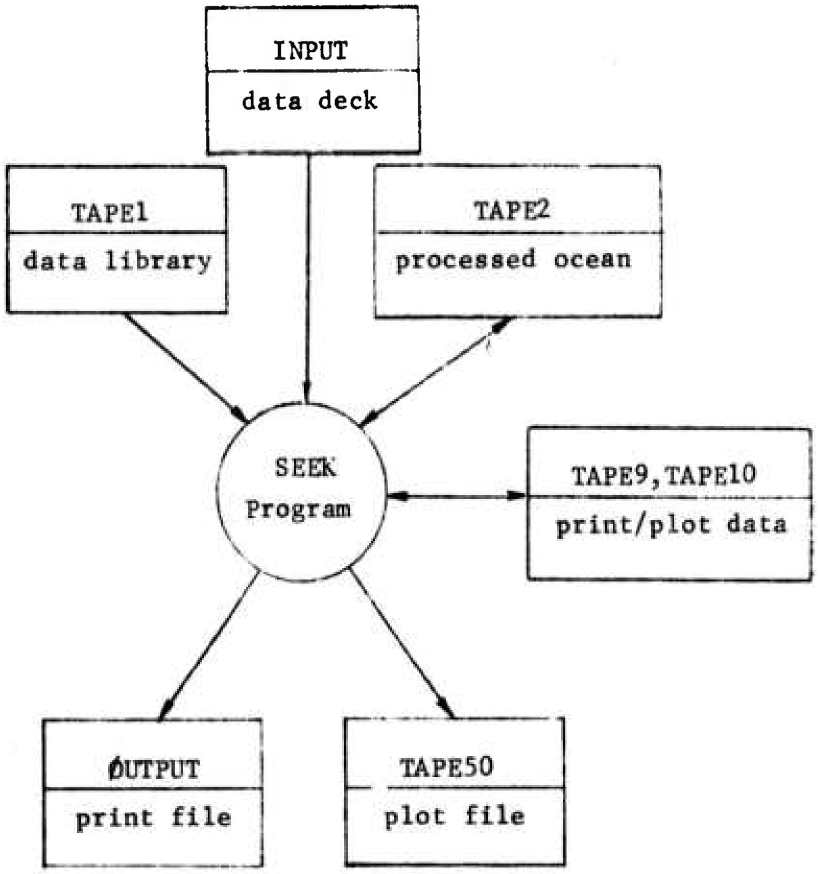


Figure 2. User-manageable files.
Scratch files are not shown.

elements which must be recomputed are determined by examining the "change notices" generated by the input processor.

The dispersion table generator allows for the print/plot of a selected eigenvector.

The grid computations are performed by one of two modules selected by an input option.

1. Near field: fast Fourier transform (FFT) and/or potential solutions. Two grid patterns are available, an x-y grid and a z-y grid. Note that, for instance, an x-y grid produces a vector at each value of x, where the components of the vector are the signal (disturbance variable) values at the various y coordinates. If the FFT option is used, it is also possible to print the dispersion tables, print/plot the individual dispersion variables and print/plot the power spectra. If only the potential flow solution is used, the dispersion relation computations are avoided. If both options are on, the resulting signal is the superposition of the two.

2. Far field: stationary phase solution. Three grid patterns are available -- an x-y, a z-y and a z-x grid. It is possible to print the dispersion tables, print the wave family edge table, and print/plot the individual dispersion variables.

In both modules, if a grid is specified, the signal data are automatically sent to the print/plot processor. It should be noted that a grid involving multiple depths is significantly slower in execution than an x-y grid.

The print/plot (PP) processor is a generalized module for data display. All data destined for the PP processor are written on files TAPE9 and TAPE10. TAPE9 has the actual data while TAPE10 has format, scaling and file structure information. In writing these files, the program provides a preset format in which to display the data. This format may vary according to the type of information being written. For each set of data on the files, the user may override the preset format, specifying no display, a display window, print only, plot only, plot scaling, etc.

Generally, data generated during the computational phase of a case are displayed at the end of that case. The PP data files are then rewound and PP data for the next case are written over the old data. However, it is possible to accumulate data for multiple cases on the PP data files and produce all the displays at the end of the last case (note that it is easy to produce rather large data files in this way). Whether the files contain all the data or just data for the last case, it may be desirable to save them at the end of the run. Then the displays can be examined and, if appropriate, another run can be made to display the saved data in a different format.

4. DATA DECK STRUCTURE

The "data deck" is a set of source cards which the program reads from the INPUT file. The reading of data and its execution are governed by "input processor control cards" (IPCCs) which are contained within the data deck.

In addition to IPCCs, the data deck may contain "data sets". A data set is defined as all the cards read when a single namelist read is executed. Note that namelist input is a system function; syntax rules for entering namelist data may be found in the Fortran manual. Generally, a data set starts with a $ in column 2, followed by a namelist name, followed by data (which may extend over several cards), and is terminated by another $. Several namelist input sets have been implemented in the program. These namelist sets categorize the data as described in Figure 3.

Namelist Name	Data Type	General Description of Data
OCEAN	O	Ocean description and dispersion table specification
SOURCE	S	Source model selection and description
GRID	G	Grid definition
PP	P	(Print/plot) editing specifications for output display

Figure 3. Input data classification

There are four different IPCCs. They are all fixed field cards which start in column 1. They have no embedded blanks except possibly for the "id" field.

<table>
<tr><th>IPCC</th><th>Description</th></tr>
<tr><td>INP,t</td><td>This card instructs the program to read a data set of data type "t" from the data deck. The "data type" is a one character code defined in Figure 3. The data set must immediately follow this card. Any data set in the data deck must be preceded by this card.</td></tr>
<tr><td>LIB,t,id</td><td>This card instructs the program to read a data set of data type "t" with identifier "id" from the data library file. The "data type" is a one character code defined in Figure 3. "id" is a 10 character identifier used to locate the desired data set in the data library file. The data library is described in Figure 5.</td></tr>
<tr><td colspan="2">Note that if a given data type appears more than once on the above cards, the current data set is overlaid on top of the previous data set(s), but see the remarks in Section PP DATA.</td></tr>
<tr><td>RUN</td><td>This card instructs the program to stop reading data and start the appropriate computations for the case. When the case is finished, the program returns to reading the data deck. With some exceptions specifically noted elsewhere, the program does not alter input values. Hence, input for the next case need only reflect changes to the case just run.</td></tr>
<tr><td>END</td><td>This card causes the program to terminate. It is the last card of the data deck. Note that the preceding card should be RUN.</td></tr>
</table>

Figure 4. Input Processor Control Card (IPCC) Description

The multi-case capability can sometimes be utilized to effect substantial savings in computer time. Specifically, only those elements of the dispersion tables which will differ from the previous case are recomputed. The program determines which elements must be recomputed by comparing the values of certain variables just before and after an input sequence. Any detectable change in the body depth, for instance, will cause up to four of the dispersion tables to be recomputed. A good general rule for multi-case runs is thus: input only those values which you honestly want changed.

4.1 Data Library Description

TAPE1 is the data library file. It consists of a collection of data sets, each of which is preceded by an identification card. When the input processor control card LIB is encountered in the data deck, the program locates the corresponding identification card on the data library file and reads the data set which follows it.

The format of the identification cards is shown in Figure 5. Note that both the type "t" and identifier "id" are matched with the LIB card.

Identification Card	Description
*t,id	The "*" is in card column 1. "t" is a one character data type defined in Figure 3. "id" is a 10 character identifier matched character-for-character with the LIB card.
*END	Last card of the data library. Starts in card column 1.

Figure 5. Data library identification card format

4.2 Ocean Data

This section describes the variables which may be input to the namelist set ØCEAN. Note that all depths are positive. Units are meters, seconds, and radians.

1. Ocean depth

OCNDEP	depth of the ocean

2. Thermocline description

NZT	number of points in the thermocline table, TDEP and SQBV (NZT $\le$ 400)
TDEP(i)	list of thermocline depths (at which N^2 is specified). TDEP(1) = top of the thermocline and is always input. If DTDEP = TDEPMX = 0, then TDEP(i), i = 2,...,NZT must also be input; otherwise, the program fills in these entries.
DTDEP	If non-zero, TDEP(i) = TDEP(1) + (i - 1) *DTDEP, i = 2,...,NZT. Otherwise, it is ignored.
TDEPMX	If DTDEP = 0 $\neq$ TDEPMX, Δ = (TDEPMX - TDEP(1))/(NZT-1) and TDEP(i) = TDEP(1) + (i-1) * Δ, i = 2,...,NZT. Otherwise, it is ignored.
NFLAG	=1 : special option allows N = Brunt-Vaisala frequency input to SQBV
	=0 : nominal option is N^2 input to SQBV
SQBV(i)	N^2 at depth TDEP(i)

3. Dispersion table description

MØDES	Number of modes in the dispersion tables (MØDES $\le$ 80)
NK	number of entries in the wave number list TABK (length of dispersion tables) (NK $\le$ 100)
TABK(i)	list of wave numbers at which dispersion relation is computed. TABK(1) must be 0. If DKRAT = 0, TABK(i), i = 2,...,NK must also be input; otherwise, the program fills in these entries.

DKRAT	If non-zero, it is the ratio of the last increment in K to the first increment, i.e., DKRAT = (TABK(NK) - TABK(NK-1))/(TABK(2) - TABK(1)) Using TABK(1), DKRAT and RKMAX, the program constructs TABK(i), i = 2,...,NK such that (TABK(i+1) - TABK(i))/(TABK(i) - TABK(i-1)) = constant. If DKRAT = 0, it is ignored.
RKMAX	If DKRAT ≠ 0, RKMAX is the largest value of K in the wave number list. Otherwise, it is ignored.

4. Processed ocean data

The variables in Sections 1, 2 and 3 above are sufficient to determine the results of the most time consuming part (eigenvalue determination) of generating the dispersion tables. Provision has been made to save these results on peripheral storage so they can be retrieved at a later date with a resulting savings in computer time.

Any change to a variable in Sections 1, 2 and 3 results in a "new ocean" which causes the program to compute the entire set of dispersion tables and write the "processed ocean data" onto file TAPE2. At the end of the run, this file can be saved. Only data from the last new ocean of the run will be on the file.

If the first case of a subsequent run requires the same ocean, the processed ocean data can be retrieved and made available to the program on file TAPE2. To use that data, set LIBSEA = 1 and make no entry to the variables in Sections 1, 2, 3.

The setting LIBSEA = 1 is valid only for the first case, since the program internally generates a more complete set of information for subsequent cases. Indeed, the program forces LIBSEA = 0 after the first case. If LIBSEA wereused for the first case and a new ocean is desired in a subsequent case, the new ocean must be completely defined by input. Input variables are not necessarily saved as processed ocean data. Note that the new ocean data will be written over the original.

LIBSEA = 0 : ocean defined by input
= 1 : use processed ocean data from TAPE2

5. Skip dispersion relation

The capability exists to skip over the program module which generates the dispersion tables. Ordinarily this would only be done when it is desired to use the print/plot module to display data which were generated earlier.

NODISP = 0 : program determines dispersion table requirements

= 1 : skip dispersion table generation

6. Display

IPRDT = 0 : option off

= 1 : the dispersion tables are printed. This is a direct print - the data are not sent to the print/plot module. The printing is done from the grid module and only modes MØDE1 through MØDEN are printed (see Section on GRID for definitions of MØDE1 and MØDEN).

IPPDT(i) = 0 : option off

= 1 : dispersion table i is sent to the print/plot processor, where table i is defined in Figure 6. These data are generated in the grid module and include modes MØDE1 through MØDEN.

IPPVEC = 0 : option off

= i : the eigenvector ψ for mode i is sent to the print/plot processor. The PP id is EVEC. The data are generated in the dispersion table module.

IPREDG = 0 : option off

= 1 : the wave family edge tables are printed. This is a direct print - the data are not sent to the print/plot module. The printing is done from the stationary phase module and only modes MØDE1 through MØDEN are printed.

i	Table	PP id
1	$d\lambda/dK$	DL/DK
2	ψ (obs depth) [eigenfunction]	W(0BS)
3	$d\psi/dz$ (obs depth)	DW/DZ(0BS)
4	$d\psi/dz$ (body depth)	DW/DZ(B0D)
5	I_W [wake integral]	TWAKE
6	ψ (super bottom) - ψ (super top)	TSUPR
7	λ [$=1/C^2$]	LAMBDA
8	$d^2\lambda/dK^2$	D2L/DK2
9	$-y/x$	-Y/X

Figure 6. Dispersion Table Display

4.3 Source Data

This section describes the variables which may be input to the namelist set SØURCE. Units are meters and seconds. The net disturbance is the superposition of all selected sources.

IBODY = 0 : option off
= 1 : Rankine body is simulated. Required inputs are BØDDEP,BØDSPD,BØDDIA,BØDLEN
= 2 : dipole body is simulated. Required inputs are BØDDEP,BØDSPD,BØDDIA

ISUPR = 0 : option off
= 1 : a superstructure with ellipsoidal cross section is simulated. Required inputs are BØDDEP,BØDSPD, SUPTØP,SUPBØT,SUPMID,SUPDIA,SUPLEN
= 2 : a superstructure with circular cross section is simulated. Required inputs are BØDDEP,BØDSPD, SUPTØP,SUPBØT,SUPMID,SUPDIA

IWAKE = 0 : option off
= 1 : a wake is simulated. Required inputs are BØDDEP,BØDSPD,CWAKR,CWAKX,RESLVS,CWAKM,BØDDIA

IPBØDY = 0 : option off
= 1 : the potential solution of a Rankine body is evaluated. Required inputs are the same as for IBØDY=1; see also XPMAX,YPMAX, and ISPHAS in the section GRID DATA.

IPSUPR = 0 : the potential solution of a superstructure with ellipsoidal cross section is evaluated. Required inputs are the same as for ISUPR=1; see also XPMAX, YPMAX and ISPHAS in the section GRID DATA.

BØDDEP	depth of the body centerline (input positive)
BØDSPD	body speed
BØDDIA	body diameter
BØDLEN	body length

SUPT0P	distance from body centerline to top of superstructure line source (positive up)
SUPB0T	distance from body centerline to bottom of superstructure line source (positive up)
SUPMID	x coordinate of center of superstructure ($x=0$ at body center)
SUPDIA	maximum transverse dimension of superstructure
SUPLEN	superstructure length
CWAKR	sizing coefficient for wake radius. $a_w = C_r \frac{D}{2} F^{\frac{1}{4}}$ where a_w = wake radius, C_r = CWAKR, D = body diameter, F = Froude number = $\frac{2\pi U}{ND}$ with U = body speed, N = local average Brunt-Vaisala frequency
CWAKX	coefficient for computing start of wake collapse. $x_w = C_x DF$ where x_w is x coordinate of start of wake collapse, C_x = CWAKX and D and F are as above.
CWAKM	wake mixing fraction = ϵ
RESLVS	The integral in the wake source term is performed numerically via trapezoidal integration. The step size is varied to hit each thermocline point within the wake while ensuring that the increment in the argument of the sine function never exceeds π/RESLVS. RESLVS=5 may be used as a rule of thumb.

4.4 Grid Data

This section describes the variables which may be input to the namelist set GRID. Units are meters and seconds.

1. Grid Variable

IVAR	The value of IVAR selects the variable (or signal) to be computed at each grid point. The options are given in Figure 7.

IVAR	Variable	Name
1	u'	X-VELOCITY (U)
2	v	Y-VELOCITY (V)
3	δ_x	X-DISPLACE (DELTA-X)
4	δ_y	Y-DISPLACE (DELTA-Y)
5	δ_z	Z-DISPLACE (DELTA-Z)
6	ϵ_x	X-STRAIN (EPSILØN-X)
7	ϵ_y	Y-STRAIN (EPSILØN-Y)
8	γ_{xy}	SHEAR STRN (GAMMAXY)
9	σ	DILATATION (SIGMA)
10	w	Z-VELOCITY (W)

Figure 7. Grid Variable Selection

2. Mode Range

The output signal will be computed as the superposition of modes MØDE1 through MØDEN inclusive. Note that $1 \leq$ MØDE1 $\leq$ MØDEN $\leq$ MØDES where MØDES is the number of modes in the dispersion tables (see the section OCEAN DATA). The range of modes in the grid computation MODEN-MODE1 + $1 \leq 40$.

MØDE1	first mode in grid computation.
MØDEN	last mode in grid computation.

3. Near/Far Field Selection

In the near field option, a fast Fourier transform (FFT) technique is used; a potential solution is also available. In the far field option, a stationary phase technique is used.

Typically, in the far field, the FFT will exhibit aliasing problems while the potential solution becomes negligible. In the near field, the stationary phase approximation becomes inaccurate.

ISPHAS = 0 : (near field) use FFT and/or potential solution.

= 1 : (far field) use stationary phase. Note: while the y coordinates of the grid are input positive, the stationary phase module actually uses negative y values. In the output displays, this is reflected by labeling the y coordinate as "-y". This quirk cannot be circumvented by input.

4. Grid Definition

The signal values are computed and displayed at the points of a two dimensional grid. The coordinates of the grid points are (X_i, Y_j, Z_k), i=1,...,NX; j=1,...,NY; k=1,...,NØBS. Exactly one of the NX, NY, NØBS must equal 1. If the FFT option is used, NY must be greater than 1.

If NØBS=1, an X-Y grid is generated; in the language of the print/plot processor, the signal is generated as a function of Y with X as the parameter. If NX=1, a Z-Y grid is generated; in the language of the print/plot processor, the signal is generated as a function of Y with Z as the parameter. If NY=1, a Z-X grid is generated; the signal is a function of X with Z as the parameter.

NØBS	number of observation depths (z grid points). If greater than 1, the observation depth ØBSDEP is successively set to the values in the table of observation depths TABØBS(i), i=1,...,NØBS. The limit is NØBS ≤ 100. If NØBS = 1, an X-Y grid is assumed, the observation depth is input to OBSDEP and TABØBS is ignored.

OBSDEP	observation depth. Depth is positive and $0 \le$ ØBSDEP $\le$ ØCNDEP.
TABØBS(i)	(used only if NØBS > 1) list of observation depths. Depths are positive and $0 \le$ TABØBS(i) $\le$ ØCNDEP, i=1,...,NØBS. The first depth is input to TABØBS(1). If DØBS = OBSMAX = 0, then TABØBS(i), i=2,...,NØBS must also be input; otherwise the program fills in these entries.
DØBS	If non-zero, TABØBS(i) = TABØBS(1) + (i-1) * DØBS, i=2,...,NØBS. Otherwise, it is ignored.
ØBSMAX	If DØBS = 0 $\neq$ ØBSMAX, Δ = (ØBSMAX - TABØBS(1))/(NØBS-1) and TABØBS(i) = TABØBS(1) + (i-1) * Δ, i=2,...,NØBS. Otherwise it is ignored.
NX	number of downstream stations (X_i) in the grid. For a Z-Y grid, NX=1 and XMIN is the desired value of X. Otherwise NX, XMIN and DX define the X coordinates of the grid. If NX=0, the grid computation is skipped. (NX $\le$ 800).
XMIN	first value of X in the grid.
DX	grid increment in the X direction. Note X_i=XMIN + (i-1)*DX.
NY	number of grid points along the cross track (Y) axis. If the FFT option is used, NY must be a power of 2 and $1 <$ NY ≤ 256. If only the potential solution is used, $1 \le$ NY ≤ 2000. If the stationary phase option is used, $1 \le$ NY ≤ 800.
YMIN	first value of Y in the grid. This applies to stationary phase only. For near field, it is assumed YMIN=0.
DY	grid increment in the Y direction. Note Y_i = YMIN + (i-1)*DY.

5. Potential Solution Grid Limits

The grid defined above applies to the wave-like solution. The potential solution is evaluated at the same grid points subject to the restrictions imposed by XPMAX and YPMAX.

XPMAX	The potential solution will be evaluated only at grid points with coordinate $X_i \le$ XPMAX.
YPMAX	The potential solution will be evaluated only at grid points with coordinate $Y_i \le$ YPMAX.

6. Skip Grid

The capability exists to skip over the program module which performs the grid computations. Ordinarily, this would only be done when it was desired to use the print/plot module to display data which were generated earlier.

NOGRID = 0 : program determines grid requirements
= 1 : skip grid computation

7. Display

Note that the signal values at the grid points are always sent to the print/plot (pp) processor. The pp id for that display is CUTS.

IPPPSD = 0 : option off
= 1 : (FFT option only) power spectral density (PSD) data are sent to the pp processor. Note that the Fourier transform (from y to η) of the signal may be written

$$F(\eta,x) = \mathrm{Re}\ (f(\eta)e^{i\xi x}) \text{ or } F(\eta,x) = i\ \mathrm{Im}(f(\eta)e^{i\xi x});$$

the PSD is computed as $|f(\eta)|^2$. A PSD display is generated for each source per Figure 8.

Source	PP id
body	BPSD
wake	WPSD
super-structure	SPSD

Figure 8. PSD Displays

4.5 PP Data

This section describes the operation and input to the print/plot (PP) processor. The PP processor is capable of displaying a function of two variables f(p,v) where p is treated as a parameter and v is used as the independent variable. "Display" means print and/or plot.

When f(p,v) is printed, the value of f is listed for each value of v; such a list is generated for each value of p. It is also possible to produce a "summary print" which lists, for each value of p: the extrema of f, the values of v at which the extrema are attained,

$$\int f(p,v)\,dv \text{ and } \int f^2(p,v)\,dv.$$

There are two plot formats available. For a "multi-trace plot", the ordinate is f, the abscissa is v and one trace is drawn for each parameter value (see Figure 9). For a "raster plot" the ordinate is v, the abscissa is p and f is plotted as a displacement along an axis parallel to the abscissa (see Figure 10). No display will be generated for any value of p which has less than two values of v.

A display data set (DDS) consists of the values of f, p and v along with a preset format. During the computational phase of a case, DDS's are written on files TAPE9 and TAPE10. The various DDS's which may be written on these files are determined by input options described earlier and summarized in the first two columns of Figure 12. Each DDS (that is, each function which may be displayed) is assigned a 1 to 10 character identifier called the "id".

The standard option is to display all DDS's on the files based on the preset format assigned to each of them. However, it may be desirable to reformat some of the displays and eliminate others, while relying on the preset formats for the remaining DDS's. The variables in the PP namelist set (PP is the namelist name) allow the preset format of a DDS to be overridden (this includes a skip or no-display capability). With the desired

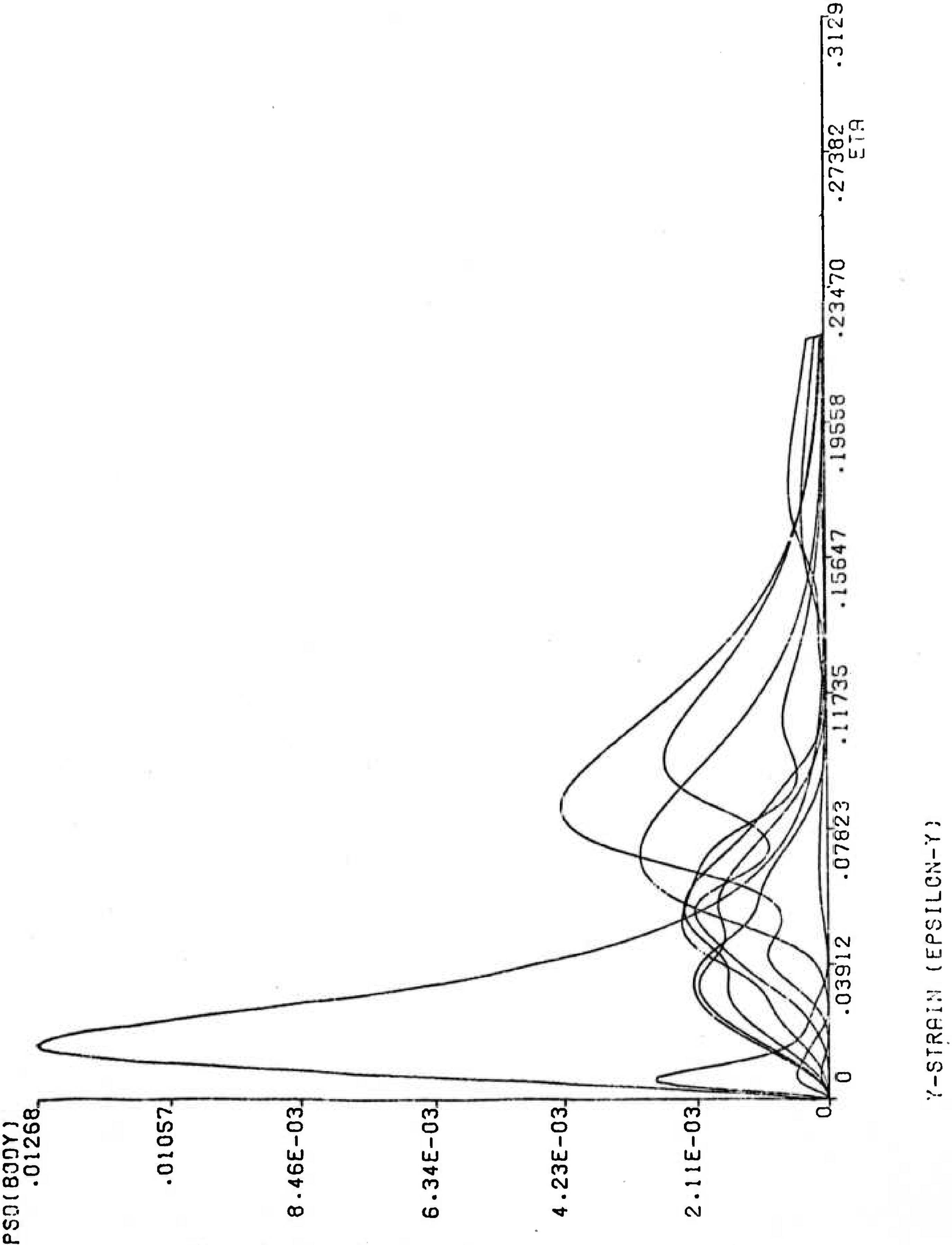


Figure 9. Example of a multi-trace plot.

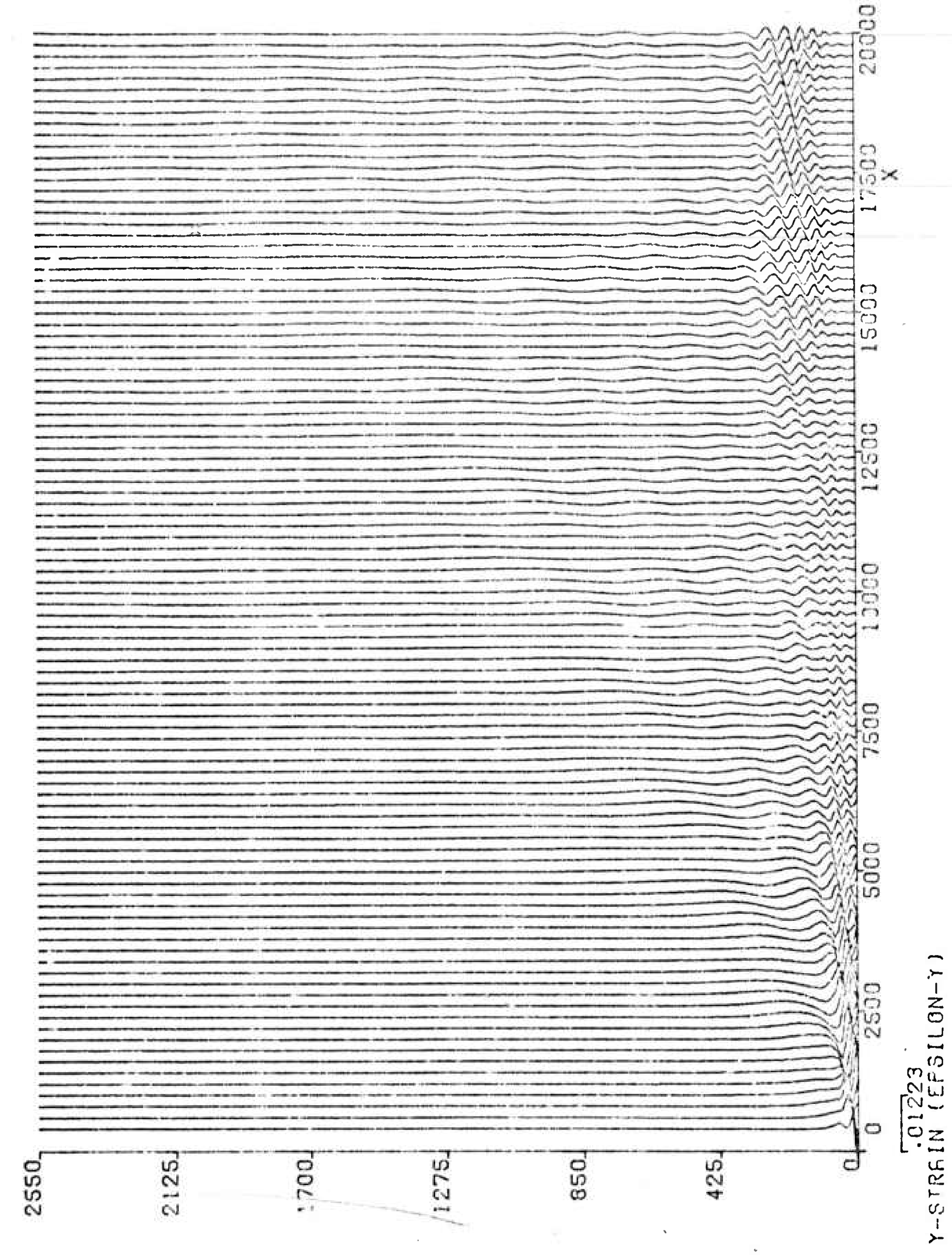


Figure 10. Example of a raster plot.

reformatting of a display reduced to namelist form, the problem is to identify which DDS on TAPE9 and TAPE10 is to be reformatted. The problem is compounded since, in a multi-case run, more than one DDS may have the same id.

The problem is solved by including in the PP namelist set two variables which act as locators. These locators select the DDS which is to be reformatted. The namelist set is thus composed of two parts:

1. the two locators
2. the remaining namelist variables, which constitute the print/plot format block (PPFB).

The two locators are named IDPP and IØCUR. The id of the desired DDS is entered as Hollerith data into IDPP. The locator IØCUR is used to differentiate between DDS's with the same identifier. Thus IØCUR=2 means the PPFB should be applied to the second DDS with the given id.

A PP namelist set must be input for each DDS which needs attention. If the locators (both of them) in one namelist set do not match the locators in any previous set, then a new PPFB is generated; that is, the PPFB is entered into a virgin array. If, on the other hand, both locators match with a previous namelist set, the PPFB is read in on top of (overlays) the matched PPFB.

Variable	Preset Value
IPRINT	2
IPLØT	1
IPLTYP	See Figure 12
ISYM	0
TITLE	See Figure 12
FNAME	See Figure 12
FMIN	min [f(p,v)]
FMAX	max [f(p,v)]
FLEN	8.
FTØDP	± 1. (minus if p = depth)
VNAME	See Figure 12
VMIN	min [v]
VMAX	max [v]
VLEN	10. (multi-trace), 8. (raster)
PNAME	See Figure 12
PMIN	min [p]
PMAX	max [p]
PLEN	10.
IEDIT	0
IØPP	0

Figure 11. Presets for PPFB Variables

IDPP	Input Option	FNAME	VNAME	PNAME	TITLE	IPLTYP	COMMENTS	
DL/DK	IPPDT(1)	DL/DK	ETA (fft)	MØDE	blanks	1 (multi-trace)	Independent variable with fft option is η; with stationary phase, it is K	
W(ØBS)	IPPDT(2)	W(ØBS)						
DW/DZ(ØBS)	IPPDT(3)	DW/DZ(ØBS)	K (stationary phase)					
DW/DZ(BØD)	IPPDT(4)	DW/DZ(BØD)						
TWAKE	IPPDT(5)	TWAKE						
TSUPR	IPPDT(6)	TSUPR						
LAMBDA	IPPDT(7)	LAMBDA	K				Available with stationary phase option only	
D2L/DK2	IPPDT(8)	D2L/DK2						
-Y/X	IPPDT(9)	-Y/X						
EVEC	IPPVEC	EIGENVECTOR	DEPTH	K	blanks	0 (raster)	IPPVEC = mode number	
BPSD	IPPPSD	PSD(BØDY)	ETA	MØDE	name selected by IVAR See Fig. 7	1 (multi-trace)	available with fft option only	
WPSD		PSD(WAKE)						
SPSD		PSD(SUPER)						
CUTS	automatic	function name selected by IVAR see Fig. 7	Y	X	blanks	0 (raster)	NØBS=1	fft
			Y	DEPTH			NX=1	
			-Y	X			NØBS=1	stationary phase
			-Y	DEPTH			NX=1	
			X	DEPTH			NY=1	

Figure 12. Available displays and specialized preset values.

The variables in the PP namelist set are defined below.

IDPP	1 to 10 character Hollerith identifier of the DDS to which this PPFB will apply. If not input, the PPFB will be applied to all displays except those specifically named in another PP namelist. Input of all blanks is equivalent to no input.
IOCUR	If not input, the PPFB will apply to all DDS's with the given IDPP except those which have IOCUR > 0. If IOCUR $= i > 0$, this PPFB will apply only to the i^{th} DDS which has the given IDPP. IOCUR $= 0$ is equivalent to no input. IOCUR < 0 is illegal. If IDPP was not input, IOCUR must not be input either.
IPRINT	= 0 : no print = 1 : print f(p,v) = 2 : print data summary (extrema, etc.) = 3 : print 1 + 2
IPLOT	= 0 : no plot = 1 : plot f(p,v)
IPLTYP	= 0 : produce raster plot (if plotting) = 1 : produce multi-trace plot (if plotting)
ISYM	applies only when generating a multi-trace plot = 0 : traces will not be labeled = 1 : a unique symbol will be drawn at the first and last point of each trace and keyed with the corresponding parameter value.
TITLE	a 20 character (2 word) title which will be included in the display (Hollerith)
FNAME	function name. This is a 20 character (2 word) Hollerith descriptor used to label the function in the display. An input to FNAME does not change the function, only its label.
FMIN	minimum value of f. This is used only for plot scaling; it is not used to limit the value of f. For a multi-trace plot, it is the value of f at the origin. For a raster plot, $f = 0$ at the origin.
FMAX	maximum value of f. This is used only for plot scaling; it is not used to limit the value of f. For a multi-trace plot, it is the value of f at the end of the f axis. For a raster plot, $\lvert f \rvert$ max = max ($\lvert$FMIN$\rvert$,$\lvert$FMAX$\rvert$) is the value of f at the end of the f axis.

FLEN	length in inches of the f axis. For a multi-trace plot, FLEN is always used. For a raster plot, it is used directly only if FTØDP = 0; otherwise the program uses FTØDP to compute FLEN.
FTØDP	(used only for raster plots to determine FLEN) if non-zero, it is the ratio of the length of the f axis to the average distance (in inches) between traces. Specifically, $\text{FLEN} = \text{FTØDP} * \left(\frac{\text{PLEN}}{N_p - 1}\right) * \left(\frac{\Delta p}{\text{PMAX-PMIN}}\right)$ where N_p is the number of parameter values in the DDS and Δp is the parameter range ($p_{max} - p_{min}$) in the DDS. If FTØDP = 0, it is ignored.
VNAME	a 10 character Hollerith descriptor used to label the independent variable in the display. An input to VNAME does not change the variable, only its label.
VMIN	minimum value of v. Data associated with values of $v < \text{VMIN}$ are discarded. VMIN is also used as the origin of the v axis for plotting.
VMAX	maximum value of v. Data associated with values of $v > \text{VMAX}$ are discarded. VMAX is also used as the value of v at the end of the v axis.
VLEN	length in inches of the v axis.
PNAME	a 10 character Hollerith descriptor used to label the parameter in the display. An input to PNAME does not change the parameter only its label.
PMIN	minimum value of p. Data associated with values of $p < \text{PMIN}$ are discarded. For a raster plot, PMIN is also used as the origin of the p axis.
PMAX	maximum value of p. Data associated with values of $p > \text{PMAX}$ are discarded. For a raster plot, PMAX is also used as the value of p at the end of the p axis.
PLEN	length in inches of the p axis for a raster plot (multi-trace plots do not have a p axis).
IEDIT	Ordinarily, all DDS's on TAPE9 and TAPE10 are displayed, except those for which a PPFB was input with IPRINT = IPLØT = 0. However, if IEDIT = 1 in <u>any</u> PPFB, then the only DDS's which will be displayed are those for which a PPFB was input. Replacing the IEDIT = 1 with IEDIT = 0 restores the nominal option.

NØPP

Ordinarily, the DDS's generated during the computational phase of a case are displayed at the end of that case. The files TAPE9 and TAPE10 are then rewound and display data generated in the next case overwrites the old data. However, if NØPP = 1 in <u>any</u> PPFB, no displays are generated and the positions of TAPE9 and TAPE10 are undisturbed. During a multi-case run, this permits all the display data to be accumulated on these files. Presumably, in the last case of the run, the NØPP = 1 will be replaced with NØPP = 0 to display the accumulated data. The advantage in accumulating data is that TAPE9 and TAPE10 can be saved and a subsequent run made to reformat the displays. The disadvantage is that TAPE9 can become quite large.

5. SAMPLE CALCULATIONS

This section presents a series of three computer runs designed to illustrate the operation of the SEEK program. The data are fictitious but not unreasonable. The parameters were chosen in a cavalier manner in order to point out common sources of error.

In the first run, the fast Fourier transform technique is used to compute the disturbance due to a Rankine body with a wake. Both the y-strain (ϵ_y) and y-velocity (v) are displayed. The ocean, body and wake descriptions are on the data library file, TAPE1. Other input data are introduced from the INPUT file. Processed ocean data on TAPE2 are saved at the end of the run. PP data (TAPE9, TAPE10) for the last case are also saved at the end of the run.

For the second run, files TAPE9 and TAPE10 are restored and the program is used to reformat the displays associated with the last case of the first run.

The third run uses the processed ocean data generated in the first run to extend the grid beyond the point at which aliasing occurs in the FFT solution.

5.1 Run 1

The next two pages are a listing of the input data library file (TAPE1). Note that three data sets are on the file. The first data set (introduced by *S,B1) defines a Rankine body. The second (introduced by *S,W1) defines a wake model. The third data set (introduced by *0,S2) defines the ocean and the dispersion tables.

The P which precedes each namelist name (e.g., P$S0URCE) activates a special TRW modification to NAMELIST which causes the card images to be printed as they are read.

It should also be noted that TRW's system presets memory to zero before execution. For installations for which this is not true, it is recommended that the input data library contain data sets which set all input variables to zero. These data sets can be called in at the start of a run by LIB cards to effect the preset.

At the end of the run, TAPE2 (processed ocean data) was saved, along with TAPE9 and TAPE10 (display data for the second case of the run).

```
*S,B1
P$SOURCE   BODDIA=10, BODLEN=100, IBODY=1$
*S,W1
P$SOURCE   IWAKE=1, CWAKR=.9, CWAKX=.2, RESLVS=5, CWAKM=.2$
*O,S2
P$OCEAN  MODES=9, NK=100, NZT=151, DKRAT=5, DTDEP=4, NFLAG=0
 OCNDEP=1000, RKMAX=.22, IPRDT=0, LIBSEA=0, TABK=0
 TDEPMX=0, TDEP=0, NODISP=0, IPPVEC=0, IPREDG=0,
 SQBV=
 0,                2.805472871E-05, 5.599502042E-05,
 8.042934824E-05, 1.016400016E-04, 1.198934558E-04, 1.354409315E-04,
 1.485189537E-04, 1.593499053E-04, 1.681425756E-04, 1.750927026E-04,
 1.803835062E-04, 1.841862130E-04, 1.866605716E-04, 1.879553565E-04,
 1.882088613E-04, 1.875493810E-04, 1.860956812E-04, 1.839574563E-04,
 1.812357746E-04, 1.780235112E-04, 1.744057686E-04, 1.704602839E-04,
 1.662578250E-04, 1.618625725E-04, 1.573324906E-04, 1.527196852E-04,
 1.430707496E-04, 1.434270989E-04, 1.388252919E-04, 1.342973418E-04,
 1.298710152E-04, 1.255701202E-04, 1.214147826E-04, 1.174217125E-04,
 1.136044590E-04, 1.099736555E-04, 1.065372538E-04, 1.033007491E-04,
 1.002673943E-04, 9.743840522E-05, 9.481315641E-05, 9.238936736E-05,
 9.016327998E-05, 8.812982720E-05, 8.628279287E-05, 8.461496328E-05,
 8.311827042E-05, 8.178392716E-05, 8.060255460E-05, 7.956430173E-05,
 7.865895756E-05, 7.787605600E-05, 7.720495356E-05, 7.663502018E-05,
 7.615552323E-05, 7.575590499E-05, 7.542575370E-05, 7.515488845E-05,
 7.493341799E-05, 7.475179371E-05, 7.460085701E-05, 7.447188109E-05,
 7.435660753E-05, 7.424727770E-05, 7.413665925E-05, 7.401806783E-05,
 7.388538428E-05, 7.373306738E-05, 7.355616243E-05, 7.335030577E-05,
 7.311172551E-05, 7.283723854E-05, 7.252424410E-05, 7.217071406E-05,
 7.177518009E-05, 7.133671786E-05, 7.085492855E-05, 7.032991779E-05,
 6.976227217E-05, 6.915303364E-05, 6.850367179E-05, 6.781605442E-05,
 6.709241535E-05, 6.633532684E-05, 6.554765568E-05, 6.473253821E-05,
 6.389333936E-05, 6.303361708E-05, 6.215708504E-05, 6.126757508E-05,
 6.036899941E-05, 5.946531284E-05, 5.856047510E-05, 5.765841359E-05,
 5.676298661E-05, 5.587794735E-05, 5.500690866E-05, 5.415330904E-05,
 5.332037978E-05, 5.251111359E-05, 5.172823477E-05, 5.097417127E-05,
 5.025102870E-05, 4.956056641E-05, 4.890417606E-05, 4.828286260E-05,
 4.769722798E-05, 4.714745777E-05, 4.663331081E-05, 4.615411214E-05,
 4.570874932E-05, 4.529567239E-05, 4.491289761E-05, 4.455801519E-05,
 4.422820113E-05, 4.392023344E-05, 4.363051286E-05, 4.335508826E-05,
 4.308968696E-05, 4.282975004E-05, 4.257047299E-05, 4.230685168E-05,
 4.203373399E-05, 4.174587721E-05, 4.143801139E-05, 4.110490886E-05,
```

```
 4.074146004E-05, 4.034275576E-05, 3.990417632E-05, 3.942148731E-05,
 3.889094259E-05, 3.830939442E-05, 3.767441105E-05, 3.698440184E-05,
 3.623875023E-05, 3.543795460E-05, 3.458377734E-05, 3.367940215E-05,
 3.272959995E-05, 3.174090339E-05, 3.072179029E-05, 2.968287615E-05,
 2.863711581E-05, 2.760001466E-05, 2.658984935E-05, 2.562789837E-05,
 2.473868251E-05, 2.395021552E-05, 2.329426503E-05, 2.280662399E-05$
*END
```

The next page is a listing of the data deck (file INPUT) used for the first run.

Note that the S0URCE data are the result of three different reads, each data set being superimposed on the previous data.

The GRID data specify an x-y grid (N0BS=1) at the surface (0BSDEP=0). The grid variable to be computed is y-strain, ϵ_y (IVAR=7). Since ISPHAS=0, the FFT technique will be used. IPPPSD=1 causes the power spectra to be displayed.

The first PP data set calls for a special format for the grid display (IDPP=4HCUTS). The x-axis is to start at zero (PMIN=0) rather than the default value of 400 (since XMIN=400). The lengths of the y and x axes have been decreased so the plot will fit on 8½ x 11 paper. The function scaling has been increased by a factor of 5 (FT0DP=5) to make the disturbance pattern more apparent. Since the second PP data set has no entry to IDPP, it applies to all displays except CUTS. The only other displays in this instance are the power spectra. The lengths of the axes are reduced to fit on 8½ x 11 paper.

The RUN card causes the first case to be executed. The second case is the same as the first except y-velocity, v (IVAR=2) will be computed and the grid increment in y is different. Also, the plot of the wake PSD will have a label on each trace.

```
LIB,O,S2
LIB,S,B1
LIB,S,W1
INP,S
P$SOURCE  BODDEP=45, BODSPC=2$
INP,G
P$GRID  OBSDEP=0, NOBS=1, DX=200, XMIN=400, NX=99, DY=10, NY=256,
      MODE1=1, MODEN=9, IVAR=7, IPPPSD=1$
INP,P
P$PP  IDPP=4HCUTS, PMIN=0, VLEN=6, PLEN=8, FTODP=5$
INP,P
P$PP  FLEN=6, VLEN=8$
RUN
INP,G
P$GRID  IVAR=2, DY=13$
INP,P
P$PP IDPP=4HWPSD, ISYM=1, FLEN=6, VLEN=8$
RUN
END
```

This is the output from the first run.

input data
card images

```
LIB,0,S2
$OCEAN  MODES=9, NK=100, NZT=151, DKRAT=5, DTDEP=4, NFLAG=0
OCNDEP=1000, RKMAX=.22, IPRDT=0, LIBSEA=0, TABK=0
TDEPMX=0, TDEP=0, NODISP=0, IPRVEC=0, IPREDG=0,
SQBV=
0,               2.805472871E-05, 5.599502042E-05,
8.042934824E-05, 1.016400016E-04, 1.198934558E-04, 1.354409315E-04,
1.485189537E-04, 1.593499053E-04, 1.681425756E-04, 1.750927026E-04,
1.803835062E-04, 1.841862130E-04, 1.866605716E-04, 1.879553565E-04,
1.882088613E-04, 1.875493810E-04, 1.860956812E-04, 1.839574563E-04,
1.812357746E-04, 1.780235112E-04, 1.744057686E-04, 1.704602839E-04,
1.662578250E-04, 1.618625725E-04, 1.573324906E-04, 1.527196852E-04,
1.480707496E-04, 1.434270989E-04, 1.388252919E-04, 1.342973418E-04,
1.298710152E-04, 1.255701202E-04, 1.214147826E-04, 1.174217125E-04,
1.136044590E-04, 1.099736555E-04, 1.065372538E-04, 1.033007491E-04,
1.002573943E-04, 9.743840522E-05, 9.481315641E-05, 9.238936736E-05,
```

```
9.016327998E-05, 8.812982720E-05, 8.628279287E-05, 8.461496328E-05,
8.311827042E-05, 8.178392716E-05, 8.060255460E-05, 7.956430173E-05,
7.865895756E-05, 7.787605600E-05, 7.720495356E-05, 7.663502018E-05,
7.615552323E-05, 7.575590499E-05, 7.542575370E-05, 7.515488845E-05,
7.493341799E-05, 7.475179371E-05, 7.460085701E-05, 7.447188109E-05,
7.435660753E-05, 7.424727770E-05, 7.413665925E-05, 7.401806783E-05,
7.388538428E-05, 7.373306738E-05, 7.355616243E-05, 7.335030577E-05,
7.311172551E-05, 7.283723854E-05, 7.252424410E-05, 7.217071406E-05,
7.177518009E-05, 7.133671786E-05, 7.085492855E-05, 7.032991779E-05,
6.976227217E-05, 6.915303364E-05, 6.850367179E-05, 6.781605442E-05,
6.709241635E-05, 6.633532684E-05, 6.554765568E-05, 6.473253821E-05,
6.389333936E-05, 6.303361708E-05, 6.215708504E-05, 6.126757508E-05,
6.035899941E-05, 5.946531284E-05, 5.856047510E-05, 5.765841359E-05,
5.676298661E-05, 5.587794735E-05, 5.500690866E-05, 5.415330904E-05,
5.332037978E-05, 5.251111359E-05, 5.172823477E-05, 5.097417127E-05,
5.025102870E-05, 4.956056641E-05, 4.890417606E-05, 4.628286260E-05,
4.769722798E-05, 4.714745777E-05, 4.663331081E-05, 4.615411214E-05,
4.570874932E-05, 4.529567239E-05, 4.491289761E-05, 4.455801519E-05,
4.422820113E-05, 4.392023344E-05, 4.363051286E-05, 4.335508826E-05,
4.308968696E-05, 4.282975004E-05, 4.257047299E-05, 4.230685168E-05,
4.203373399E-05, 4.174587721E-05, 4.143801139E-05, 4.110490886E-05,
4.074146004E-05, 4.034275576E-05, 3.990417632E-05, 3.942148731E-05,
3.889094259E-05, 3.830939442E-05, 3.767441105E-05, 3.698440184E-05,
3.623875023E-05, 3.543795460E-05, 3.458377734E-05, 3.367940215E-05,
3.272959995E-05, 3.174090339E-05, 3.072179029E-05, 2.968287615E-05,
2.863711581E-05, 2.760001466E-05, 2.658984935E-05, 2.562789837E-05,
2.473868251E-05, 2.395021552E-05, 2.329426503E-05, 2.280662399E-05$
LIB,S,B1
$SOURCE  BODDIA=10, BODLEN=100, IBODY=1$
LIB,S,W1
$SOURCE  IWAKE=1, CWAKR=.9, CWAKX=.2, RESLVS=5, CWAKM=.2$
INP,S
$SOURCE  BODDEP=45, BODSPD=2$
INP,G
$GRID  OBSDEP=0, NOBS=1, DX=200, XMIN=400, NX=99, DY=10, NY=256,
     MODE1=1, MODEN=9, IVAR=7, IPPPSD=1$
INP,P
$PP  IDPP=4HCUTS, PMIN=0, VLEN=6, PLEN=8, FTODP=5$
INP,P
$PP  FLEN=6, VLEN=8$
RUN
```

end of input data for
first case

```
FUNCTION=PSD(BODY)                                              Y-STRAIN (EPSILON-Y)                    BPSD

       VALUE OF      MAXIMUM OF      ETA             MINIMUM OF      ETA               INTEGRAL      SQUARE
       MODE          FUNCTION        OF MAX          FUNCTION        OF MIN                          INTEGRAL
 1              1    1.268350E-02    1.595340E-02    0.              2.208932E-01      4.0994E-04    3.2921E-06
 2              2    2.774154E-03    4.908739E-03    0.              2.208932E-01      3.7436E-05    5.4509E-08
 3              3    3.018039E-03    7.117671E-02    0.              2.208932E-01      1.9222E-04    3.8781E-07
 4              4    4.265357E-03    8.344855E-02    0.              2.208932E-01      2.9193E-04    7.992[illegible]E-07
 5              5    2.629050E-03    9.817477E-02    0.              2.208932E-01      2.3022E-04    3.9311E-07
 6              6    2.353725E-03    5.154175E-02    0.              2.208932E-01      1.5177E-04    2.2428E-07
 7              7    2.314894E-03    5.522331E-02    0.              2.208932E-01      1.3577E-04    2.1766E-07
 8              8    2.075821E-03    3.313399E-02    0.              2.208932E-01      1.4551E-04    1.9392E-07
 9              9    2.165217E-03    3.313399E-02    0.              2.208932E-01      1.3573E-04    1.6832E-07
```

FNAME

VNAME

TITLE

IDPP

PNAME

Significant contributions from higher modes are apparent here. A large contribution is missing near the track due to modal truncation.

SUMMARY PRINT FOR BODY PSD.

```
FUNCTION=PSD(WAKE)                                    Y-STRAIN (EPSILON-Y)                             WPSD

      VALUE OF      MAXIMUM OF      ETA            MINIMUM OF      ETA                                    SQUARE
      MODE          FUNCTION        OF MAX         FUNCTION        OF MIN                INTEGRAL          INTEGRAL
   1          1     3.728019E-04    5.767768E-02   0.              2.208932E-01          3.2148E-05        8.6859E-09
   2          2     1.805806E-04    8.344855E-02   0.              2.208932E-01          1.1961E-05        1.4544E-09
   3          3     6.994878E-03    8.835729E-02   0.              2.208932E-01          5.3510E-04        2.6780E-06
   4          4     1.911711E-02    9.572040E-02   0.              2.208932E-01          1.4292E-03        1.9182E-05
   5          5     1.890187E-02    1.030835E-01   0.              2.208932E-01          1.3723E-03        1.6628E-05
   6          6     9.352480E-03    7.117671E-02   0.              2.208932E-01          7.3147E-04        4.3520E-06
   7          7     1.098521E-02    6.258642E-02   0.              2.208932E-01          6.3873E-04        4.6171E-06
   8          8     1.095096E-02    6.258642E-02   0.              2.208932E-01          9.2739E-04        5.9864E-06
   9          9     1.090747E-02    3.681554E-02   0.              2.208932E-01          1.0719E-03        7.8906E-06
```

Large contribution from higher modes is missing.

```
FUNCTION=Y-STRAIN (EPSILON-Y)                                                                          CJTS

         VALUE OF      MAXIMUM OF      Y                 MINIMUM OF       Y                                       SQUARE
         X             FUNCTION        OF MAX            FUNCTION         OF MIN               INTEGRAL          INTEGRAL
   1           400     5.608127E-03    0.                -2.029639E-04    1.600000E+02         8.5873E-02        5.0598E-04
   2           500     5.441450E-03    3.000000E+01      -6.395762E-03    0.                   8.5887E-02        9.2269E-04
   3           800     6.978813E-03    3.000000E+01      -1.222528E-02    0.                   8.5899E-02        2.3933E-03
   4          1000     5.068187E-03    4.000000E+01      -8.231572E-03    0.                   8.5907E-02        1.5597E-03
   5          1200     2.290723E-03    5.000000E+01      -5.088173E-03    2.000000E+01         8.5902E-02        5.7792E-04
   6          1400     6.486435E-03    0.                -6.422745E-03    3.000000E+01         8.5884E-02        1.0503E-03
   7          1600     5.181528E-03    0.                -6.035420E-03    3.000000E+01         8.5864E-02        1.1925E-03
   8          1800     4.226421E-03    2.000000E+01      -5.557813E-03    4.000000E+01         8.5853E-02        8.1345E-04
   9          2000     3.946343E-03    2.000000E+01      -4.465501E-03    5.000000E+01         8.5849E-02        7.1914E-04
  10          2200     5.302210E-03    3.000000E+01      -3.601485E-03    6.000000E+01         8.5833E-02        8.2262E-04
  11          2400     5.556462E-03    4.000000E+01      -3.015507E-03    2.000000E+01         8.5812E-02        9.0312E-04
  12          2600     4.316599E-03    5.000000E+01      -3.231857E-03    2.000000E+01         8.5784E-02        7.7949E-04
  13          2800     4.952786E-03    5.000000E+01      -3.424235E-03    3.000000E+01         8.5783E-02        6.9052E-04
  14          3000     4.099985E-03    6.000000E+01      -3.619099E-03    4.000000E+01         8.5826E-02        7.4063E-04
  15          3200     2.832414E-03    7.000000E+01      -4.264741E-03    4.000000E+01         8.5907E-02        5.6619E-04
  16          3400     2.014038E-03    3.000000E+01      -4.399669E-03    5.000000E+01         8.5999E-02        4.1576E-04
  17          3600     3.454869E-03    0.                -3.245094E-03    5.000000E+01         8.5079E-02        4.9394E-04
  18          3800     3.229947E-03    4.000000E+01      -3.167805E-03    6.000000E+01         8.6145E-02        4.3279E-04
  19          4000     2.687821E-03    5.000000E+01      -2.407857E-03    7.000000E+01         8.6205E-02        3.3870E-04
  20          4200     2.812150E-03    5.000000E+01      -2.471058E-03    0.                   8.6250E-02        4.2776E-04
  21          4400     2.447103E-03    6.000000E+01      -2.539776E-03    0.                   8.6254E-02        4.5033E-04
  22          4600     2.017261E-03    6.000000E+01      -1.534827E-03    4.000000E+01         8.5188E-02        3.6481E-04
  23          4800     2.139118E-03    7.000000E+01      -1.905166E-03    5.000000E+01         8.6050E-02        3.4295E-04
  24          5000     1.909822E-03    8.000000E+01      -2.115389E-03    2.000000E+01         8.5870E-02        3.2670E-04
  25          5200     2.001549E-03    8.000000E+01      -1.858744E-03    6.000000E+01         8.5599E-02        2.5142E-04
  26          5400     1.475201E-03    9.000000E+01      -1.719850E-03    3.000000E+01         8.5577E-02        1.9622E-04
  27          5600     1.492263E-03    2.000000E+01      -1.843061E-03    7.000000E+01         8.5522E-02        1.9761E-04
  28          5800     1.611606E-03    2.000000E+01      -1.903947E-03    8.000000E+01         8.5530E-02        2.0851E-04
  29          6000     1.416256E-03    6.000000E+01      -1.340792E-03    6.000000E+01         8.5585E-02        2.0192E-04
  30          6200     1.515008E-03    7.000000E+01      -1.208707E-03    5.000000E+01         8.5572E-02        2.1033E-04
  31          6400     1.247628E-03    0.                -1.268696E-03    2.000000E+01         8.5767E-02        2.2965E-04
  32          6600     1.258718E-03    8.000000E+01      -1.442649E-03    6.000000E+01         8.5842E-02        2.1193E-04
  33          6800     1.247175E-03    5.000000E+01      -1.050326E-03    7.000000E+01         8.5874E-02        1.7070E-04
  34          7000     1.140783E-03    5.000000E+01      -9.500394E-04    7.000000E+01         8.5851E-02        1.4288E-04
  35          7200     1.374349E-03    6.000000E+01      -1.277618E-03    4.000000E+01         8.5794E-02        1.2182E-04
  36          7400     9.147203E-04    6.000000E+01      -1.028469E-03    4.000000E+01         8.5750E-02        9.8063E-05
  37          7600     8.915520E-04    3.000000E+01      -1.313700E-03    5.000000E+01         8.5782E-02        8.8967E-05
```

38	7800	1.006190E-03	4.000000E+01	-1.098868E-03	6.000000E+01	8.5930E-02	1.0321E-04
39	8000	1.182060E-03	4.000000E+01	-1.113086E-03	6.000000E+01	8.6189E-02	1.1868E-04
40	8200	1.070622E-03	5.000000E+01	-8.642943E-04	3.000000E+01	8.6495E-02	1.2131E-04
41	8400	8.150841E-04	5.000000E+01	-7.662449E-04	3.000000E+01	8.6739E-02	1.2174E-04
42	8600	9.698890E-04	6.000000E+01	-1.040709E-03	4.000000E+01	8.6808E-02	1.2185E-04
43	8800	7.560990E-04	6.000000E+01	-7.449086E-04	2.200000E+02	8.6628E-02	1.0947E-04
44	9000	8.676926E-04	1.700000E+02	-8.041308E-04	5.000000E+01	8.6199E-02	9.5193E-05
45	9200	9.148520E-04	1.900000E+02	-8.103912E-04	1.500000E+02	8.5512E-02	1.0027E-04
46	9400	8.208848E-04	2.000000E+02	-9.322757E-04	1.600000E+02	8.5029E-02	1.1483E-04
47	9600	7.423811E-04	1.200000E+02	-9.625347E-04	1.700000E+02	8.4632E-02	1.2531E-04
48	9800	8.114450E-04	1.400000E+02	-8.213171E-04	1.800000E+02	8.4565E-02	1.3508E-04
49	10000	9.400127E-04	1.500000E+02	-7.254291E-04	1.100000E+02	8.4877E-02	1.4142E-04
50	10200	8.619420E-04	1.600000E+02	-7.863352E-04	1.200000E+02	8.5489E-02	1.3762E-04
51	10400	6.962145E-04	1.700000E+02	-6.373906E-04	1.300000E+02	8.6215E-02	1.2551E-04
52	10600	8.382511E-04	1.100000E+02	-7.676157E-04	9.000000E+01	8.6818E-02	1.1262E-04
53	10800	7.061511E-04	1.200000E+02	-7.919409E-04	1.000000E+02	8.7092E-02	1.0124E-04
54	11000	8.056999E-04	8.000000E+01	-8.867655E-04	1.000000E+02	8.6951E-02	8.9755E-05
55	11200	9.780937E-04	9.000000E+01	-8.419774E-04	1.100000E+02	8.6468E-02	8.3631E-05
56	11400	8.762872E-04	9.000000E+01	-6.556464E-04	7.000000E+01	8.5855E-02	8.9517E-05
57	11600	1.011059E-03	1.000000E+02	-1.001105E-03	8.000000E+01	8.5378E-02	9.9570E-05
58	11800	7.901273E-04	1.100000E+02	-9.113742E-04	9.000000E+01	8.5245E-02	1.0148E-04
59	12000	9.028487E-04	7.000000E+01	-9.426521E-04	9.000000E+01	8.5508E-02	9.8480E-05
60	12200	1.023167E-03	8.000000E+01	-8.219198E-04	1.000000E+02	8.6041E-02	9.4661E-05
61	12400	8.509638E-04	8.000000E+01	-6.125825E-04	6.000000E+01	8.6590E-02	8.7112E-05
62	12600	8.504120E-04	9.000000E+01	-8.073002E-04	7.000000E+01	8.6891E-02	8.0844E-05
63	12800	5.963200E-04	1.700000E+02	-7.303215E-04	2.100000E+02	8.6790E-02	8.2435E-05
64	13000	7.332314E-04	1.800000E+02	-8.869323E-04	2.200000E+02	8.6321E-02	9.1709E-05
65	13200	9.299810E-04	1.900000E+02	-9.553981E-04	2.300000E+02	8.5690E-02	1.0629E-04
66	13400	1.038969E-03	2.000000E+02	-9.098434E-04	1.700000E+02	8.5175E-02	1.2257E-04
67	13600	1.055917E-03	2.100000E+02	-1.117714E-03	1.800000E+02	8.4988E-02	1.3630E-04
68	13800	1.026990E-03	2.300000E+02	-1.315513E-03	1.900000E+02	8.5162E-02	1.4581E-04
69	14000	9.611214E-04	2.400000E+02	-1.374836E-03	2.000000E+02	8.5544E-02	1.5119E-04
70	14200	1.157086E-03	1.800000E+02	-1.325681E-03	2.100000E+02	8.5881E-02	1.5448E-04
71	14400	1.315773E-03	1.900000E+02	-1.189962E-03	2.200000E+02	8.5968E-02	1.6073E-04
72	14600	1.328345E-03	2.000000E+02	-1.083270E-03	1.600000E+02	8.5773E-02	1.7084E-04
73	14800	1.244165E-03	2.100000E+02	-1.142313E-03	1.800000E+02	8.5465E-02	1.8221E-04
74	15000	1.170026E-03	1.500000E+02	-1.193354E-03	1.800000E+02	8.5322E-02	1.9508E-04
75	15200	1.228968E-03	1.600000E+02	-1.243825E-03	1.900000E+02	8.5554E-02	2.0906E-04
76	15400	1.259792E-03	1.700000E+02	-1.236317E-03	2.000000E+02	8.6150E-02	2.2179E-04
77	15600	1.340464E-03	1.300000E+02	-1.259039E-03	1.500000E+02	8.6828E-02	2.3215E-04
78	15800	1.333822E-03	1.900000E+02	-1.318014E-03	1.600000E+02	8.7167E-02	2.3859E-04

```
79    16000   1.319673E-03   2.000000E+02   -1.406984E-03   1.200000E+02    8.6832E-02   2.3890E-04
80    15200   1.309146E-03   2.100000E+02   -1.447247E-03   1.800000E+02    8.5831E-02   2.3832E-04
81    15400   1.329014E-03   2.200000E+02   -1.484134E-03   1.900000E+02    8.4438E-02   2.4521E-04
82    16600   1.417492E-03   1.700000E+02   -1.473515E-03   2.000000E+02    8.3356E-02   2.5373E-04
83    16800   1.515062E-03   1.800000E+02   -1.465062E-03   2.100000E+02    8.3111E-02   2.5154E-04
84    17000   1.550047E-03   1.900000E+02   -1.490451E-03   2.200000E+02    8.3889E-02   2.4336E-04
85    17200   1.527190E-03   2.000000E+02   -1.527710E-03   2.300000E+02    8.5363E-02   2.4223E-04
86    17400   1.499813E-03   2.100000E+02   -1.634329E-03   1.800000E+02    8.6342E-02   2.4377E-04
87    17600   1.540623E-03   2.100000E+02   -1.644815E-03   1.900000E+02    8.7586E-02   2.3635E-04
88    17800   1.535585E-03   2.200000E+02   -1.574402E-03   2.000000E+02    8.7253E-02   2.2374E-04
89    18000   1.649208E-03   1.800000E+02   -1.727035E-03   2.000000E+02    8.5080E-02   2.1796E-04
90    18200   1.633143E-03   1.900000E+02   -1.771063E-03   2.100000E+02    8.4757E-02   2.2298E-04
91    18400   1.813364E-03   1.900000E+02   -1.705145E-03   2.200000E+02    8.4027E-02   2.3185E-04
92    18600   1.961748E-03   2.000000E+02   -1.616692E-03   2.300000E+02    8.4248E-02   2.4001E-04
93    18800   1.945226E-03   2.100000E+02   -1.518645E-03   1.800000E+02    8.5133E-02   2.5479E-04
94    19000   1.834804E-03   2.200000E+02   -1.841818E-03   1.900000E+02    8.6157E-02   2.7810E-04
95    19200   1.693058E-03   2.300000E+02   -1.939212E-03   2.000000E+02    8.6490E-02   2.9771E-04
96    19400   1.645844E-03   1.800000E+02   -1.888996E-03   2.100000E+02    8.5934E-02   3.0786E-04
97    19600   1.818871E-03   1.900000E+02   -1.841214E-03   1.700000E+02    8.4337E-02   3.1397E-04
98    19800   1.791067E-03   2.000000E+02   -1.673257E-03   1.800000E+02    8.3923E-02   3.1488E-04
99    20000   1.662545E-03   1.600000E+02   -1.812141E-03   1.800000E+02    8.3811E-02   3.0865E-04
```

```
CP TIMES                      the program was run on a CDC Cyber 74 (6600)
CASE          149.978         grand total for case
INCON            .132         total input processing
DYCON         132.039         total dispersion table generation
DTEVAL         88.299               eigenvalue
DTEVEC         20.992               eigenvector
DTDER2         19.425               d^2λ/dK^2
DTWAKE          2.709               wake integral
FTCON          12.621         total grid computation (FFT)
FTDTAB           .173               final adjustments on dispersion table
FTGEND           .546               transforms without x-dependence
FTNEWX          3.455               transforms and modal summation for all x
FTFFT           6.585               fast Fourier transform for all x
FTPOT           0.000               potential solution
PPCON           5.181         total print/plot processing
SPCON           0.000         total grid computation (stationary phase)
SPDTAB          0.000               final adjustments on dispersion table
SPWFAM          0.000               wave family edge table
SP1PNT          0.000               stationary phase grid                     end of case 1
INP,G
$GRID  IVAR=2, DY=13$                           input data card
INP,P
$PP IDPP=4HWPSD, ISYM=1, FLEN=6, VLEN=8$        images for case 2
RUN
```

```
FUNCTION=PSD(BODY)                                Y-VELOCITY (V)                                 BPSD

      VALUE OF       MAXIMUM OF       ETA              MINIMUM OF      ETA                                  SQUARE
      MODE           FUNCTION        OF MAX            FUNCTION        OF MIN             INTEGRAL         INTEGRAL
 1           1     1.245491E-02     0.                 0.                2.199492E-01    1.7409E-04     1.2038E-06
 2           2     2.325117E-03     3.775953E-03       0.                2.199492E-01    1.7374E-05     2.6604E-08
 3           3     1.778367E-04     5.663929E-03       0.                2.199492E-01    4.7682E-06     3.4028E-10
 4           4     5.411790E-05     7.740703E-02       0.                2.199492E-01    3.7714E-06     1.2306E-10
 5           5     4.818974E-05     4.625542E-02       0.                2.199492E-01    3.0263E-06     8.8809E-11
 6           6     5.181420E-05     4.436744E-02       0.                2.199492E-01    2.8369E-06     1.0028E-10
 7           7     5.561819E-05     2.548768E-02       0.                2.199492E-01    2.7935E-06     1.0699E-10
 8           8     5.873266E-05     2.643167E-02       0.                2.199492E-01    2.5539E-06     9.7498E-11
 9           9     5.217434E-05     2.737566E-02       0.                2.199492E-01    2.1082E-06     7.3125E-11
```

Disturbance seems to be dominated by lower modes, but contributions from higher modes, while small, are not decreasing with increasing mode number. Prudence would indicate a need for more modes to demonstrate convergence.

```
FUNCTION=PSD(WAKE)                                     Y-VELOCITY (V)                                  WPSD

      VALUE OF        MAXIMUM OF      ETA            MINIMUM OF       ETA
      MODE            FUNCTION        OF MAX         FUNCTION         OF MIN                INTEGRAL       SQUARE
                                                                                                          INTEGRAL
  1           1      3.739254E-05   2.265572E-02   0.              2.199492E-01          1.9537E-06     4.7908E-11
  2           2      3.265910E-05   0.             0.              2.199492E-01          4.7322E-07     5.2605E-12
  3           3      1.063202E-04   7.363108E-02   0.              2.199492E-01          6.8515E-06     4.9956E-10
  4           4      2.072930E-04   8.495894E-02   0.              2.199492E-01          1.3236E-05     1.7960E-09
  5           5      1.527658E-04   9.723078E-02   0.              2.199492E-01          1.1526E-05     1.0967E-09
  6           6      1.349049E-04   5.286334E-02   0.              2.199492E-01          8.9145E-06     7.6983E-10
  7           7      1.619398E-04   5.569530E-02   0.              2.199492E-01          1.0455E-05     1.1773E-09
  8           8      1.900718E-04   2.454369E-02   0.              2.199492E-01          1.3712E-05     1.9965E-09
  9           9      3.023964E-04   7.551905E-03   0.              2.199492E-01          1.6474E-05     3.6179E-09
```

Significant contributions from higher modes are apparent here. The wake disturbance as calculated using 9 modes can be expected to be significantly in error near the track.

```
FUNCTION=Y-VELOCITY (V)                                                                                  CUTS

     VALUE OF    MAXIMUM OF      Y              MINIMUM OF      Y
     X           FUNCTION        OF MAX         FUNCTION        OF MIN              INTEGRAL        SQUARE
                                                                                                    INTEGRAL
  1   400   1.526650E-04   2.080000E+02   -1.228648E-03   1.300000E+01    5.4883E-02   4.8850E-05
  2   600   7.148348E-05   3.900000E+02   -1.253655E-03   2.600000E+01   -2.4283E-02   6.3843E-05
  3   800   3.065192E-05   5.590000E+02   -6.783726E-04   3.900000E+01   -6.7585E-02   4.5637E-05
  4  1000   7.003061E-04   1.300000E+01   -4.332219E-04   1.170000E+02   -7.4145E-02   3.7146E-05
  5  1200   9.295235E-04   1.300000E+01   -2.703715E-04   1.690000E+02   -6.4851E-02   3.7847E-05
  6  1400   8.245250E-04   2.600000E+01   -1.556510E-04   2.730000E+02   -5.9453E-02   2.1048E-05
  7  1600   4.648917E-04   3.900000E+01   -2.572251E-04   1.300000E+01   -6.1298E-02   8.2141E-06
  8  1800   3.019019E-04   3.900000E+01   -5.029996E-04   1.300000E+01   -6.0933E-02   9.7501E-06
  9  2000   3.483344E-04   5.200000E+01   -5.132793E-04   2.600000E+01   -5.0253E-02   1.2647E-05
 10  2200   3.462673E-04   6.500000E+01   -3.538358E-04   2.600000E+01   -3.0778E-02   1.6528E-05
 11  2400   3.729273E-04   7.800000E+01   -3.704854E-04   3.900000E+01   -9.9279E-03   2.1272E-05
 12  2600   3.730184E-04   9.100000E+01   -3.337407E-04   5.200000E+01    7.2564E-03   2.4009E-05
 13  2800   3.083018E-04   1.170000E+02   -4.628690E-04   5.200000E+01    2.0913E-02   2.3735E-05
 14  3000   2.254179E-04   1.560000E+02   -4.262788E-04   6.500000E+01    3.2358E-02   2.0166E-05
 15  3200   3.379405E-04   3.900000E+01   -3.220338E-04   7.800000E+01    3.9905E-02   1.5998E-05
 16  3400   3.691465E-04   5.200000E+01   -2.114250E-04   9.100000E+01    3.9830E-02   1.2507E-05
 17  3600   3.017469E-04   5.200000E+01   -1.102054E-04   2.600000E+01    3.0312E-02   1.0544E-05
 18  3800   2.510361E-04   6.500000E+01   -2.367911E-04   3.900000E+01    1.3201E-02   1.1958E-05
 19  4000   1.654315E-04   1.820000E+02   -2.479778E-04   3.120000E+02   -7.9862E-03   1.5771E-05
 20  4200   2.241548E-04   2.080000E+02   -2.749074E-04   3.640000E+02   -3.0621E-02   1.9997E-05
 21  4400   2.496994E-04   2.340000E+02   -2.788258E-04   4.030000E+02   -5.3020E-02   2.3711E-05
 22  4600   2.416998E-04   2.730000E+02   -2.591677E-04   4.680000E+02   -7.2785E-02   2.5815E-05
 23  4800   2.015540E-04   3.120000E+02   -2.308119E-04   5.330000E+02   -8.6323E-02   2.5254E-05
 24  5000   2.238722E-04   1.170000E+02   -2.017903E-04   6.240000E+02   -9.0392E-02   2.2440E-05
 25  5200   1.855104E-04   1.430000E+02   -1.802844E-04   7.150000E+02   -8.3824E-02   1.8737E-05
 26  5400   1.312089E-04   1.560000E+02   -1.881246E-04   9.100000E+01   -6.7642E-02   1.5454E-05
 27  5600   1.223708E-04   7.800000E+01   -1.543889E-04   1.170000E+02   -4.4155E-02   1.3951E-05
 28  5800   1.658095E-04   7.800000E+01   -1.950515E-04   4.550000E+02   -1.6577E-02   1.4882E-05
 29  6000   1.763003E-04   3.380000E+02   -2.266281E-04   5.070000E+02    1.0607E-02   1.7606E-05
 30  6200   2.038634E-04   3.770000E+02   -2.405319E-04   5.590000E+02    3.1873E-02   2.0386E-05
 31  6400   2.031655E-04   4.160000E+02   -2.357105E-04   6.110000E+02    4.2203E-02   2.1289E-05
 32  6600   1.823956E-04   4.550000E+02   -2.208205E-04   6.760000E+02    3.8942E-02   1.9589E-05
 33  6800   1.468052E-04   5.070000E+02   -2.001637E-04   7.410000E+02    2.2501E-02   1.5984E-05
 34  7000   1.223187E-04   2.730000E+02   -1.771698E-04   8.190000E+02   -4.2111E-03   1.2209E-05
 35  7200   1.197935E-04   1.430000E+02   -1.563780E-04   8.970000E+02   -3.6582E-02   1.0018E-05
 36  7400   1.034319E-04   1.560000E+02   -1.398645E-04   9.880000E+02   -6.8722E-02   9.8692E-06
 37  7600   1.022806E-04   5.200000E+01   -1.294481E-04   5.980000E+02   -9.4097E-02   1.0820E-05
```

38	7800	1.207520E-04	4.810000E+02	-1.629922E-04	6.500000E+02	-1.0699E-01	1.1571E-05
39	8000	1.335896E-04	5.200000E+02	-1.833316E-04	7.020000E+02	-1.0430E-01	1.1852E-05
40	8200	1.285603E-04	5.590000E+02	-1.920350E-04	7.670000E+02	-8.5569E-02	1.2496E-05
41	8400	1.062198E-04	3.380000E+02	-1.881691E-04	8.320000E+02	-5.7540E-02	1.4110E-05
42	8600	1.120375E-04	1.950000E+02	-1.748622E-04	8.970000E+02	-2.3049E-02	1.6013E-05
43	8800	1.245160E-04	2.210000E+02	-1.540028E-04	9.620000E+02	1.0245E-02	1.6585E-05
44	9000	1.271073E-04	2.834000E+03	-1.344808E-04	1.690000E+02	3.5074E-02	1.4905E-05
45	9200	1.062125E-04	2.886000E+03	-1.391367E-04	1.950000E+02	4.9970E-02	1.2161E-05
46	9400	1.346137E-04	1.560000E+02	-1.261177E-04	2.080000E+02	5.0500E-02	1.1143E-05
47	9600	1.484268E-04	6.370000E+02	-1.243876E-04	8.190000E+02	3.9545E-02	1.3583E-05
48	9800	1.705648E-04	6.760000E+02	-1.345317E-04	8.840000E+02	2.1262E-02	1.8118E-05
49	10000	1.741251E-04	7.280000E+02	-1.408030E-04	9.360000E+02	3.8134E-04	2.1428E-05
50	10200	1.607075E-04	7.800000E+02	-1.473462E-04	1.001000E+03	-1.9224E-02	2.1286E-05
51	10400	1.472430E-04	5.070000E+02	-1.569389E-04	1.066000E+03	-3.5239E-02	1.8540E-05
52	10600	1.171744E-04	5.460000E+02	-1.699902E-04	1.144000E+03	-4.7033E-02	1.6040E-05
53	10800	1.061289E-04	3.510000E+02	-1.819821E-04	1.222000E+03	-5.5339E-02	1.5651E-05
54	11000	8.318333E-05	1.040000E+02	-1.887048E-04	1.287000E+03	-6.1433E-02	1.6679E-05
55	11200	7.469360E-05	7.150000E+02	-1.836738E-04	1.365000E+03	-6.5905E-02	1.7248E-05
56	11400	9.168772E-05	7.670000E+02	-1.919088E-04	9.520000E+02	-6.7722E-02	1.6592E-05
57	11600	1.095599E-04	8.190000E+02	-1.959334E-04	1.014000E+03	-6.4284E-02	1.5401E-05
58	11800	1.276661E-04	8.580000E+02	-1.798871E-04	1.066000E+03	-5.2655E-02	1.4343E-05
59	12000	1.428231E-04	9.100000E+02	-1.496653E-04	1.131000E+03	-3.1267E-02	1.3256E-05
60	12200	1.531123E-04	9.620000E+02	-1.143066E-04	1.196000E+03	-1.2684E-03	1.2101E-05
61	12400	1.544836E-04	1.027000E+03	-8.566981E-05	1.261000E+03	3.2959E-02	1.1796E-05
62	12600	1.466938E-04	1.092000E+03	-1.015219E-04	2.054000E+03	6.4332E-02	1.3008E-05
63	12800	1.341266E-04	8.060000E+02	-1.023756E-04	2.132000E+03	8.4662E-02	1.4603E-05
64	13000	1.525933E-04	2.210000E+02	-1.081491E-04	1.508000E+03	8.7133E-02	1.4640E-05
65	13200	1.596377E-04	2.340000E+02	-1.314739E-04	1.586000E+03	6.8901E-02	1.3157E-05
66	13400	1.417320E-04	2.470000E+02	-1.396410E-04	1.664000E+03	3.2665E-02	1.2680E-05
67	13600	1.147236E-04	2.158000E+03	-1.656228E-04	1.235000E+03	-1.3673E-02	1.4903E-05
68	13800	1.288251E-04	1.820000E+02	-1.911741E-04	1.287000E+03	-5.9247E-02	1.7788E-05
69	14000	1.390075E-04	1.950000E+02	-1.918611E-04	1.352000E+03	-9.3481E-02	1.7725E-05
70	14200	1.314195E-04	2.080000E+02	-1.712409E-04	1.404000E+03	-1.0923E-01	1.4511E-05
71	14400	1.105107E-04	2.210000E+02	-1.618752E-04	1.820000E+02	-1.0483E-01	1.1949E-05
72	14600	1.016287E-04	1.690000E+02	-1.648267E-04	1.950000E+02	-8.4332E-02	1.2584E-05
73	14800	1.188406E-04	1.690000E+02	-1.419816E-04	2.080000E+02	-5.5859E-02	1.4099E-05
74	15000	1.391169E-04	1.820000E+02	-1.244106E-04	1.560000E+02	-2.8469E-02	1.3047E-05
75	15200	1.359320E-04	1.950000E+02	-1.169085E-04	1.690000E+02	-8.8705E-03	1.0607E-05
76	15400	1.363331E-04	1.170000E+03	-1.192181E-04	1.690000E+02	6.0583E-04	1.1362E-05
77	15600	1.361984E-04	1.560000E+02	-1.287599E-04	1.820000E+02	1.9690E-03	1.6165E-05
78	15800	1.480838E-04	2.522000E+03	-1.415828E-04	3.055000E+03	-3.3120E-04	1.9817E-05

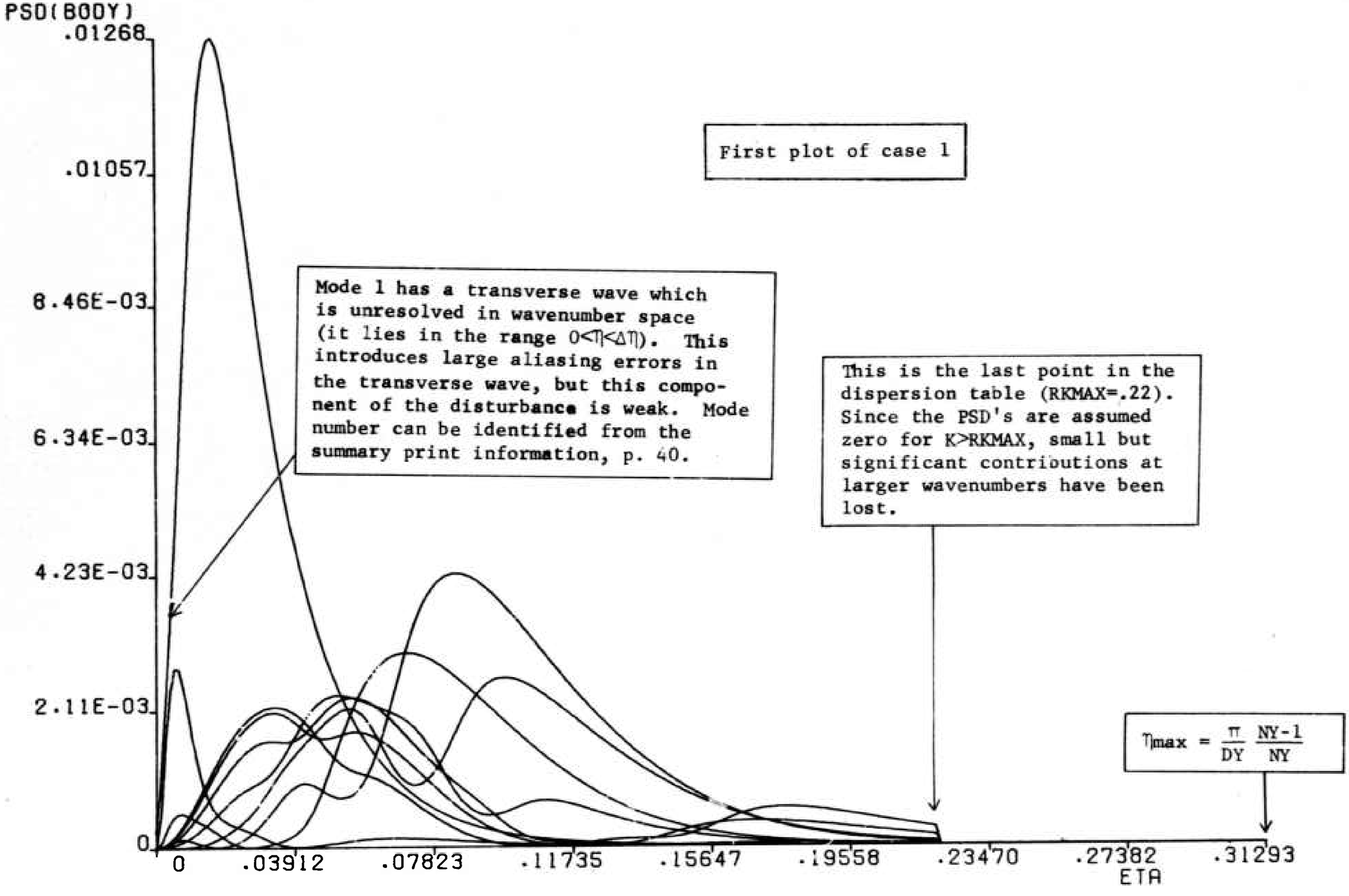
```
79    16000   1.234152E-04   2.574000E+03   -1.506938E-04   2.730000E+02   -2.0859E-03   1.8042E-05
80    16200   1.362294E-04   1.976000E+03   -1.542820E-04   2.860000E+02   -1.5726E-03   1.3810E-05
81    16400   1.445338E-04   2.054000E+03   -1.527425E-04   2.210000E+02   -3.7900E-04   1.3148E-05
82    16600   1.215363E-04   2.119000E+03   -1.577647E-04   2.340000E+02   -2.4445E-03   1.6598E-05
83    16800   1.483231E-04   3.055000E+03   -1.578031E-04   2.340000E+02   -1.1717E-02   1.9302E-05
84    17000   1.475415E-04   2.210000E+02   -1.494751E-04   1.482000E+03   -2.9564E-02   1.9114E-05
85    17200   1.728720E-04   2.210000E+02   -1.836505E-04   1.534000E+03   -5.3333E-02   1.9585E-05
86    17400   2.075016E-04   2.340000E+02   -2.073205E-04   1.599000E+03   -7.6931E-02   2.3079E-05
87    17600   2.123348E-04   2.470000E+02   -2.095703E-04   1.664000E+03   -9.3203E-02   2.5597E-05
88    17800   2.002735E-04   1.950000E+02   -1.882470E-04   1.729000E+03   -9.5945E-02   2.2438E-05
89    18000   1.914388E-04   2.080000E+02   -1.552736E-04   1.794000E+03   -8.6989E-02   1.5861E-05
90    18200   1.726688E-04   2.080000E+02   -1.500497E-04   2.470000E+02   -6.6572E-02   1.2294E-05
91    18400   1.647386E-04   2.210000E+02   -1.863234E-04   1.950000E+02   -4.1739E-02   1.3677E-05
92    18600   1.513812E-04   1.378000E+03   -1.891594E-04   2.080000E+02   -1.8701E-02   1.6279E-05
93    18800   1.817755E-04   1.417000E+03   -2.166876E-04   2.080000E+02   -1.4007E-03   1.7167E-05
94    19000   1.736609E-04   1.456000E+03   -2.181775E-04   2.210000E+02    9.4673E-03   1.7004E-05
95    19200   1.566027E-04   1.183000E+03   -1.752621E-04   2.340000E+02    1.5513E-02   1.6408E-05
96    19400   1.570568E-04   9.490000E+02   -1.738043E-04   1.820000E+02    1.8437E-02   1.4533E-05
97    19600   1.620176E-04   7.540000E+02   -1.492153E-04   1.950000E+02    1.8018E-02   1.2271E-05
98    19800   1.688590E-04   7.800000E+02   -1.027359E-04   1.950000E+02    1.1741E-02   1.2076E-05
99    20000   1.842200E-04   1.820000E+02   -1.393340E-04   2.600000E+03   -3.5280E-03   1.3510E-05
```

```
CP TIMES
CASE          17.895
INCON           .020
DTCON          0.000
DTEVAL         0.000
DTEVEC         0.000
DTDER2         0.000
DTWAKE         0.000
FTCON         12.564
FTDTAB          .161
FTGEND          .644
FTNEWX         3.434
FTFFT          6.519
FTPOT          0.000
PPCON          5.292
SPCON          0.000
SPDTAB         0.000
SPWFAM         0.000
SP1PNT         0.000
END
```

Note time saved by using the dispersion tables from the first case

end of second case

last card of input data = end of run



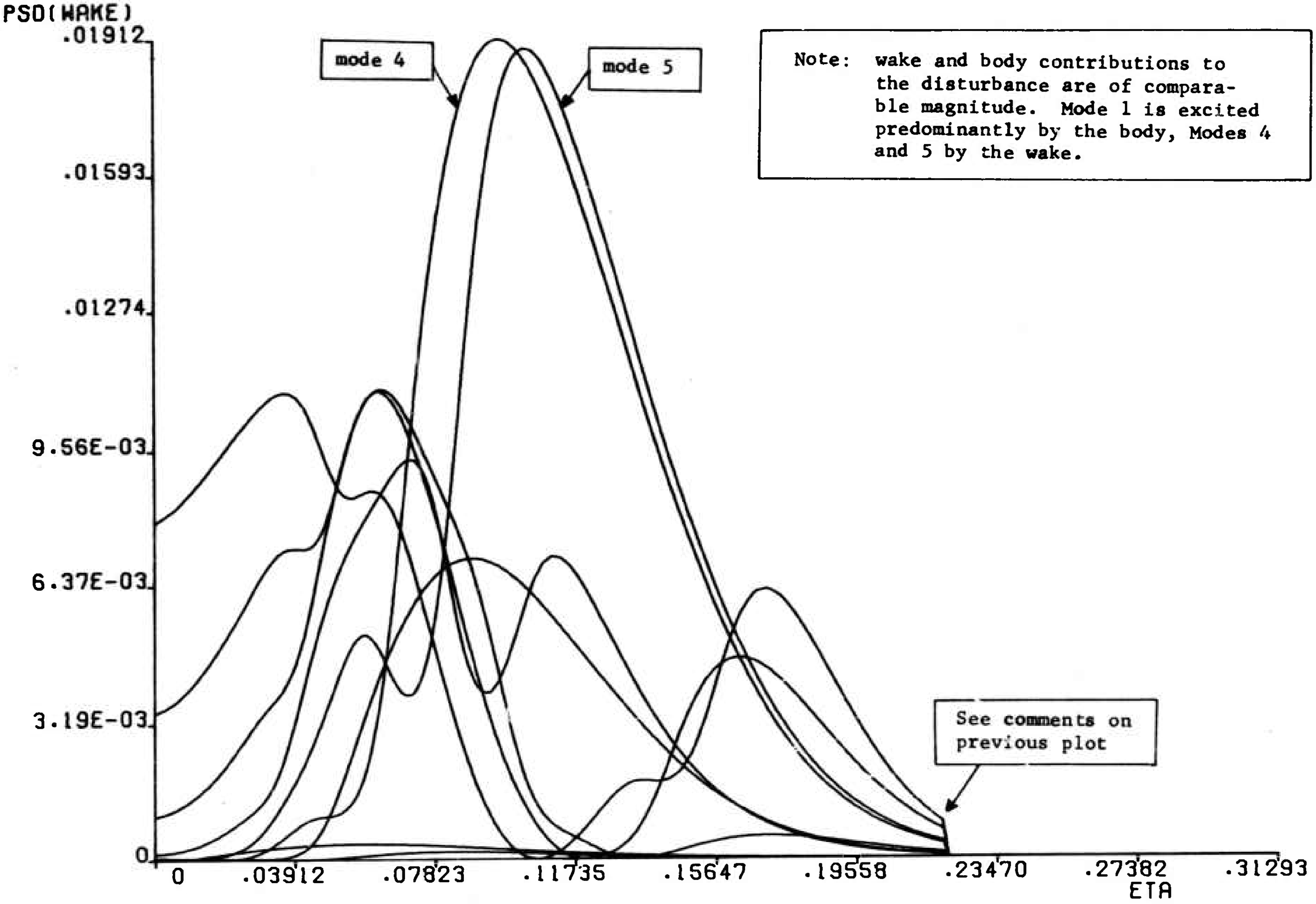
First plot of case 1

Mode 1 has a transverse wave which is unresolved in wavenumber space (it lies in the range $0<\eta<\Delta\eta$). This introduces large aliasing errors in the transverse wave, but this component of the disturbance is weak. Mode number can be identified from the summary print information, p. 40.

This is the last point in the dispersion table (RKMAX=.22). Since the PSD's are assumed zero for K>RKMAX, small but significant contriputions at larger wavenumbers have been lost.

$\eta_{max} = \frac{\pi}{DY} \frac{NY-1}{NY}$

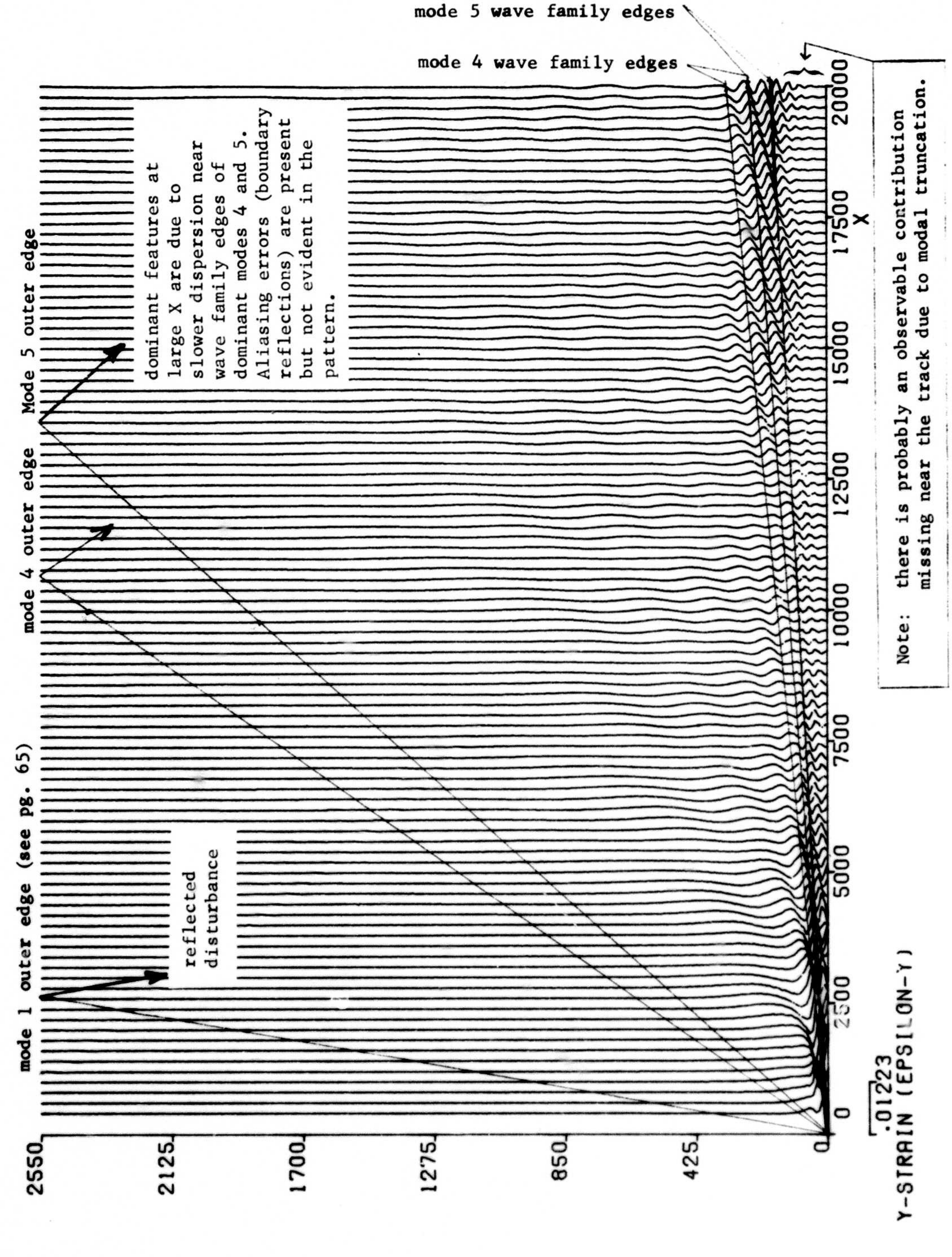
Y-STRAIN (EPSILON-Y)



Note: wake and body contributions to the disturbance are of comparable magnitude. Mode 1 is excited predominantly by the body, Modes 4 and 5 by the wake.

See comments on previous plot

Y-STRAIN (EPSILON-Y)



mode 1 outer edge (see pg. 65)

reflected disturbance

mode 4 outer edge

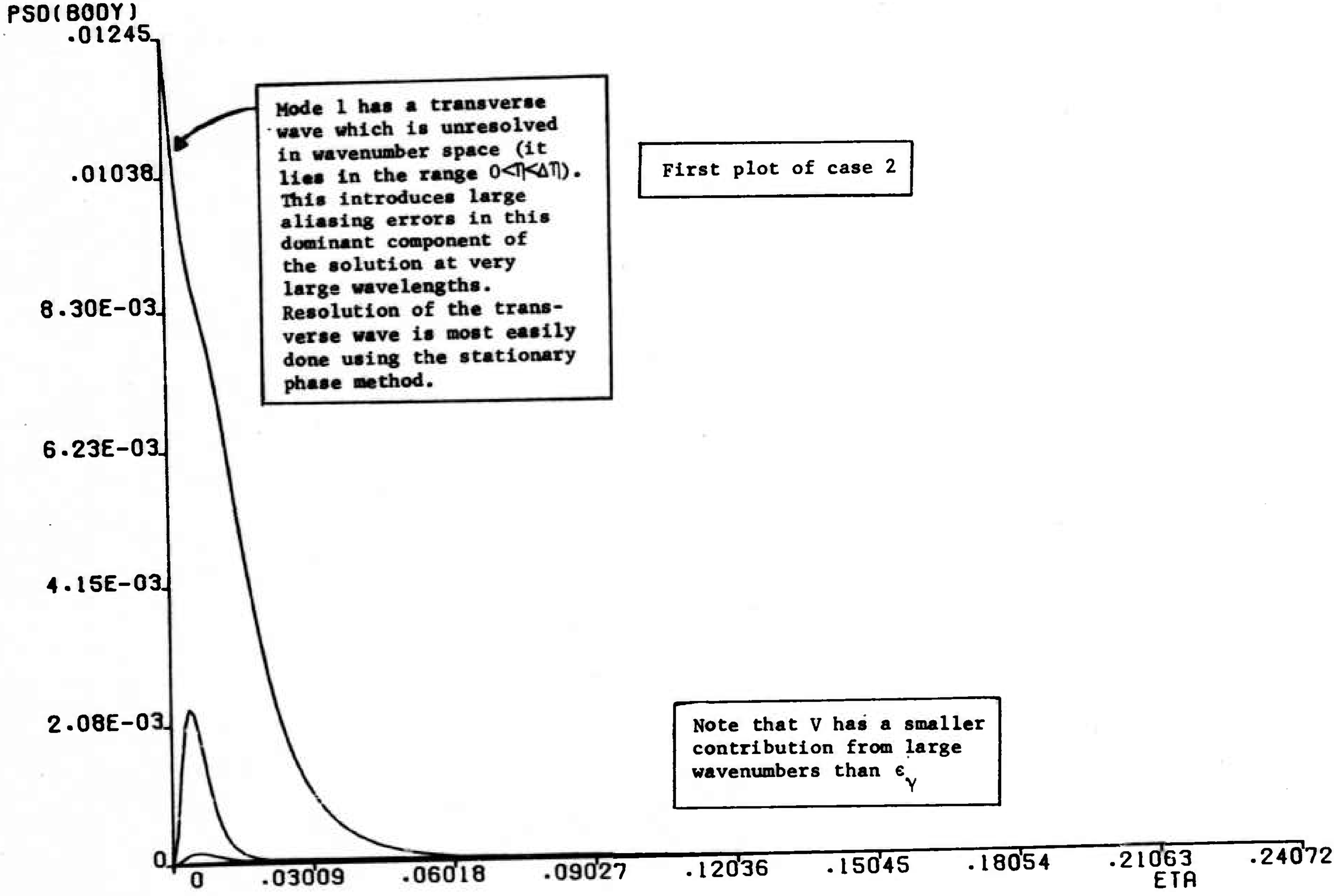
Mode 5 outer edge

dominant features at large X are due to slower dispersion near wave family edges of dominant modes 4 and 5. Aliasing errors (boundary reflections) are present but not evident in the pattern.

mode 5 wave family edges

mode 4 wave family edges

Note: there is probably an observable contribution missing near the track due to modal truncation.

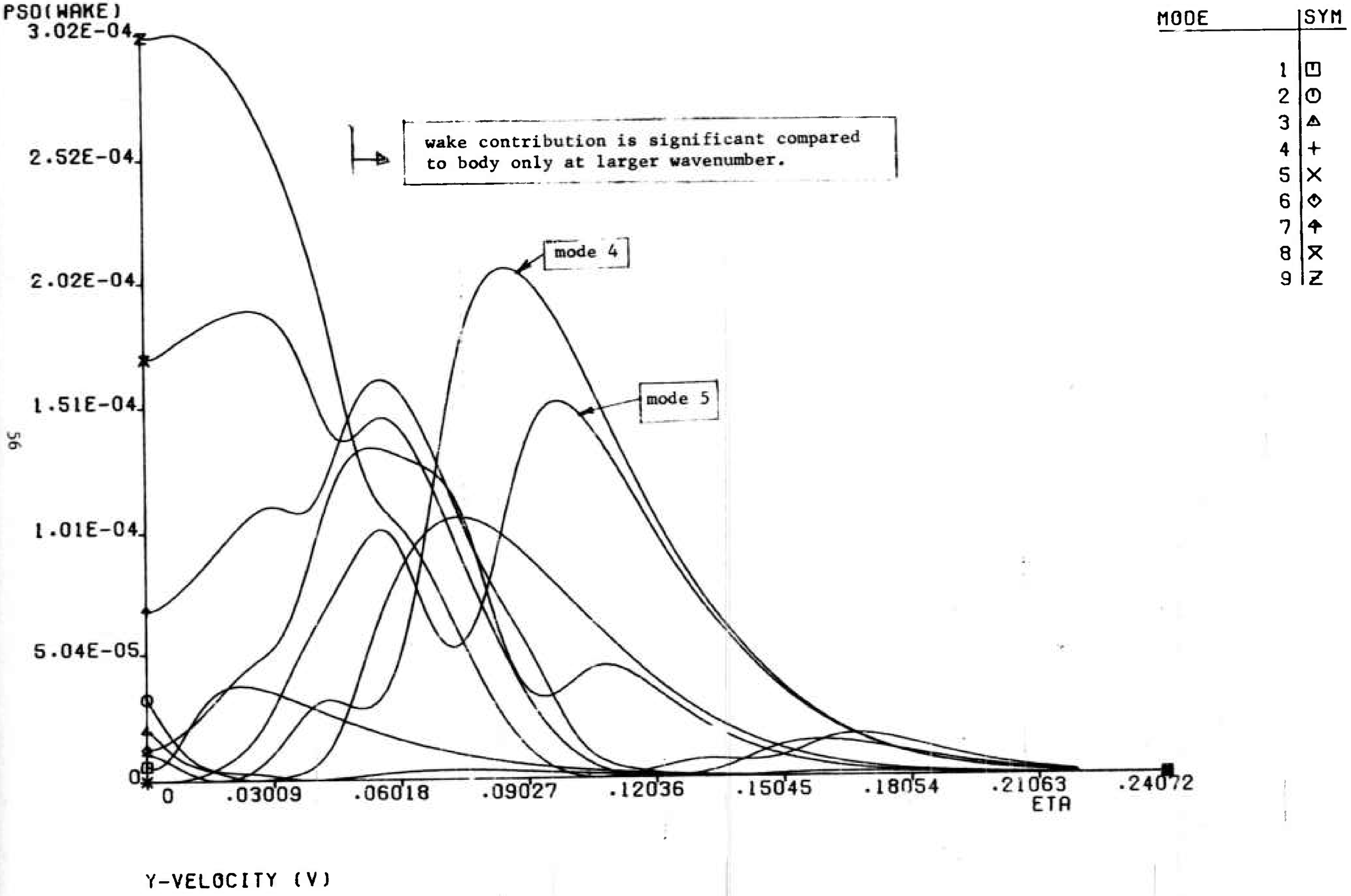


Mode 1 has a transverse wave which is unresolved in wavenumber space (it lies in the range $0<\eta<\Delta\eta$). This introduces large aliasing errors in this dominant component of the solution at very large wavelengths. Resolution of the transverse wave is most easily done using the stationary phase method.

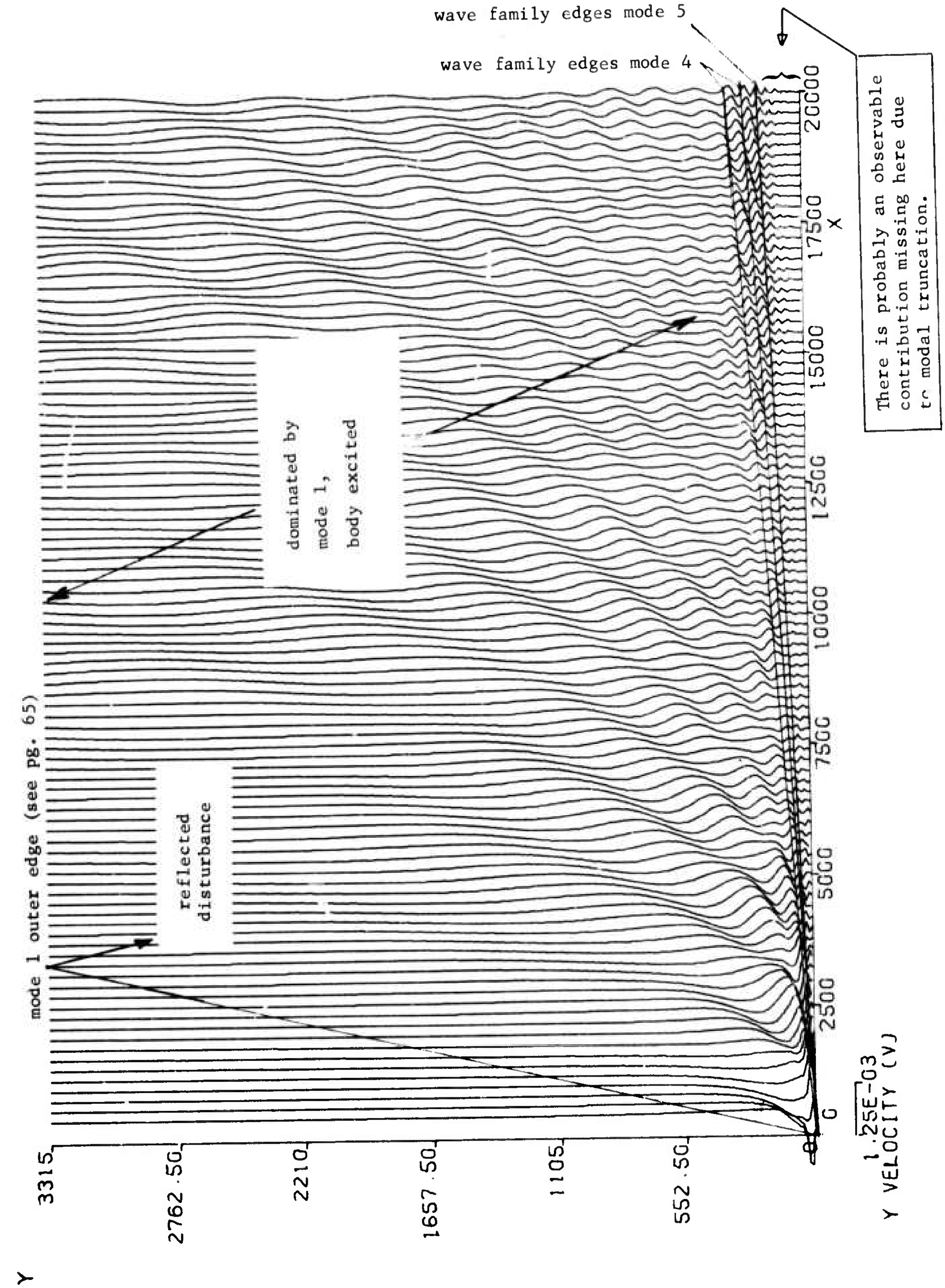
First plot of case 2

Note that V has a smaller contribution from large wavenumbers than ϵ_Y

Y-VELOCITY (V)



wake contribution is significant compared to body only at larger wavenumber.


wave family edges mode 5
wave family edges mode 4
There is probably an observable contribution missing here due to modal truncation.
mode 1 outer edge (see pg. 65)
reflected disturbance
dominated by mode 1, body excited
0
2500
5000
7500
10000
12500
15000
17500
20000
X
3315
2762.50
2210
1657.50
1105
552.50
0
Y
1.25E-03
Y VELOCITY (V)

5.2 Run 2

The second run is a simple illustration of reformatting a display. TAPE9 and TAPE10, which were saved at the end of run 1 are now restored. These files contain all the print/plot data for the second case of run 1. It is desired to re-plot the y-velocity display, cutting it off at X = 12500, but otherwise keeping the same scaling.

The data deck is listed below.

```
INP,O
P$OCEAN  NODISP=1$
INP,G
P$GRID  NOGRID=1$
INP,P
P$PP  IDPP=4HCUTS, IEDIT=1, PMIN=0, PMAX=12500, PLEN=5,
  VLEN=6, FTODP=0., FLEN=.4, IPRINT=0$
RUN
END
```

The entries NØDISP = 1 and NØGRID = 1 cause the program to bypass the dispersion table and grid modules and directly enter the print/plot processor.

The entry IDPP = 4HCUTS indicates that this PP data set applies to the grid display (if there were more than one grid display, IØCUR could be used to specify which one). IEDIT = 1 causes the program to process only those displays called out in a PP data set. Since CUTS is the only display mentioned by input, this eliminates the BPSD and WPSD displays. This could also have been done by entering data sets for BPSD and WPSD with IPRINT = 0 and IPLØT = 0.

The variables PMIN and PMAX specify the range of the parameter X to be displayed. The lengths of the X and Y axes are set to 5 and 6 inches, respectively. Setting FTØDP = 0 causes the program to bypass the automatic function scaling and use FLEN as the length of the function axis (.4 was arrived at by measuring the length of the function axis in the display from run 1). IPRINT = 0 turns off all printing for this display.

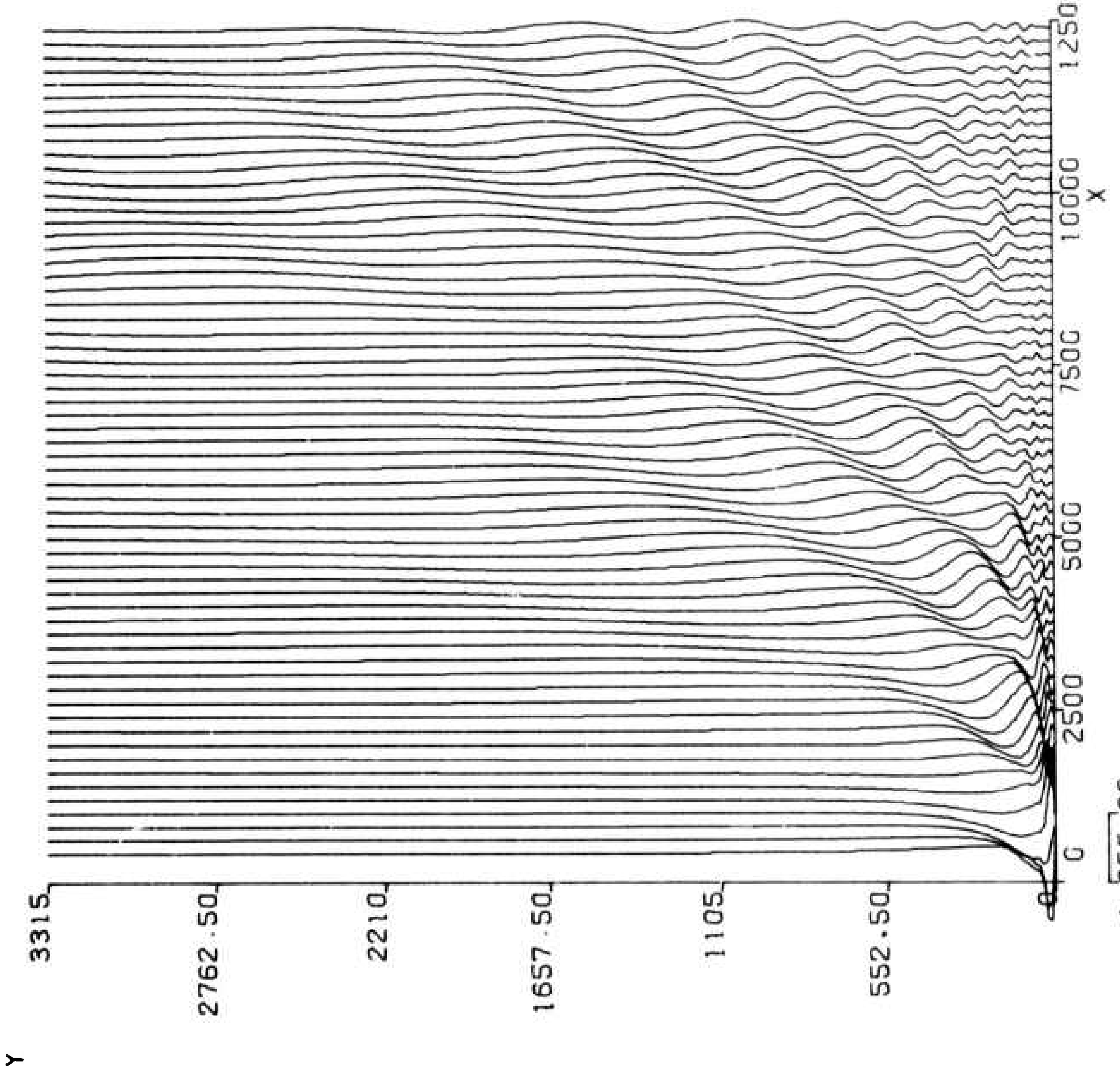
This is the output from run 2.

```
INP,O
$OCEAN  NODISP=1$
INP,G
$GRID  NCGRID=1$
INP,P
$PP  IDPP=4HCUTS, IEDIT=1, PMIN=0, PMAX=12500, PLEN=5,
 VLEN=6, FTODP=0., FLEN=.4, IPRINT=0$
RUN
```

input card images

```
CP TIMES
CASE          2.255
INCON          .024
DTCON         0.000
DTEVAL        0.000
DTEVEC        0.000
DTDER2        0.000
DTWAKE        0.000
FTCON         0.000
FTDTAB        0.000
FTGENO        0.000
FTNEWX        0.000
FTFFT         0.000
FTPOT         0.000
PPCON         2.228
SPCON         0.000
SPDTAB        0.000
SPWFAM        0.000
SP1PNT        0.000
END
```

last input card = end of run


Y
3315
2762.50
2210
1657.50
1105
552.50
0
0
2500
5000
7500
10000
12500
X
1.25E-03
Y-VELOCITY (V)

5.3 Run 3

Examination of the grid plot for y-velocity in run 1 indicates that aliasing becomes obvious at about 10000 to 15000 meters downtrack. In run 2, the grid was cut off at X = 12400 meters. It is now desired to extend the grid from 12600 (cross cuts are computed at 200 meter intervals) back to 14000 meters using the stationary phase method, and plot it with the same scaling as before.

The data deck is listed below.

```
INP,O
P$OCEAN  LIBSEA=1, IPREDG=1$
LIB,S,B1
LIB,S,W1
INP,S
P$SOURCE  BODDEP=45, BODSPD=2$
INP,G
P$GRID  OBSDEP=0, NOBS=1, DX=200, XMIN=12600, NX=8, DY=13, NY=255,
  YMIN=13, MODE1=1, MODEN=9, IVAR=2, ISPHAS=1$
INP,P
P$PP  IDPP=4HCUTS, PMIN=10000, PMAX=15000, PLEN=2, VMIN=0, VLEN=5
  FMAX=1.253655E-3, FMIN=-1.253655E-3, FTDDP=0, FLEN=-.4$
RUN
END
```

The processed ocean data on TAPE2 which was saved at the end of run 1 is restored for this run. The same input data library (TAPE1) which was used in run 1 is also used here.

The entry LIBSEA=1 causes the processed ocean data to be read from TAPE2. IPREDG causes the wave family edge table to be printed. The two LIB cards bring in the same body and wake models used before. The grid data are the same as the second case of run 1 except that stationary phase is used (ISPHAS=1) and the range of X is 12600 to 14000 (XMIN=12600, NX=8, DX=200). No wall reflections appear in the solutions since the lateral boundary is at y→∞.

The plot scaling is designed to allow this plot to be laid at the end of the run 2 plot to form a single picture. PMIN and PMAX are values of X at tic marks on the old plot and which bracket the actual range of X. Since the tic marks are always spaced one inch apart, the appropriate length of the X axis is given by PLEN=2. VMIN=0 causes the origin of the Y axis to be zero instead of 13, which is where the stationary phase grid starts. To force the same scaling as before, FMAX and FMIN are set to ± maximum value found in the previous run. Note that the negative value input to FLEN causes the function axis to be reversed; thus negative function values are plotted to the right, positive to the left. This was done so that the pattern would appear to be a continuation of the run 2 plot (remember that for an FFT grid $Y \geq 0$, for a stationary phase grid $Y < 0$, and $v(-Y) = -v(Y)$).

This is the output from run 3.

```
INP,O
$OCEAN  LIBSEA=1, IPREDG=1$
LIB,S,B1
$SOURCE  BODDIA=10, BODLEN=100, IBODY=1$
LIB,S,W1
$SOURCE  IWAKE=1, CWAKR=.9, CWAKX=.2, RESLVS=5, CWAKM=.2$
INP,S
$SOURCE  BODDEP=45, BODSPD=2$
INP,G
$GRID  OBSDEP=0, NOBS=1, DX=200, XMIN=12600, NX=8, DY=13, NY=255,
 YMIN=13, MODE1=1, MODEN=9, IVAR=2, ISPHAS=1$
INP,P
$PP  ICPP=4HCUTS, FMIN=10000, FMAX=15000, PLEN=2, VMIN=0, VLEN=6
 FMAX=1.253655E-3, FMIN=-1.253655E-3, FTODP=0, FLEN=-.4$
RUN
```

input card
images

```
WAVE FAMILY EDGE TABLE

*****MODE   1
        -Y/X             LAMBDA
 1  0.                  2.500000E-01
 2  9.635177E-01        2.623598E-01

*****MODE   2
        -Y/X             LAMBDA
 1  5.897563E-01        9.687781E-01

*****MODE   3
        -Y/X             LAMBDA
 1  3.439305E-01        2.363482E+00
 2  2.017545E-02        2.476951E+01
 3  2.259361E-02        3.665403E+01

*****MODE   4
        -Y/X             LAMBDA
 1  2.394440E-01        4.610456E+00
 2  1.294382E-02        4.813916E+01
 3  1.619656E-02        7.231856E+01

*****MODE   5
        -Y/X             LAMBDA
 1  1.851882E-01        7.539763E+00
 2  9.391829E-03        8.044321E+01
 3  1.267460E-02        1.188345E+02

*****MODE   6
        -Y/X             LAMBDA
 1  1.509090E-01        1.122766E+01
 2  7.335937E-03        1.215445E+02
 3  1.043991E-02        1.761452E+02

*****MODE   7
        -Y/X             LAMBDA
 1  1.275534E-01        1.561582E+01
 2  6.024340E-03        1.712508E+02
 3  8.903058E-03        2.439366E+02

*****MODE   8
```

transverse wave bounded by $0 \le -\frac{Y}{X} \le .9635177$

$\lambda = C^{-2}$ (C=phase speed) $= U^{-2}$ at axis

diverging wave bounded by $.9635177 \ge -\frac{Y}{X} \ge 0$

(the last wave family in a mode extends from the specified $-\frac{Y}{X}$ to the axis)

mode 2 is superfroude (no transverse wave) with only one wave family in the range $.5897563 \ge -\frac{Y}{X} \ge 0$

three wave families bounded by $.34 \ge -\frac{Y}{X} \ge .020$

$.020 \le -Y/X \le .022$

$.022 \ge -Y/X \ge 0$

```
    -Y/X            LAMBDA
 1  1.104642E-01   2.073788E+01
 2  5.143669E-03   2.293228E+02
 3  7.789543E-03   3.215933E+02

*****MCDE   9
    -Y/X            LAMBDA
 1  9.742230E-02   2.659046E+01
 2  4.469846E-03   2.953763E+02
 3  6.942648E-03   4.094663E+02
```

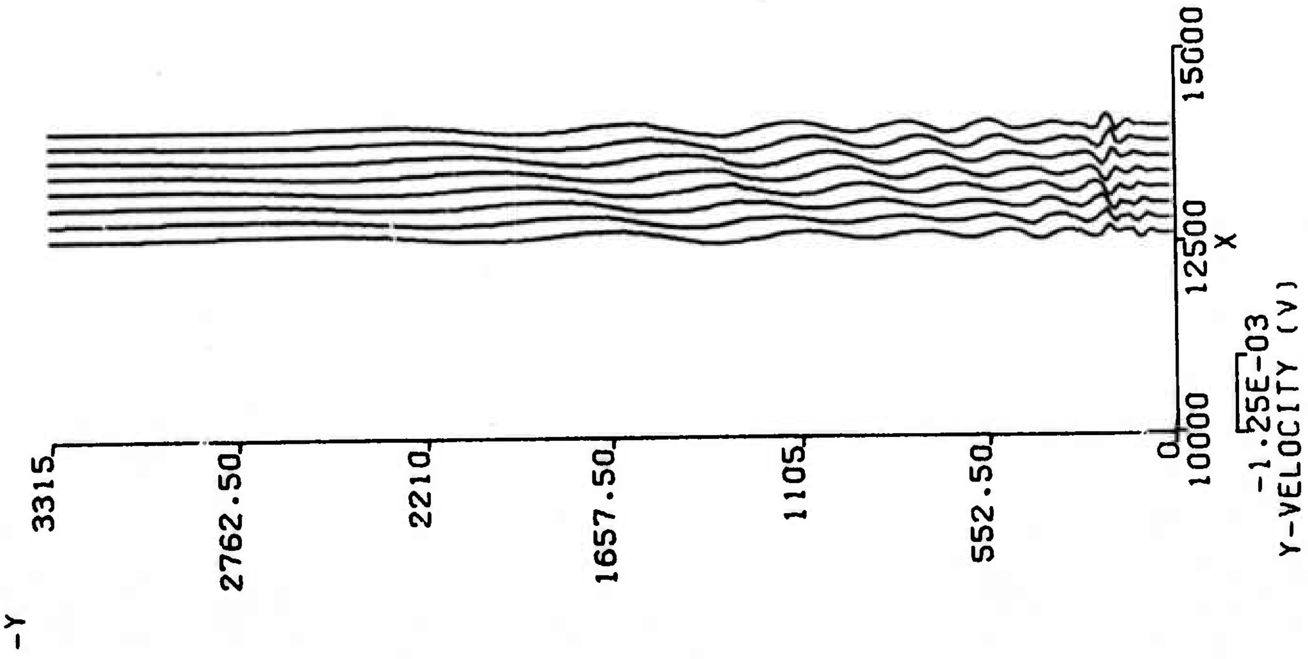
```
FUNCTION=Y-VELOCITY (V)                                                                               CUTS

     VALUE OF     MAXIMUM OF      -Y              MINIMUM OF       -Y
     X             FUNCTION      OF MAX          FUNCTION         OF MIN                 INTEGRAL        SQUARE
                                                                                                        INTEGRAL
 1     12600     1.333061E-04   1.339000E+03   -1.061699E-04   1.820000E+02          1.2968E-02      6.4553E-06
 2     12800     1.255116E-04   1.404000E+03   -1.278309E-04   1.950000E+02          1.0000E-02      7.7036E-06
 3     13000     1.259981E-04   1.690000E+02   -1.224875E-04   2.080000E+02          8.6580E-03      8.0900E-06
 4     13200     1.536575E-04   1.820000E+02   -9.049036E-05   9.230000E+02          9.6609E-03      9.0081E-06
 5     13400     1.739499E-04   1.950000E+02   -1.042883E-04   9.750000E+02          1.1350E-02      9.9096E-06
 6     13600     1.708294E-04   2.080000E+02   -1.096172E-04   1.014000E+03          1.2500E-02      1.0418E-05
 7     13800     1.419522E-04   1.287000E+03   -1.695386E-04   1.820000E+02          1.5004E-02      1.0672E-05
 8     14000     1.419206E-04   1.339000E+03   -1.785126E-04   1.950000E+02          1.4643E-02      9.9074E-06
```

```
CP TIMES
CASE          49.883
INCON           .046
DTCON         25.071
DTEVAL         1.212
DTEVEC        20.696
DTDER2         0.000
DTWAKE         2.702
FTCON          0.000
FTDTAB         0.000
FTGENO         0.000
FTNEWX         0.000
FTFFT          0.000
FTPOT          0.000
PPCON           .368
SPCON         24.356
SPDTAB          .287
SPWFAM          .165
SP1PNT        22.754
END
```

DTCON 25.071 ← note reduction in CP time (as compared to the first case of run 1) due to use of processed ocean data

SP1PNT 22.754 ← note that stationary phase requires significantly more time to generate a 256 point cross cut than does FFT. The stationary phase method should be used judiciously.


-Y
3315
2762.50
2210
1657.50
1105
552.50
0
10000
12500
15000
X
-1.25E-03
Y-VELOCITY (V)

6. REFERENCES

1. Baum, E. and Henryson, D., "Submarine Effects Engineering Code: 1. Dispersion Relation Calculation", TRW Report No. 20086-6010-RU-00, February 1974.

2. Baum, E., "Submarine Effects Engineering Code: 2. Disturbance Calculation: Far-Field", TRW Report No. 20086-6011-RU-00, July 1974.

3. Baum, E., "Submarine Effects Engineering Code: 3. Disturbance Calculation: Near-Field", TRW Report No. 20086-6012-RU-00, October 1974.

7. PROGRAM LISTING

This section contains a complete listing of program SEEK. The subroutines are broadly divided into seven categories.

1. Control and general purpose. Routines are ordered alphabetically.
2. Dispersion table generation. Routines are ordered alphabetically. Routine names begin with DT.
3. Fourier transform solution. Routine names begin with FT. Ordering is alphabetical.
4. Input processor. The order is alphabetical, the prefix is IN.
5. Print/plot processor. The order is alphabetical, the prefix is PP.
6. Stationary phase. The order is alphabetical, the prefix is SP.
7. Math. General purpose math routines in no particular order.

The program requires a field length of about 215000 (octal) words on a CDC 6400.

```
      PROGRAM SEEK(TAPE1=1004, TAPE2, TAPE3, TAPE4
     1,    INPUT=1004,TAPE5=INPUT, OUTPUT,TAPE6=OUTPUT
     2,    TAPE7=104, TAPE8=1004
     3,    TAPE9, TAPE10=1004, TAPE50=104)
C     FOR FILE USAGE, SEE ROUTINE INRUN1
C
      COMMON /CONTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
     1,      NODISP, NOGRID
      EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
     1,      (MODBOD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,      X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODEL1, MODEN
     2,      IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,      ISPHAS
C
C
C     INITIALIZE CASE NUMBER
      ICASE = 0
   10 CALL TIMER(0)
C     READ INPUT FOR NEXT CASE
      CALL INCON
C     INITIALIZE THE CASE
      CALL CASE1
      ITHOBS = 1
C     GENERATE NEW DISPERSION TABLES IF NECESSARY
   20 IF (JDISP .NE. 0)  CALL DTCON
C     SKIP IF NO GRID
      IF (NOGRID .NE. 0)  GO TO 30
C     STATIONARY PHASE SOLUTION CONTROL
      IF (ISPHAS .NE. 0)  CALL SPCON
C     FOURIER TRANSFORM AND POTENTIAL SOLUTION CONTROL
      IF (JFFT+JPOT .NE. 0)  CALL FTCON
C     SKIP IF ALL OBSERVATION DEPTHS HAVE BEEN DONE
      IF (ITHOBS .GE. NOBS)  GO TO 30
C     NEXT OBS DEPTH
      ITHOBS = ITHOBS +1
      OBSDEP = TABOBS(ITHOBS)
C     RESET CHECKLIST FLAGS INDICATING ONLY OBS DEPTH HAS CHANGED
      MODSEA = 0
      MODBOD = 0
      MODWAK = 0
      MODSUP = 0
      MODOBS = 1
      GO TO 20
C
C     PRINT/PLOT CONTROL
   30 CALL PPCON
      GO TO 10
      END
      SUBROUTINE BODY1
C     COMPUTE BODY SOURCE PARAMETERS--STRENGTH AND 1/2 SEPARATION
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,      RBSEP2, RBSTR, RBLIM
C
      COMMON /CONST/ JDK, JDMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,      JDCKSV, JDMSP, JDEDGE
C
C
C     TEST BODY MODEL
      IF (IBODY .NE. 1  .AND.  IPBODY .NE. 1)  GO TO 50
```

```
C     RANKINE BODY
      BD2 = BODDIA/2.
      SQBD2 = BD2**2
      BL2 = BODLEN/2.
      SQBL2 = BL2**2
C     INITIAL GUESS FOR SQUARE OF (SOURCE SEPARATION)/2
      SQA = 0.
C     ITERATE FOR SQA
      DO 20  I=1,20
      OLDSQA = SQA
      SQA = SQBL2 - BD2 *(SQBL2*(SQA+SQBD2))**.25
   20 IF (ABS(SQA-OLDSQA)/SQBL2 .LE. 1.E-10)  GO TO 40
      WRITE(6,30) SQA,OLDSQA
   30 FORMAT(41H NO CONVERGENCE ON BODY SOURCE SEPARATION,2E21.13)
      CALL ERRXIT
C     1/2 SOURCE TO SINK SEPARATION
   40 RBSEP2 = SQRT(SQA)
C     SOURCE STRENGTH (VOLUME/TIME)
      RBSTR = PI*BODSPD*SQBD2*SQRT(SQA+SQBD2) /RBSEP2
C
   50 IF (IBODY .NE. 2  .AND.  IPBODY .NE. 2)  GO TO 60
C     DIPOLE BODY.  RBLIM=LIM(RBSEP2*RBSTR) AS RBSEP2 GOES TO 0
      RBLIM = PI*BODSPD*BODDIA**3 /8.
   60 RETURN
      END
      SUBROUTINE CASE1
C     ONCE PER CASE INITIALIZATION
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,      RBSEP2, RBSTR, RBLIM
C
      COMMON /CONTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
     1,      NODISP, NOGRID
      EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
     1,      (MODBOD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,      X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,      IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,      ISPHAS
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,      SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMMON /WAKE/ IWAKE, CWAKR, CWAKX, XWAKE, WAKRAD, XWNOM
     1,      RESLVS, CWAKM
C
C
C     SET JDISP-- 0=NO DISP REQUIRED, 1=RECOMPUTE DISP TABLE
      JDISP = 0
C     INPUT FLAG TO BYPASS DISP TAB OVERIDES ALL ELSE
      IF (NODISP .NE. 0)  GO TO 10
C     SKIP IF NO FFT OR STATIONARY PHASE
      IF (IBODY+IWAKE+ISUPR .EQ. 0)  GO TO 10
C     RECOMPUTE IF OCEAN WAS CHANGED
      IF (MODSEA .NE. 0)  JDISP = 1
C     ...IF OBSERVATION DEPTH WAS CHANGED
      IF (NOBS .GT. 1)  MODOBS = 1
      IF (MODOBS .NE. 0)  JDISP = 1
C     ...IF BODY PARAMETERS WERE CHANGED AND BODY IS ON
      IF (MODBOD .NE. 0  .AND.  IBODY .NE. 0)  JDISP = 1
C     ...IF WAKE PARAMETERS WERE CHANGED AND WAKE IS ON
```

```
      IF (MODWAK .NE. 0  .AND.  IWAKE .NE. 0)  JDISP = 1
C     ...IF SUPERSTRUCTURE PARAMETERS WERE CHANGED AND SUPR IS ON
      IF (MODSUP .NE. 0  .AND.  ISUPR .NE. 0)  JDISP = 1
C
   10 JFFT = 0
      JPOT = 0
C     INPUT FLAG TO BYPASS GRID COMPUTATION OVERIDES ALL ELSE
      IF (NOGRID .NE. 0)  GO TO 220
C     JUMP IF USING STATIONARY PHASE
      IF (ISPHAS .NE. 0)  GO TO 220
C     USING FFT (AND/OR POTENTIAL)
C     SET JFFT-- 0=NO FFT, 1=USING FFT
      IF (IBODY+ISUPR+IWAKE .NE. 0)  JFFT = 1
C     SET JPOT-- 0=NO POTENTIAL, 1=POTENTIAL
      IF (IPBODY+IPSUPR .NE. 0)  JPOT = 1
  200 IF (NX .LE. 1  .OR.  NOBS .LE. 1)  GO TO 220
      WRITE(6,210) NX,NOBS
  210 FORMAT(39H ERROR--BOTH NX AND NOBS GREATER THAN 1,2I10)
      CALL ERRXIT
C
C     SKIP IF X-Y SCAN AT CONSTANT Z
  220 IF (NOBS .LE. 1)  GO TO 260
C     PRESET INTERNAL INCREMENT IN OBS DEPTHS
      DEL = DOBS
C     IF IT WAS INPUT, USE IT TO CONSTRUCT LIST OF DEPTHS
      IF (DEL .NE. 0.)  GO TO 230
C     IF MAX DEPTH IS ALSO ZERO, LIST WAS INPUT DIRECTLY
      IF (OBSMAX .EQ. 0.)  GO TO 250
C     COMPUTE INCREMENT FROM INPUT MAX, MIN AND NUMBER OF POINTS
      DEL = (OBSMAX-TABOBS(1)) / FLOAT(NOBS-1)
C     CONSTRUCT EQUAL INCREMENT TABLE
  230 DO 240  I=2,NOBS
  240 TABOBS(I) = TABOBS(I-1) + DEL
  250 OBSDEP = TABOBS(1)
  260 CONTINUE
      RETURN
      END
      SUBROUTINE ENDRUN
C     END OF RUN PROCEDURE
C
      COMMON /PRTPLT/ IPLTON, XAORG, YAORG, XPORG, YPORG
C
C
C     SKIP IF NO PLOTING HAS BEEN DONE
      IF (IPLTON .EQ. 0)  GO TO 10
C     WRAP UP PLOTS
      CALL PLOT(0., 0., 999)
   10 CONTINUE
      CALL EXIT
      END
      SUBROUTINE ERRXIT
C     ERROR EXIT PROCEDURE
      CALL ENDRUN
      END
      SUBROUTINE SETID(ID,IT,PN,VN,FN)
C     PRESET PP ID BLOCK
C
      COMMON /PPCOM/
     1       LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,      VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,      IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,      IDPP, NV, ISYM
```

```
     E,     ENDPP, IBLOKS(1)
      DIMENSION FN(2)
      DATA BIG/1.E30/
C
C
C     JUMP IF PREVIOUS OUTPUT SET IS FINISHED
      IF (IDPP .EQ. 1H )  GO TO 20
C     A PP SET IS STILL IN PROGRESS--DONT START ANOTHER
      WRITE(6,10) IDPP,ID
   10 FORMAT(35H FILE CONFLICT DUE TO PROGRAM ERROR,A10,3X,A10)
      CALL ERRXIT
C     PLOT ID
   20 IDPP = ID
C     PLOT TYPE-- 0=RASTER, 1=MULTI-TRACE
      IPLTYP = IT
C     PARAMETER, VARIABLE AND FUNCTION NAMES
      PNAME = PN
      VNAME = VN
      FNAME(1) = FN(1)
      FNAME(2) = FN(2)
C
C     PARAMETER, VARIABLE AND FUNCTION MAXIMA AND MINIMA
      PMAX = -BIG
      VMAX = -BIG
      FMAX = -BIG
      PMIN = BIG
      VMIN = BIG
      FMIN = BIG
C     PARAMETER, VARIABLE AND FUNCTION AXIS LENGTHS
      PLEN = 10.
      VLEN = 10.
      IF (IPLTYP .EQ. 0)  VLEN = 8.
      FLEN = 8.
C
      TITLE(1) = 1H
      TITLE(2) = 1H
C     OCCURANCE NUMBER IS IGNORED
      IOCUR = 0
C     IPLOT-- 0=OFF, 1=PLOT
      IPLOT = 1
C     IPRINT-- 0=OFF, 1=PRINT ALL, 2=SUMMARY, 3=1+2
      IPRINT = 2
C     EDIT FLAG IS IGNORED
      IEDIT = 0
C     SUPPRESS PP IS IGNORED
      NOPP = 0
C     ISYM-- 0=NO SYMBOLS, 1=LABEL TRACES ON A MULTI-TRACE PLOT
      ISYM = 0
C     FOR RASTER, COMPUTE FLEN FROM FLEN=FTODP*PLEN/(NP-1)
      FTODP = 1.
C     NUMBER OF PARAMETER VALUES
      NP = 0
C     (MAX) LENGTH OF VARIABLE LIST
      NV = 0
C     IVLIST-- 1=EQUAL INC, 2=FIXED, 3=VARIABLE
      IVLIST = 1
      RETURN
      END
      SUBROUTINE SUPR1
C     COMPUTE SUPERSTRUCTURE SOURCE PARAMETERS--STRENGTH AND 1/2 SEP
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
```

```
     1,     RBSEP2, RBSTR, RBLIM
C
      COMMON /CONST/ JDK, JDMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,     JDCKSV, JDMSP, JDEDGE
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,     SUPMID, SULIM, SUPDIA, SUPLEN
C
C
C     TEST SUPERSTRUCTURE MODEL
      IF (ISUPR .NE. 1  .AND.  IPSUPR .NE. 1)  GO TO 50
C     OVAL CROSS SECTION
      D2 = SUPDIA/2.
      RL2 = SUPLEN/2.
      SQL2 = RL2**2
      PI2 = PI/2.
C     INITIAL GUESS FOR SUSEP2 (SUSEP2 IS 1/2 SOURCE SEPARATION)
      SUSEP2 = RL2 - SUPDIA/(2.*PI)
C     ITERATE TO FIND SUSEP2
      DO 20  I=1,20
      OLDSEP = SUSEP2
      SUSEP2 = SQRT(SQL2 - SUSEP2*D2/(PI2-ATAN(D2/SUSEP2)))
   20 IF (ABS(SUSEP2-OLDSEP)/RL2 .LE. 1.E-10)  GO TO 40
      WRITE(6,30) SUSEP2,OLDSEP
   30 FORMAT(41H NO CONVERGENCE ON SUPR SOURCE SEPARATION,2E21.13)
      CALL ERRXIT
C     SOURCE STRENGTH (VOLUME/TIME/LENGTH)
   40 SUSTR = PI*BODSPD*(SQL2/SUSEP2-SUSEP2)
C
   50 IF (ISUPR .NE. 2  .AND.  IPSUPR .NE. 2)  GO TO 60
C     CIRCULAR SECTION.  SULIM=LIM(SUSTR*SUSEP2) AS SUSEP2 GOES TO 0
      SULIM = PI*BODSPD*(SUPDIA/2.)**2
   60 RETURN
      END
      SUBROUTINE TIMER(ID)
C     COLLECT AND PRINT TIMING INFORMATION FOR SELECTED SUBROUTINES
      DIMENSION TIMES(20), NAMES(20), TSTRT(20)
      DATA TCASE/-1./
      DATA TIMES/20*0./
      DATA NAMES/5HINCON, 5HDTCON, 6HDTEVAL, 6HDTEVEC, 6HDTDER2
     1,     6HDTWAKE, 5HFTCON, 6HFTDTAB, 6HFTGENO, 6HFTNEWX, 5HFTFFT
     2,     5HFTPOT, 5HPPCON, 5HSPCON, 6HSPDTAB, 6HSPWFAM, 6HSP1PNT
     3,     3*1H /
C     IABS(ID) IS INDEX NUMBER OF ROUTINE BEING TIMED
C     ID .GT. 0 = START OF ROUTINE, ID .LT. 0 = END OF ROUTINE
C     ID .EQ. 0 = START OF NEW CASE
C
C
      I = IABS(ID)
C     RETURN IF ILLEGAL INDEX VALUE
      IF (I .GT. 20)  RETURN
      IF (ID)  20,30,10
C
   10 CALL SECOND(TSTRT(I))
      RETURN
C
   20 CALL SECOND(TEND)
      TIMES(I) = TIMES(I) + TEND-TSTRT(I)
      RETURN
C
   30 CALL SECOND(T)
C     SKIP IF START OF 1ST CASE
```

```
      IF (TCASE .LT. 0.)  GO TO 80
      TCASE = T-TCASE
      WRITE(6,50) TCASE
   50 FORMAT(9H1CP TIMES/5H CASE,F15.3)
      DO 70  I=1,20
      IF (NAMES(I) .NE. 1H )  WRITE(6,60) NAMES(I),TIMES(I)
   60 FORMAT(1H ,A10,F9.3)
   70 TIMES(I) = 0.
   80 TCASE = T
      RETURN
      END
      SUBROUTINE WRTDAT(LOC1,LOC2,FLIST,INC,PVAL)
C     WRITE PP DATA RECORD ASSUMING IVLIST .NE. 3    FIND MIN AND MAX
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,      NTDTAB, NTEVEC, NTTEMP
C
      COMMON /PPCOM/
     1       LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,      VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,      ICCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,      IDPP, NV, ISYM
     E,      ENDPP, IBLOKS(1)
      DIMENSION FLIST(1)
C
C
C     BUMP NUMBER OF PARAMETER VALUES
      NP = NP+1
C     ADDRESS OF 1ST ENTRY IN FLIST
      I1 = (LOC1-1)*INC + 1
C     ADDRESS OF LAST ENTRY IN FLIST
      LAST = (LOC2-1)*INC + 1
C     WRITE PP DATA RECORD (ASSUME IVLIST .NE. 3)
      WRITE(NTPDAT) PVAL,LOC1,LOC2,(FLIST(I),I=I1,LAST,INC)
C     FIND MAX AND MIN
      DO 10  I=I1,LAST,INC
      IF (FMAX .LT. FLIST(I))  FMAX = FLIST(I)
   10 IF (FMIN .GT. FLIST(I))  FMIN = FLIST(I)
      IF (PMAX .LT. PVAL)  PMAX = PVAL
      IF (PMIN .GT. PVAL)  PMIN = PVAL
      RETURN
      END
      SUBROUTINE WRTD3(NWORDS,VLIST,FLIST,INC,PVAL)
C     WRITE PP DATA RECORD ASSUMING IVLIST=3   FIND MIN AND MAX
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,      NTDTAB, NTEVEC, NTTEMP
C
      COMMON /PPCOM/
     1       LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,      VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,      ICCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,      IDPP, NV, ISYM
     E,      ENDPP, IBLOKS(1)
      DIMENSION VLIST(1), FLIST(1)
C
C
C     BUMP NUMBER OF PARAMETER VALUES
      NP = NP+1
C     ADDRESS OF LAST ENTRY IN VLIST, FLIST
      LAST = (NWORDS-1)*INC + 1
C     WRITE PP DATA RECORD (ASSUME IVLIST=3)
```

```
      WRITE(NTPDAT) PVAL,NWORDS,(VLIST(I),FLIST(I),I=1,LAST,INC)
C     FIND MAX AND MIN
      DO 10  I=1,LAST,INC
      IF (FMAX .LT. FLIST(I))  FMAX = FLIST(I)
      IF (FMIN .GT. FLIST(I))  FMIN = FLIST(I)
      IF (VMAX .LT. VLIST(I))  VMAX = VLIST(I)
   10 IF (VMIN .GT. VLIST(I))  VMIN = VLIST(I)
      IF (PMAX .LT. PVAL)  PMAX  = PVAL
      IF (PMIN .GT. PVAL)  PMIN  = PVAL
      RETURN
      END
      SUBROUTINE WRTID(N,VLIST,INC)
C     WRITE OUT THE ID RECORD(S) FOR CURRENT PP SET
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,     NTDTAB, NTEVEC, NTTEMP
C
      COMMON /PPCOM/
     1      LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,     VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,     IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,     IDPP, NV, ISYM
     E,     ENDPP, IBLOKS(1)
      DIMENSION IDBLOK(1), DUMMY(1), VLIST(1)
      EQUIVALENCE (DUMMY,LENPP),(IDBLOK,DUMMY(2))
C
C
C     SET LENGTH OF ID BLOCK
      LENPP = LOCF(ENDPP) - LOCF(LENPP)
C     SET NUMBER OF ENTRIES IN VARIABLE LIST
      NV = N
C     SKIP IF VARIABLE LIST(S) PREVIOUSLY DEFINED
      IF (IVLIST .NE. 2)  GO TO 20
C     VLIST IS IN CALLING SEQ.   FIND MAX AND MIN
      LIM = (NV-1)*INC + 1
      DO 10  I=1,LIM,INC
      IF (VMAX .LT. VLIST(I))  VMAX = VLIST(I)
   10 IF (VMIN .GT. VLIST(I))  VMIN = VLIST(I)
C     WRITE ID RECORD
   20 WRITE(NTPID) LENPP,(IDBLOK(I),I=1,LENPP)
C
      IF (IVLIST .EQ. 2)  WRITE(NTPID) (VLIST(I),I=1,LIM,INC)
C     FLAG ROUTINE SETID THAT THIS PP SET HAS BEEN COMPLETED
      IDPP = 1H
      RETURN
      END
      SUBROUTINE DTCON
C     DISPERSION TABLE CONTROL
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,     RBSEP2, RBSTR, RBLIM
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,     NTDTAB, NTEVEC, NTTEMP
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,     X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,     IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,     ISPHAS
C
      COMMON /OCEAN/ LIBSEA, MODES, NK, TABK(100), NZT, DKRAT, DTDEP
     1,     TDEPMX, TDEP(400), NFLAG, SQBV(400), OCNDEP, RKMAX
```

```
     2,     SQN(400), NKT, IPPVEC
C
      COMMON /PPCOM/
     1      LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,     VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,     IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,     IDPP, NV, ISYM
     E,     ENDPP, IBLOKS(1)
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,     SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMMON /WAKE/ IWAKE, CWAKR, CWAKX, XWAKE, WAKRAD, XWNOM
     1,     RESLVS, CWAKM
C
      COMMON // RK, ITHK, ZT(400), LWAVEC, NZT1, NZT2, ZT1D
     1,     ZTN, ITOP, IBOT, SQK, TOP1, TOP2, TOP3, TOP4, TOP7, TOP8
     2,     BOT1, BOT2, BOT3, BOT4, BOT7, BOT8, IERR, OIC(400)
     3,     NTROT, Z, ZS, TEVAL(400). TDLDK(80), TPOBS(80)
     4,     TDPOBS(80), TDPSRC(80), TWI(80), TPSUPT(80), TPSUPB(80)
     5,     TDLDK2(80), EVEC(400,80), TEMP(400,5), DC(400)
     6,     TRIMAT(400,3), SU(400), SDL(400), SDL2(400), DL(400), D(400)
     7,     DU(400), CON(400), OMEG(400), BND(400), IEVAL(400)
     8,     EV3PST(80), PSIN2I(80)
C      *****SUMMARY OF APPROACH*****
C     EACH ENTRY TO DTCON GENERATES A COMPLETE DISPERSION TABLE (DT)
C     ON FILE NTDTAB.  FOR EACH VALUE OF WAVENUMBER K, THE FILE HAS
C     A RECORD CONTAINING K,TEVAL(M),TDLDK(M),TPOBS(M),TDPOBS(M),
C     TDPSRC(M),TWI(M),TSUPT(M),TSUPB(M),TDLDK2(M) WHERE INDEX M IS THE
C     MODE NUMBER.  THESE ARE EIGENVALUE LAMBDA=1/C**2, D(LAMBDA)/DK,
C     PSI(OBS DEPTH), D(PSI)/DZ(OBS DEPTH), D(PSI)/DZ(SOURCE DEP),
C     PARTIAL WAKE SOURCE TERM, PSI(TOP OF SUPER), PSI(BOTTOM OF SUP),
C     D2(LAMBDA)/DK2 WHERE PSI=EIGENVECTOR.
C     ON THE FIRST CASE, THE EIGENVALUES MAY BE READ FROM A FILE
C     GENERATED AND SAVED DURING A PREVIOUS RUN.  FOR SUBSEQUENT CASES,
C     THE INPUT PROCESSOR EXAMINES THE INPUT VARIABLES TO DETERMINE
C     WHAT HAS BEEN CHANGED AND SETS FLAGS (PREFIXED BY THE LETTERS
C     MOD) INDICATING WHICH OF THE DT VARIABLES WILL BE AFFECTED.  TO
C     MINIMIZE REDUNDANT COMPUTATIONS, BOTH THE DT AND A FILE WITH THE
C     COMPLETE SET OF EIGENVECTORS ARE AVAILABLE FROM THE PREVIOUS CASE.
C
C
      CALL TIMER(2)
C     DT INITIALIZATION
      CALL DTINIT
C     JPPVEC-- 0=OFF, 1=PRT/PLT EIGENVECTOR WITH MODE NUMBER OF JPPVEC
      JPPVEC = 0
      IF (ITHOBS .EQ. 1)  JPPVEC = IPPVEC
C     SKIP IF OPTION IS OFF
      IF (JPPVEC .EQ. 0)  GO TO 5
      CALL SETID(4HEVEC, 0, 1HK, 5HDEPTH, 20HEIGENVECTOR         )
      IVLIST = 2
    5 CONTINUE
C
C     LOOP FOR EACH VALUE OF WAVENUMBER K
      DO 10  ITHK=1,NKT
C     COMPUTE EIGENVALUES LAMBDA (ALL SPECIFIED MODES)
      CALL DTEVAL
C     JUMP IF ROUTINE COULD NOT DO IT
      IF (IERR .NE. 0)  GO TO 40
C     COMPUTE CORRESPONDING EIGENVECTORS PSI
      CALL DTEVEC
```

```
C     JUMP IF ROUTINE COULD NOT DO IT
      IF (IERR .NE. 0)  GO TO 40
C     D(LAMBDA)/DK
      CALL DTDER
C     D**2(LAMBDA)/DK**2
      CALL DTDER2
C     JUMP IF 2ND DERIV COULD NOT BE FOUND
      IF (IERR .NE. 0)  GO TO 40
C     PSI AND D(PSI)/DZ AT OBSERVATION DEPTH
      CALL DTOBS
C     D(PSI)/DZ AT SOURCE DEPTH
      IF (IBODY .NE. 0)  CALL DTBODY
C     (PARTIAL) WAKE SOURCE TERM (INCLUDES INTEGRAL)
      IF (IWAKE .NE. 0)  CALL DTWAKE
C     PSI AT TOP AND BOTTOM OF SUPERSTRUCTURE
      IF (ISUPR .NE. 0)  CALL DTSUPR
C     WRITE EIGENVECTOR IF CALLED FOR
      IF (JPPVEC .NE. 0) CALL WRTDAT(1,NZT,EVEC(NZT*JPPVEC-NZT+1),1,RK)
C     SAVE DISPERSION TABLE, GENERATE NEW DT LIBRARY IF OCEAN CHANGED
   10 CALL DTWRIT
C
C     WAIT FOR LAST SET OF EIGENVECTORS TO BE WRITTEN BEFORE CONTINUING
   20 IF (UNIT,NTEVEC)  20,30,30,30
   30 IF (JPPVEC .NE. 0)  CALL WRTID(NZT, ZT, 1)
      CALL TIMER(-2)
      RETURN
C
C     FATAL ERROR IF EIGENVALUE/VECTOR ROUTINES BOMBED EARLY IN DT
   40 IF (ITHK .LT. NKT/3)  CALL ERRXIT
C     (TRY TO) MAKE DO WITH AS MUCH OF DT AS THERE IS
      NKT = ITHK-1
      WRITE(6,50) NKT,TABK(NKT)
   50 FORMAT(31H K TABLE TRUNCATED AT ENTRY NO.,I4,3H K=,E14.7)
      GO TO 20
      END
      FUNCTION DTAVEN(DIA)
C     COMPUTE AVERAGE BV FREQ OVER DIAMETER=DIA CENTERED AT ZS
C
      COMMON /OCEAN/ LIBSEA, MODES, NK, TABK(100), NZT, DKRAT, DTDEP
     1,     TDEPMX, TDEP(400), NFLAG, SQBV(400), OCNDEP, RKMAX
     2,     SQN(400), NKT, IPPVEC
C
      COMMON // RK, ITHK, ZT(400), LWAVEC, NZT1, NZT2, ZT1D
     1,     ZTN, ITOP, IBOT, SQK, TOP1, TOP2, TOP3, TOP4, TOP7, TOP8
     2,     BOT1, BOT2, BOT3, BOT4, BOT7, BOT8, IERR, QIC(400)
     3,     NTRDT, Z, ZS, TEVAL(400), TDLDK(80), TPOBS(80)
     4,     TDPOBS(80), TDPSRC(80), TWI(80), TPSUPT(80), TPSUPB(80)
     5,     TDLDK2(80), EVEC(400,80), TEMP(400,5), DC(400)
     6,     TRIMAT(400,3), SD(400), SDL(400), SDL2(400), DL(400), D(400)
     7,     DU(400), CON(400), OMEG(400), BND(400), IEVAL(400)
     8,     EV3PST(80), PSIN2I(80)
C
C
C     UPPER AND LOWER LIMITS
      ZL = ZS - DIA/2.
      ZU = ZS + DIA/2.
C
C     TRAPEZOIDAL INTEGRATION OF N FROM ZL TO ZU
      AVE = 0.
      ZI1 = ZL
      IF (ZL .LT. ZT(1))  ZI1 = ZT(1)
      DO 10  I=2,NZT
```

```
   10 IF (ZI1 .LT. ZT(I))  GO TO 20
C     N AT LOWER LIMIT
   20 BVI1 = SQRT(SQN(I-1))
      BVI = SQRT(SQN(I))
      BVI1 = BVI1 + (BVI-BVI1) / (ZT(I)-ZT(I-1)) * (ZI1-ZT(I-1))
      DO 30  J=1,NZT
      BVI = SQRT(SQN(J))
      IF (ZT(J) .GE. ZU)  GO TO 40
      AVE = AVE + .5*(BVI+BVI1)*(ZT(J)-ZI1)
      BVI1 = BVI
   30 ZI1 = ZT(J)
      GO TO 50
C
C     N AT UPPER LIMIT
   40 BVI = BVI1 + (BVI-BVI1) / (ZT(J)-ZI1) * (ZU-ZI1)
      AVE = AVE + .5*(BVI+BVI1)*(ZU-ZI1)
   50 DTAVEN = AVE/(ZU-ZL)
      RETURN
      END
      SUBROUTINE CTBODY
C     COMPUTE DERIVATIVE D(PSI)/DZ OF EIGENFUNCTION AT SOURCE DEPTH
C
      COMMON /CONTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
     1,     NODISP, NOGRID
      EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
     1,     (MODBOD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
      COMMON /OCEAN/ LIBSEA, MODES, NK, TABK(100), NZT, DKRAT, DTDEP
     1,     TDEPMX, TDEP(400), NFLAG, SQBV(400), OCNDEP, RKMAX
     2,     SQN(400), NKT, IPPVEC
C
      COMMON // RK, ITHK, ZT(400), LWAVEC, NZT1, NZT2, ZT1D
     1,     ZTN, ITOP, IBOT, SQK, TOP1, TOP2, TOP3, TOP4, TOP7, TOP8
     2,     BOT1, BOT2, BOT3, BOT4, BOT7, BOT8, IERR, QIC(400)
     3,     NTRDT, Z, ZS, TEVAL(400), TDLDK(80), TPOBS(80)
     4,     TDPOBS(80), TDPSRC(80), TWI(80), TPSUPT(80), TPSUPB(80)
     5,     TDLDK2(80), EVEC(400,80), TEMP(400,5), DC(400)
     6,     TRIMAT(400,3), SD(400), SDL(400), SDL2(400), DL(400), D(400)
     7,     DU(400), CON(400), OMEG(400), BND(400), IEVAL(400)
     8,     EV3PST(80), PSIN2I(80)
      DIMENSION DUMMY(3),CDP(3)
C
C
C     RETURN IF SAME AS PREVIOUS CASE
      IF (LIBSEA+MODSEA+MODBOD .EQ. 0)  RETURN
C     GET COEFFS FOR COMPUTING D(PSI)/DZ AT SOURCE DEPTH
C     ON 1ST PASS OR IF OUTSIDE TCLINE
      IF (ITHK .EQ. 1)  IND = 0
      IF (IND .LE. 0)  CALL DTPSIC(ZS, IND, DUMMY, CDP)
C     COMPUTE D(PSI)/DZ AT SOURCE DEPTH
      CALL DTPSI(TDPSRC, IND, CDP)
      RETURN
      END
      SUBROUTINE DTDER
C     COMPUTE DERIVATIVE OF EIGENVALUE LAMBDA WRT K
C
      COMMON /CONTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
     1,     NODISP, NOGRID
      EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
     1,     (MODBOD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
      COMMON /OCEAN/ LIBSEA, MODES, NK, TABK(100), NZT, DKRAT, DTDEP
```

```
     1,     TDEPMX, TDEP(400), NFLAG, SQBV(400), OCNDEP, RKMAX
     2,     SQN(400), NKT, IPPVEC
C
      COMMON // RK, ITHK, ZT(400), LWAVEC, NZT1, NZT2, ZT1D
     1,     ZTN, ITOP, IBOT, SQK, TOP1, TOP2, TOP3, TOP4, TOP7, TOP8
     2,     BOT1, BOT2, BOT3, BOT4, BOT7, BOT8, IERR, QIC(400)
     3,     NTRDT, Z, ZS, TEVAL(400), TDLDK(80), TPOBS(80)
     4,     TDPOBS(80), TDPSRC(80), TWI(80), TPSUPT(80), TPSUPB(80)
     5,     TDLDK2(80), EVEC(400,80), TEMP(400,5), DC(400)
     6,     TRIMAT(400,3), SD(400), SDL(400), SDL2(400), DL(400), D(400)
     7,     DU(400), CON(400), OMEG(400), BND(400), IEVAL(400)
     8,     EV3PST(80), PSIN2I(80)
C
C
C     RETURN IF NO CHANGE FROM PREVIOUS CASE
      IF (MODSEA .EQ. 0  .AND.  LIBSEA .EQ. 0)  RETURN
C     D(LAMBDA)/DK WHERE LAMBDA=1/C**2
C     PSIN2I=NORMALIZED INTEGRAL OF (PSI*N)**2 COMPUTED IN DTEVEC
      DO 10  MODE=1,MODES
   10 TDLDK(MODE) = 2.*RK/PSIN2I(MODE)
      RETURN
      END
      SUBROUTINE DTDER2
C     COMPUTE 2ND DERIVATIVE OF EIGENVALUE LAMBDA WRT K
C
      COMMON /CONTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
     1,     NODISP, NOGRIC
      EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
     1,     (MODBOD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
      COMMON /OCEAN/ LIBSEA, MODES, NK, TABK(100), NZT, DKRAT, DTDEP
     1,     TDEPMX, TDEP(400), NFLAG, SQBV(400), OCNDEP, RKMAX
     2,     SQN(400), NKT, IPPVEC
C
      COMMON // RK, ITHK, ZT(400), LWAVEC, NZT1, NZT2, ZT1D
     1,     ZTN, ITOP, IBOT, SQK, TOP1, TOP2, TOP3, TOP4, TOP7, TOP8
     2,     BOT1, BOT2, BOT3, BOT4, BOT7, BOT8, IERR, QIC(400)
     3,     NTRDT, Z, ZS, TEVAL(400), TDLDK(80), TPOBS(80)
     4,     TDPOBS(80), TDPSRC(80), TWI(80), TPSUPT(80), TPSUPB(80)
     5,     TDLDK2(80), EVEC(400,80), TEMP(400,5), DC(400)
     6,     TRIMAT(400,3), SD(400), SDL(400), SDL2(400), DL(400), D(400)
     7,     DU(400), CON(400), OMEG(400), BND(400), IEVAL(400)
     8,     EV3PST(80), PSIN2I(80)
C
C
C     RETURN IF NO CHANGE FROM PREVIOUS CASE OR IF DERIV WAS READ
C     FROM D.T. LIBRARY
      IF (MODSEA .EQ. 0)  RETURN
      CALL TIMER(5)
      IF (RK .NE. 0.)  GO TO 20
C     TAKE LIMIT FOR K=0
C     PSIN2I IS NORMALIZED INTEGRAL OF (PSI*N)**2 COMPUTED IN DTEVEC
      DO 10  MODE=1,MODES
   10 TDLDK2(MODE) = 2./PSIN2I(MODE)
      GO TO 100
C
   20 TWOK = RK + RK
      TEMPT = RK*ZTN + TOP8*(1.-RK*ZTN*TOP8)
      TEMPB = RK*ZT1D + BOT8*(1.-RK*ZT1D*BOT8)
C     LOOP FOR EACH MODE
      DO 90  MODE=1,MODES
      EVAL = TEVAL(MODE)
```

```
      DLDK = TDLDK(MODE)
C     SET UP TRI-DIAGONAL MATRIX FOR OMEGA=D(PSI)/DK
C     FWA-1 OF EIGENVECTOR
      I1 = (MODE-1)*NZT
      DO 30  I=2,NZT1
C     SAME MATRIX AS FOR PSI BUT SUBTRACT EIGENVALUE FROM DIAGONAL
      DL(I) = TRIMAT(I)
      D(I) = TRIMAT(I+NZT) - EVAL
      DU(I) = TRIMAT(I+NZT2)
   30 CON(I) = (DLDK-TWOK/SQN(I))*EVEC(I1+I)
      DU(NZT1) = 0.
      DL(2) = 0.
C     ENFORCE BOUNDARY CONDITION IF TCLINE DOES NOT EXTEND TO SURFACE
C     (OR FLOOR).  NOTE MATRIX ELEMENTS HAVE BEEN ADJUSTED AUTOMATICALLY
      IF (ITOP .NE. 0)  CON(NZT1) = CON(NZT1)
     1                  +EVEC(I1+NZT)*DU(NZT1)*TOP4*TEMPT
      IF (IBOT .NE. 0)  CON(2) = CON(2) + EVEC(I1+2)*DL(2)*BOT4*TEMPB
C     NOTE MATRIX FOR OMEGA IS SINGULAR.  TO REMOVE SINGULARITY,
C     REPLACE ONE OF THE DIFFERENCE EQUATIONS (SAY AT POINT 3)
C     WITH A NORMALIZATION EQUATION.  CHOOSE OMEGA(3)=(D(PSI)/DK BY
C     FINITE DIFFERENCE).  PRESET NORMALIZATION FOR K=0
      DPDK = 1.
C     PICK UP EFUNCTION AT POINT 3 (FROM TOP)
      EV3 = EVEC(I1+NZT-2)
      IF (ITHK .NE. 1)  DPDK = (EV3-EV3PST(MODE)) / (RK-TABK(ITHK-1))
C     SAVE EFUNCTION FOR USE AT NEXT K
      EV3PST(MODE) = EV3
      DL(NZT-2) = 0.
      D(NZT-2) = 1.
      DU(NZT-2) = 0.
      CON(NZT-2) = DPDK
C
C     SOLVE SYSTEM OF EQUATIONS FOR OMEGA
      CALL TRID(DL(2), D(2), DU(2), CON(2), OMEG(2), NZT-2, IERR)
      IF (IERR .EQ. 0)  GO TO 50
      WRITE(6,40) MODE,ITHK,RK
   40 FORMAT(33H MATRIX FOR D(PSI)/DK IS SINGULAR,2I10,E16.8)
      GO TO 100
C     SET END POINTS OF OMEGA FROM BOUNDARY CONDITIONS
   50 OMEG(NZT) = 0.
      IF (ITOP .NE. 0)  OMEG(NZT) = TOP4 * (TOP2*OMEG(NZT1)
     1                    +TOP1*OMEG(NZT-2)-TEMPT*EVEC(I1+NZT))
      OMEG(1) = 0.
      IF (IBOT .NE. 0)  OMEG(1) = BOT4 * (BOT2*OMEG(2)
     1                    +BOT1*OMEG(3)-TEMPB*EVEC(I1+1))
C
C     POMI=INTEGRAL(PSI*OMEGA), POMNI=INTEGRAL(PSI*OMEGA*SQN)
      POMI = 0.
      POMNI = 0.
C     SKIP IF TCLINE EXTENDS TO SURFACE
      IF (ITOP .EQ. 0)  GO TO 60
C     INTEGRAL(PSI*OMEGA) FROM ZT(NZT) TO 0.
      TMP = ZTN*EVEC(I1+NZT)
      POMI=(((OMEG(NZT)/TMP-TOP8)*RK-.5/ZTN)*TOP3-.5)*TMP**2/(RK*ZTN)
C     SKIP IF TCLINE EXTENDS TO FLOOR
   60 IF (IBOT .EQ. 0)  GO TO 70
C     INTEGRAL(PSI*OMEGA) FROM -OCNDEP TO ZT(1)
      TMP = ZT1D*EVEC(I1+1)
      POMI = POMI + (((OMEG(1)/TMP-BOT8)*RK -.5/ZT1D)*BOT3 +.5)
     1           *TMP**2/(RK*ZT1D)
C     QUADRATIC INTEGRATION FROM ZT(1) TO ZT(NZT)
   70 DO 80  I=1,NZT
```

```
      QPOM = QIC(I)*EVEC(I1+I)*OMEG(I)
      POMI = POMI + QPOM
   80 POMNI = POMNI + QPOM*SQN(I)
C     SECOND DERIV OF LAMBDA WRT K
   90 TDLDK2(MODE) = ((1.-DLDK*POMNI)/RK +2.*POMI) * DLDK
  100 CALL TIMER(-5)
      RETURN
      END
      SUBROUTINE CTEVAL
C     GENERATE EIGENVALUES
C
      COMMON /CONTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
     1,      NODISP, NOGRID
      EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
     1,      (MODBOD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,      NTDTAB, NTEVEC, NTTEMP
C
      COMMON /OCEAN/ LIBSEA, MODES, NK, TABK(100), NZT, DKRAT, DTDEP
     1,      TDEPMX, TDEP(400), NFLAG, SQBV(400), OCNDEP, RKMAX
     2,      SQN(400), NKT, IPPVEC
C
      COMMON // RK, ITHK, ZT(400), LWAVEC, NZT1, NZT2, ZT1D
     1,      ZTN, ITOP, IBOT, SQK, TOP1, TOP2, TOP3, TOP4, TOP7, TOP8
     2,      BOT1, BOT2, BOT3, BOT4, BOT7, BOT8, IERR, QIC(400)
     3,      NTRDT, Z, ZS, TEVAL(400), TDLDK(80), TPOBS(80)
     4,      TDPOBS(80), TDPSRC(80), TWI(80), TPSUPT(80), TPSUPB(80)
     5,      TDLDK2(80), EVEC(400,80), TEMP(400,5), DC(400)
     6,      TRIMAT(400,3), SD(400), SDL(400), SDL2(400), DL(400), D(400)
     7,      DU(400), CON(400), OMEG(400), BND(400), IEVAL(400)
     8,      EV3PST(80), PSIN2I(80)
C
C
      CALL TIMER(3)
      IERR = 0
C     SKIP IF OCEAN HAS BEEN CHANGED FROM PREVIOUS CASE
      IF (MODSEA .NE. 0  .OR.  LIBSEA .NE. 0)  GO TO 20
C     NO CHANGE.  READ ALL DISPERSION TABLE DATA (INCLUDING
C     EIGENVALUES) PERTAINING TO THIS VALUE OF K
      IF (ITHK .GT. 1)  GO TO 3
      REWIND NTRDT
      READ(NTRDT) MODEST
    3 READ(NTRDT) RK,(TEVAL(M),TDLDK(M),TPOBS(M),TDPOBS(M),TDPSRC(M),
     1            TWI(M),TPSUPT(M),TPSUPB(M),TDLDK2(M), M=1,MODEST)
      IF (EOF,NTRDT) 5,10
    5 IERR = 1
   10 TABK(ITHK) = RK
      GO TO 190
C
C     COMPUTE EIGENVALUES.  SKIP IF NOT FIRST PASS
   20 IF (ITHK .NE. 1)  GO TO 50
C     LOOP THROUGH POINTS IN TCLINE TABLE. NZT2=FWA-1 OF UPPER DIAGONAL
      DO 30  I=2,NZT1
      D32 = ZT(I+1) - ZT(I)
      D31 = ZT(I+1) - ZT(I-1)
      D21 = ZT(I) - ZT(I-1)
C     SET UP PARAMETER USED IN COMPUTING MATRIX DIAGONAL
      DC(I) = 2./(D32*D21)
C     LOWER AND UPPER DIAGONALS OF THE TRI-DIAGONAL MATRIX
      TRIMAT(I) = -2. / (D31*D21*SQN(I))
   30 TRIMAT(I+NZT2) = -2. / (D31*D32*SQN(I))
```

```
C      SKIP IF TOP OF TCLINE IS AT SURFACE
       IF (ITOP .EQ. 0)  GO TO 40
C      LOWER DIAGONAL ELEMENT NZT-1 WILL VARY WITH K.  SAVE CURRENT VALUE
       DLSAV = TRIMAT(NZT1)
C      NOTE THAT HERE D32 = ZT(NZT)-ZT(NZT-1), D31=ZT(NZT)-ZT(NZT-2),
C           D21=ZT(NZT-1)-ZT(NZT-2)
       TOP1 = D32/(D31*D21)
       TOP2 = -D31/(D32*D21)
C      SKIP IF BOTTOM OF TCLINE IS AT FLOOR
    40 IF (IBOT .EQ. 0)  GO TO 50
C      UPPER DIAG ELEMENT (2) WILL VARY WITH K.  SAVE CURRENT VALUE
       DUSAV = TRIMAT(NZT2+2)
       BOT1 = -(ZT(2)-ZT(1)) / ((ZT(3)-ZT(1)) *(ZT(3)-ZT(2)))
       BOT2 = (ZT(3)-ZT(1)) / ((ZT(3)-ZT(2)) *(ZT(2)-ZT(1)))
C
C
C      COMPUTATIONS FOR EACH VALUE OF K
    50 RK = TABK(ITHK)
       SQK = RK**2
C      GENERATE DIAGONAL ELEMENTS OF THE TRI-DIAGONAL MATRIX
       DO 60  I=2,NZT1
    60 TRIMAT(I+NZT) = (SQK+DC(I)) / SQN(I)
C
C      SKIP IF TCLINE EXTENDS TO SURFACE
       IF (ITOP .EQ. 0)  GO TO 90
       IF (RK .NE. 0.)  GO TO 70
C      TAKE LIMIT FOR K=0
       TOP4 = 1./(TOP1+TOP2+1./ZTN)
       GO TO 80
    70 TOP7 = EXP(2.*RK*ZTN)
C      COTH(RK*ZTN)
       TOP8 = (TOP7+1.) / (TOP7-1.)
       TOP4 = 1./(RK*TOP8+TOP1+TOP2)
C      RESET NEXT-TO-LAST ELEMENTS OF DIAGONAL AND LOWER DIAGONAL
    80 TRIMAT(NZT2-1) = TRIMAT(NZT2-1) + TOP4*TOP2*TRIMAT(NZT2+NZT1)
       TRIMAT(NZT1) = DLSAV + TOP4*TOP1*TRIMAT(NZT2+NZT1)
C
C      SKIP IF TCLINE EXTENDS TO FLOOR
    90 IF (IBOT .EQ. 0)  GO TO 120
       IF (RK .NE. 0.)  GO TO 100
C      TAKE LIMIT FOR K=0
       BOT4 = 1./(BOT1+BOT2+1./ZT1D)
       GO TO 110
   100 BOT7 = EXP(-2.*RK*ZT1D)
       BOT8 = (1.+BOT7) / (1.-BOT7)
       BOT4 = 1./(RK*BOT8+BOT1+BOT2)
C      RESET ELEMENTS OF DIAGONAL AND UPPER DIAGONAL
   110 TRIMAT(NZT+2) = TRIMAT(NZT+2) + BOT4*BOT2*TRIMAT(2)
       TRIMAT(NZT2+2) = DUSAV + BOT4*BOT1*TRIMAT(2)
C
C      SYMMETRIZE THE MATRIX
   120 CALL FIGI(NZT, NZT-2, TRIMAT(2), SD, SDL, SDL2, IERR)
       IF (IERR .EQ. 0)  GO TO 140
       WRITE(6,130) IERR,ITHK,RK
   130 FORMAT(29H ERROR IN SYMMETRIZING MATRIX,2I10,E16.8)
       GO TO 190
C
C
C      SKIP IF EIGENVALUES MUST BE COMPUTED
   140 IF (MODSEA .NE. 0)  GO TO 170
C      READ DISPERSION TABLE LIBRARY FILE
       READ(NTDLIB) (TEVAL(M),IEVAL(M),TDLDK2(M), M=1,MODES)
```

```
      IF (EOF,NTDLIB) 150,160
  150 IERR = 1
      GO TO 190
C     SPLIT MATRIX INTO SUB-MATRICES IF OFF-DIAGONAL ELEMENTS
C     ARE NEGLIGIBLE (REQUIRED BY TINVIT, NORMALLY DONE BY RATQR)
C     PMACHE IS A MACHINE DEPENDENT PARAMETER (=MACHEP IN RATQR)
  160 PMACHE = 2.**(-47)
      SDL2(1) = 0.
      LIM = NZT - 2
      DO 155  I=2,LIM
  165 IF (ABS(SDL(I)) .LE. PMACHE*(ABS(SD(I))+ABS(SD(I-1))))
     1                                                    SDL2(I) = 0.
      GO TO 190
C
C     COMPUTE THE LOWEST EIGENVALUES
  170 EPS1 = 0.
      IDEF = 1
      CALL RATQR(NZT-2, EPS1, SD, SDL, SDL2, MODES, TEVAL, IEVAL,
     1             BND, .TRUE., IDEF, IERR)
      IF (IERR .EQ. 0)  GO TO 190
      WRITE(6,180) IERR,ITHK,RK
  180 FORMAT(31H ERROR IN COMPUTING EIGENVALUES,2I10,E16.8)
  190 CALL TIMER(-3)
      RETURN
      END
      SUBROUTINE DTEVEC
C     GENERATE EIGENVECTORS
C
      COMMON /CONTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
     1,      NODISP, NOGRID
      EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
     1,      (MODBCD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,      NTDTAB, NTEVEC, NTTEMP
C
      COMMON /OCEAN/ LIBSEA, MODES, NK, TABK(100), NZT, DKRAT, DTDEP
     1,      TDEPMX, TDEP(400), NFLAG, SQBV(400), OCNDEP, RKMAX
     2,      SQN(400), NKT, IPPVEC
C
      COMMON // RK, ITHK, ZT(400), LWAVEC, NZT1, NZT2, ZT1D
     1,      ZTN, ITOP, IBOT, SQK, TOP1, TOP2, TOP3, TOP4, TOP7, TOP8
     2,      BOT1, BOT2, BOT3, BOT4, BOT7, BOT8, IERR, QIC(400)
     3,      NTRCT, Z, ZS, TEVAL(400), TDLDK(80), TPOBS(80)
     4,      TDPOBS(80), TDPSRC(80), TWI(80), TPSUPT(80), TPSUPB(80)
     5,      TDLDK2(80), EVEC(400,80), TEMP(400,5), DC(400)
     6,      TRIMAT(400,3), SD(400), SDL(400), SDL2(400), DL(400), D(400)
     7,      DU(400), CON(400), OMEG(400), BND(400), IEVAL(400)
     8,      EV3PST(80), PSIN2I(80)
C
C
      CALL TIMER(4)
C     JUMP IF OCEAN HAS BEEN CHANGED FROM PREVIOUS CASE
      IF (MODSEA .NE. 0  .OR.  LIBSEA .NE. 0)  GO TO 50
C     NO CHANGE.  READ ALL EIGENVECTORS FOR THIS K
      BUFFER IN(NTEVEC,1) (EVEC(1),EVEC(LWAVEC))
C     WAIT FOR READ TO BE COMPLETED
   10 IF (UNIT,NTEVEC) 10,220,20,20
   20 WRITE(6,30) ITHK,MODES,NZT
   30 FORMAT(25H EOF READING EIGENVECTORS,3I10)
      CALL ERRXIT
C
```

```
C
C     COMPUTE EIGENVECTORS.  SKIP IF NOT FIRST PASS
   50 IF (ITHK .NE. 1)  GO TO 70
C     SET UP COEFFICIENTS TO QUADRATICLY INTEGRATE FUNCTIONS
C     SPECIFIED AT POINTS ZT(I)
      QIC(1) = 0.
C     INTEGRATE PARABOLA FITTED TO POINTS 1-2-3, 3-4-5, 5-6-7...
C     LOOP FOR EACH PARABOLA
      DO 60  I=2,NZT1,2
      D32 = ZT(I+1) - ZT(I)
      D31 = ZT(I+1) - ZT(I-1)
      D21 = ZT(I) - ZT(I-1)
      QIC(I-1) = QIC(I-1) + (1.-.5*D32/D21)*D31/3.
      QIC(I)   = D31**3 / (6.*D32*D21)
   60 QIC(I+1) = (1.-.5*D21/D32)*D31/3.
C     SKIP IF NZT IS ODD
      IF (2*(NZT/2) .NE. NZT)  GO TO 70
C     SET UP COEFFS TO INTEGRATE FROM ZT(NZT-1) TO ZT(NZT)
      QIC(NZT-2) = QIC(NZT-2) - D32**3/(6.*D31*D21)
      QIC(NZT1) = QIC(NZT1) + (3.+D32/D21)*D32/6.
      QIC(NZT) = (2.+D21/D31)*D32/6.
C
C     COMPUTE EIGENVECTORS FOR THIS K.
C     WAIT UNTIL FINISHED WRITING OLD SET OF EVECTORS
   70 IF (UNIT,NTEVEC) 70,80,80,80
C     EIGENVECTORS CORRESPONDING TO SYMMETRIC MATRIX
   80 CALL TINVIT(NZT, NZT-2, SD, SDL, SDL2, MODES, TEVAL, IEVAL,
     1        EVEC(2), IERR, TEMP(1,1),TEMP(1,2),TEMP(1,3),TEMP(1,4),
     2        TEMP(1,5))
      IF (IERR .EQ. 0)  GO TO 100
      WRITE(6,90) IERR,ITHK,RK
   90 FORMAT(32H ERROR IN COMPUTING EIGENVECTORS,2I10,E16.8)
      GO TO 220
C     TRANSFORM EIGENVECTORS BACK TO NON-SYMMETRIC MATRIX SYSTEM
  100 CALL BAKVEC(NZT, NZT-2, TRIMAT(2), SDL, MODES, EVEC(2), IERR)
      IF (IERR .EQ. 0)  GO TO 120
      WRITE(6,110) IERR,ITHK,RK
  110 FORMAT(40H ERROR IN BACK TRANSFORMING EIGENVECTORS,2I10,E16.8)
      GO TO 220
C
C     SKIP IF TOP OF TCLINE IS AT SURFACE
  120 IF (ITOP .EQ. 0)  GO TO 140
C     PARAMETER USED TO INTEGRATE FROM ZT(NZT) TO SURFACE
      IF (RK .NE. 0.)  GO TO 130
C     TAKE LIMIT FOR K=0
      TOP3 = -ZTN/3.
      GO TO 140
C     TOP7,8 COMPUTED IN DTEVAL.  TOP7=EXP(2.*RK*ZTN)
C           TOP8=COTH(RK*ZTN)
  130 TOP3 = 2.*ZTN*TOP7/(TOP7-1.)**2 - .5*TOP8/RK
C     SKIP IF BOTTOM OF TCLINE IS AT FLOOR
  140 IF (IBOT .EQ. 0)  GO TO 160
C     PARAMETER USED TO INTEGRATE FROM FLOOR TO ZT(1)
      IF (RK .NE. 0.)  GO TO 150
C     TAKE LIMIT FOR K=0
      BOT3 = ZT1D/3.
      GO TO 160
C     BOT7,8 COMPUTED IN DTEVAL.  BOT7=EXP(-2.*RK*ZT1D)
C           BOT8=COTH(RK*ZT1D)
  150 BOT3 = -2.*ZT1D*BOT7/(1.-BOT7)**2 + .5*BOT8/RK
C
C     NORMALIZE EACH EIGENVECTOR.  I1=FWA OF EVECTOR, IE=LWA
```

```
  160 IE = 0
      DO 210  MODE=1,MODES
      I1 = IE +1
      IE = IE + NZT
C     P2I IS NORMALIZATION INTEGRAL FOR CURRENT EIGENVECTOR
      P2I = 0.
      EVEC(IE) = 0.
C     SKIP IF TCLINE EXTENDS TO SURFACE
      IF (ITOP .EQ. 0)  GO TO 170
C     ENFORCE BOUNDARY CONDITION AT TOP OF TCLINE.  TOP1,2,4
C     ARE SET IN DTEVAL
      EVEC(IE) = TOP4* (TOP1*EVEC(IE-2)+TOP2*EVEC(IE-1))
C     INTEGRAL FROM ZT(NZT) TO 0.
      P2I = TOP3*EVEC(IE)**2
C
  170 EVEC(I1) = 0.
C     SKIP IF TCLINE EXTENDS TO FLOOR
      IF (IBOT .EQ. 0)  GO TO 180
C     ENFORCE BOUNDARY CONDITION AT BOTTOM OF TCLINE.  BOT1,2,4 ARE
C     SET IN DTEVAL
      EVEC(I1) = BOT4* (BOT1*EVEC(I1+2)+BOT2*EVEC(I1+1))
C     ADD IN INTEGRAL FROM -OCNDEP TO ZT(1)
      P2I = P2I + BOT3*EVEC(I1)**2
C
C     QUADRATIC INTEGRATION OF EVECTOR**2 FROM ZT(1) TO ZT(NZT)
C     INTEGRATE (EVECTOR*N)**2 AT SAME TIME
  180 PN2I = 0.
      J= 1
      DO 190  I=I1,IE
      QP2 = QIC(J)*EVEC(I)**2
      P2I = P2I + QP2
      PN2I = PN2I + QP2*SQN(J)
  190 J = J+1
C
C     NORMALIZE AND SAVE INTEGRAL (PSI*N)**2
      PSIN2I(MODE) = PN2I/P2I
C     NORMALIZE BY MULTIPLYING EVECTOR BY ENORM
      ENORM = 1./SQRT(P2I)
C     ATTACH SIGN OF 1ST NON-ZERO ELEMENT OF EIGENVECTOR TO
C     NORMALIZATION CONSTANT TO ENSURE ALL EIGENVECTORS FOR A GIVEN
C     MODE HAVE THE SAME PARITY
      ENORM = SIGN(ENORM,EVEC(IE))
      IF (EVEC(IE) .EQ. 0.)  ENORM = SIGN(ENORM,EVEC(IE-1))
C     DO THE NORMALIZATION
      DO 200  I=I1,IE
  200 EVEC(I) = ENORM*EVEC(I)
  210 CONTINUE
C
C     START WRITING THE EIGENVECTORS AND PROCEED WITH COMPUTATIONS
      BUFFER OUT(NTEVEC,1) (EVEC(1),EVEC(LWAVEC))
C
  220 CALL TIMER(-4)
      RETURN
      END
      SUBROUTINE DTINIT
C     INITIALIZATION FOR D.T. COMPUTATIONS
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,     RBSEP2, RBSTR, RBLIM
C
      COMMON /CONTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
     1,     NODISP, NOGRID
```

```
      EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
     1,      (MODBOD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,      NTDTAB, NTEVEC, NTTEMP
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,      X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,      IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,      ISPHAS
C
      COMMON /OCEAN/ LIBSEA, MODES, NK, TABK(100), NZT, DFRAT, DTDEP
     1,      TDEPMX, TDEP(400), NFLAG, SQBV(400), OCNDEP, RKMAX
     2,      SQN(400), NKT, IPPVEC
C
      COMMON // RK, ITHK, ZT(400), LWAVEC, NZT1, NZT2, ZT1D
     1,      ZTN, ITOP, IBOT, SQK, TOP1, TOP2, TOP3, TOP4, TOP7, TOP8
     2,      BOT1, BOT2, BOT3, BOT4, BOT7, BOT8, IERR, QIC(400)
     3,      NTRDT, Z, ZS, TEVAL(400), TDLDK(80), TPOBS(80)
     4,      TOPOBS(80), TOPSRC(80), TWI(80), TPSUPT(80), TPSUPB(80)
     5,      TDLDK2(80), EVEC(400,80), TEMP(400,5), DC(400)
     6,      TRIMAT(400,3), SD(400), SDL(400), SDL2(400), DL(400), D(400)
     7,      DU(400), CON(400), OMEG(400), BND(400), IEVAL(400)
     8,      EV3PST(80), PSIN2I(80)
C
C
C     FLIP LOGICAL TAPE NUMBERS--NTRDT=FILE FROM WHICH TO READ DT FROM
C     PREVIOUS CASE, NTDTAB=FILE ON WHICH TO WRITE NEW DT,
C     NTTEMP=TEMP STORAGE (NOT USED BY DT MODULE)
      NTRDT = NTDTAB
      NTDTAB = NTTEMP
      NTTEMP = NTRDT
C
C     EIGENVECTOR FILE HAS PARALLEL I/O.  BE SURE ITS READY
   10 IF (UNIT,NTEVEC)  10,20,20,20
   20 REWIND NTEVEC
   30 IF (UNIT,NTEVEC)  30,40,40,40
C
C     SKIP IF OCEAN DATA WAS INPUT
   40 IF (MODSEA .NE. 0)  GO TO 70
C     DT LIBRARY FILE IS USED ONLY ON 1ST PASS OF 1ST CASE
C     (AFTER THAT, DT SAVER FILE IS USED)
      IF (ICASE .NE. 1  .OR.  ITHOBS .GT. 1)  GO TO 80
C     VERY 1ST PASS.  SKIP IF USING DT LIBRARY
      IF (LIBSEA .NE. 0)  GO TO 60
      WRITE(6,50)
   50 FORMAT(24H NO OCEAN DATA SPECIFIED)
      CALL ERRXIT
C     READ 1ST RECORD OF DT LIBRARY
   60 REWIND NTDLIB
      READ(NTDLIB) NKT,(TABK(I),I=1,NKT),NZT,(TDEP(I),I=1,NZT)
     1,      (SQN(I),I=1,NZT),OCNDEP,MODES
      GO TO 90
C     SET UP K LIST AND TCLINE TABLE
   70 CALL DTSETK
      CALL DTSETN
C     START NEW DT LIBRARY
      REWIND NTDLIB
      WRITE(NTDLIB) NKT,(TABK(I),I=1,NKT),NZT,(TDEP(I),I=1,NZT)
     1,      (SQN(I),I=1,NZT),OCNDEP,MODES
C
   80 LIBSEA = 0
```

```
C      CONVERT INCREASING TCLINE DEPTHS TO INCREASING Z COORDINATE
   90 DO 100  I=1,NZT
  100 ZT(I) = -TDEP(NZT-I+1)
C
C      ASSORTED WIDELY USED VARIABLES
       NZT1 = NZT-1
       NZT2 = NZT+NZT
C      LWA OF LAST EIGENVECTOR
       LWAVEC = NZT*MODES
       ZT1D = ZT(1) + OCNDEP
       ZTN = ZT(NZT)
       Z = -OBSDEP
       ZS = -BODDEP
C      SET FLAG ITOP=0 IF TOP OF TCLINE IS AT SURFACE
       ITOP = 0
       IF (ABS(ZTN/ZT(1)) .GT. 1.E-10)  ITOP = 1
C      SET FLAG IBOT=0 IF BOTTOM OF TCLINE IS AT OCEAN FLOOR
       IBOT = 0
       IF (ABS(ZT1D/ZT(1)) .GT. 1.E-10)  IBOT = 1
       RETURN
       END
       SUBROUTINE DTOBS
C      COMPUTE AND STORE EIGENFUNCTION AND ITS DERIVATIVE AT OBS DEPTH
C
       COMMON /CONTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
      1,     NODISP, NOGRID
       EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
      1,     (MODBOD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
       COMMON /OCEAN/ LIBSEA, MODES, NK, TABK(100), NZT, DKRAT, DTDEP
      1,     TDEPMX, TDEP(400), NFLAG, SQBV(400), OCNDEP, RKMAX
      2,     SGN(400), NKT, IPPVEC
C
       COMMON // RK, ITHK, ZT(400), LWAVEC, NZT1, NZT2, ZT1D
      1,     ZTN, ITOP, IBOT, SQK, TOP1, TOP2, TOP3, TOP4, TOP7, TOP8
      2,     BOT1, BOT2, BOT3, BOT4, BOT7, BOT8, IERR, QIC(400)
      3,     NTROT, Z, ZS, TEVAL(400), TDLDK(80), TPOBS(80)
      4,     TDPOBS(80), TDPSRC(80), TWI(80), TPSUPT(80), TPSUPB(80)
      5,     TDLDK2(80), EVEC(400,80), TEMP(400,5), DC(400)
      6,     TRIMAT(400,3), SD(400), SDL(400), SDL2(400), DL(400), D(400)
      7,     DU(400), CON(400), OMEG(400), BND(400), IEVAL(400)
      8,     EV3PST(80), PSIN2I(80)
       DIMENSION CP(3),CDP(3)
C
C
C      RETURN IF SAME AS PREVIOUS CASE
       IF (LIBSEA+MODSEA+MODOBS .EQ. 0)  RETURN
C      GET COEFFS FOR COMPUTING PSI AND D(PSI)/DZ AT OBSERVATION DEPTH
C      ON 1ST PASS OR IF Z IS OUTSIDE TCLINE
       IF (ITHK .EQ. 1)  IND = 0
       IF (IND .LE. 0)  CALL DTPSIC(Z, IND, CP, CDP)
C      GENERATE PSI FOR EACH MODE
       CALL DTPSI(TPOBS, IND, CP)
C      GENERATE D(PSI)/DZ FOR EACH MODE
       CALL DTPSI(TDPOBS, IND, CDP)
       RETURN
       END
       SUBROUTINE DTPSI(PSI, IND, COEF)
C      GENERATE EIGENFUNCTION (OR ITS DERIVATIVE)
C
       COMMON /OCEAN/ LIBSEA, MODES, NK, TABK(100), NZT, DKRAT, DTDEP
      1,     TDEPMX, TDEP(400), NFLAG, SQBV(400), OCNDEP, RKMAX
```

```
     2,     SQN(400), NKT, IPPVEC
C
      COMMON // RK, ITHK, ZT(400), LWAVEC, NZT1, NZT2, ZT1D
     1,     ZTN, ITOP, IBOT, SQK, TOP1, TOP2, TOP3, TOP4, TOP7, TOP8
     2,     BOT1, BOT2, BOT3, BOT4, BOT7, BOT8, IERR, QIC(400)
     3,     NTRDT, Z, ZS, TEVAL(400), TDLDK(80), TPDBS(80)
     4,     TDPOBS(80), TDPSRC(80), TWI(80), TPSUPT(80), TPSUPB(80)
     5,     TDLDK2(80), EVEC(400,80), TEMP(400,5), DC(400)
     6,     TRIMAT(400,3), SD(400), SDL(400), SDL2(400), DL(400), D(400)
     7,     DU(400), CON(400), OMEG(400), BND(400), IEVAL(400)
     8,     EV3PST(80), PSIN2I(80)
      DIMENSION COEF(3), PSI(1)
C
C
C     PRESET TO PICK UP LAST ELEMENT OF EVECTOR
      J = NZT
C     JUMP IF DESIRED DEPTH IS BELOW,ABOVE,INSIDE TCLINE
      IF (IND) 10,20,40
C     BELOW TCLINE.  SET TO PICK UP 1ST ELEMENT OF EVECTOR
   10 J = 1
C     LOOP FOR EACH EVECTOR
   20 DO 30  MODE=1,MODES
C     ANALYTIC EXPRESSION
      PSI(MODE) = COEF*EVEC(J)
   30 J = J + NZT
      RETURN
C
C     INSIDE TCLINE.  LOOP FOR EACH EVECTOR
   40 J = IND
      DO 50  MODE=1,MODES
C     QUADRATIC INTERPOLATION
      PSI(MODE) = COEF(1)*EVEC(J-1)+ COEF(2)*EVEC(J)+ COEF(3)*EVEC(J+1)
   50 J = J + NZT
      RETURN
      END
      SUBROUTINE DTPSIC(ZDES, IND, CP, CDP)
C     GENERATE COEFFICIENTS FOR DETERMINING EFUNCTION AND DERIV AT
C
      COMMON /OCEAN/ LIBSEA, MODES, NK, TABK(100), NZT, DKRAT, DTDEP
     1,     TDEPMX, TDEP(400), NFLAG, SQBV(400), OCNDEP, RKMAX
     2,     SQN(400), NKT, IPPVEC
C
      COMMON // RK, ITHK, ZT(400), LWAVEC, NZT1, NZT2, ZT1D
     1,     ZTN, ITOP, IBOT, SQK, TOP1, TOP2, TOP3, TOP4, TOP7, TOP8
     2,     BOT1, BOT2, BOT3, BOT4, BOT7, BOT8, IERR, QIC(400)
     3,     NTRDT, Z, ZS, TEVAL(400), TDLDK(80), TPDBS(80)
     4,     TDPOBS(80), TDPSRC(80), TWI(80), TPSUPT(80), TPSUPB(80)
     5,     TDLDK2(80), EVEC(400,80), TEMP(400,5), DC(400)
     6,     TRIMAT(400,3), SD(400), SDL(400), SDL2(400), DL(400), D(400)
     7,     DU(400), CON(400), OMEG(400), BND(400), IEVAL(400)
     8,     EV3PST(80), PSIN2I(80)
C     POSITION ZDES
      DIMENSION CP(3), CDP(3)
C
C
C     SKIP IF DESIRED POINT IS BELOW TOP OF TCLINE
      IF (ZDES .LE. ZTN)  GO TO 20
C     PROTECT FROM LOW ORDER BITS IF TCLINE GOES TO SURFACE
      IF (ITOP .EQ. 0)  GO TO 20
C     SET FLAG SHOWING ABOVE TCLINE
      IND = 0
      IF (RK .NE. 0.)  GO TO 10
```

```
      CP = ZDES/ZTN
      CDP = 1./ZTN
      GO TO 70
   10 TMP = EXP(RK*(ZTN-ZDES)) / (EXP(2.*RK*ZTN)-1.)
      TMP1 = EXP(2.*RK*ZDES)
C        CP = SINH(RK*ZDES)/SINH(RK*ZTN)
      CP = TMP*(TMP1-1.)
C        CDP = RK*COSH(RK*ZDES)/SINH(RK*ZTN)
      CDP = RK*TMP*(TMP1+1.)
      GO TO 70
C
C     SKIP IF DESIRED POINT IS ABOVE BOTTOM OF TCLINE
   20 IF (ZDES .GE. ZT(1))  GO TO 40
      IF (IBOT .EQ. 0)  GO TO 40
C     SET FLAG SHOWING BELOW TCLINE
      IND = -1
      IF (RK .NE. 0.)  GO TO 30
      CP = (ZDES+OCNDEP)/ZT1D
      CDP = 1./ZT1D
      GO TO 70
   30 TMP = EXP(RK*(ZDES-ZT(1))) / (1.-EXP(-2.*RK*ZT1D))
      TMP1 = EXP(-2.*RK*(ZDES+OCNDEP))
C        CP = SINH(RK*(ZDES+OCNDEP)) / SINH(RK*ZT1D)
      CP = TMP*(1.-TMP1)
C        CDP = RK*COSH(RK*(ZDES+OCNDEP)) / SINH(RK*ZT1D)
      CDP = RK*TMP*(1.+TMP1)
      GO TO 70
C
C     ZDES IS WITHIN THE TCLINE
C     INCOMING VALUE OF IND IS LOWER LIMIT TO SEARCH FOR I,
C     WHERE ZT(I-1) .LE. ZDES .LE. ZT(I)
   40 LIM = MAX0(3,IND)
      DO 50  I=LIM,NZT1
   50 IF (ZDES .LE. ZT(I))  GO TO 60
      I = NZT1
C     ADJUST I SO THAT ZT(I) IS CLOSEST TABLE POINT TO ZDES
   60 IF (ZDES-ZT(I-1) .LT. ZT(I)-ZDES)  I = I-1
C     SAVE POSITION IN TABLE.  NOTE IND .GT. 0 IMPLIES WITHIN TCLINE
      IND = I
      DEL = ZDES - ZT(I-1)
      DEL2 = 2.*DEL
      D32 = ZT(I+1) - ZT(I)
      D31 = ZT(I+1) - ZT(I-1)
      D21 = ZT(I) - ZT(I-1)
C     COEFFS FOR QUADRATIC INTERPOLATION OF EIGENFUNCTION PSI
      CP(1) = 1. + DEL*(DEL-D31-D21) /(D31*D21)
      CP(2) = DEL*(D31-DEL) /(D32*D21)
      CP(3) = DEL*(DEL-D21) /(D32*D31)
C     COEFFS FOR INTERPOLATING D(PSI)/DZ
      CDP(1) = (DEL2-D31-D21) / (D21*D31)
      CDP(2) = (D31-DEL2) / (D32*D21)
      CDP(3) = (DEL2-D21) / (D32*D31)
C
   70 RETURN
      END
      SUBROUTINE DTSETK
C     SET UP WAVE NUMBER K LIST
C
      COMMON /CONST/ JDK, JDMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,     JDCKSV, JDMSP, JDEDGE
C
      COMMON /OCEAN/ LIBSEA, MODES, NK, TABK(100), NZT, DKRAT, DTDEP
```

```
     1,     TDEPMX, TDEP(400), NFLAG, SQBV(400), OCNDEP, RKMAX
     2,     SQN(400), NKT, IPPVEC
C
C
C     SKIP IF DESIRED SIZE OF K LIST IS WITHIN DIMENSION
      IF (0 .LT. NK  .AND.  NK .LE. JDK)  GO TO 20
      WRITE(6,10) NK,JDK
   10 FORMAT(4H NK=,I4,23H IS ILLEGAL. DIMENSION=,I4)
      CALL ERRXIT
C     SKIP IF K LIST WAS INPUT DIRECTLY
   20 IF (DKRAT .LE. 0.)  GO TO 50
C     PRESET FOR EQUAL INTERVAL TABLE
      C = 1.
      DEL = RKMAX/FLOAT(NK-1)
      IF (DKRAT .EQ. 1.)  GO TO 30
C     GENERATE K LIST WITH DELTA K INCREASED BY FACTOR C FOR EACH POINT
      C = DKRAT**(1./FLOAT(NK-2))
      DEL = RKMAX * (C-1.) / (C*DKRAT-1.)
   30 TABK(1) = 0.
      DO 40  I=2,NK
      TABK(I) = TABK(I-1) + DEL
   40 DEL = C*DEL
C
   50 NKT = NK
      RETURN
      END
      SUBROUTINE DTSETN
C     SET UP THERMOCLINE TABLE
C
      COMMON /CONST/ JDK, JDMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,     JDCKSV, JDMSP, JDEDGE
C
      COMMON /OCEAN/ LIBSEA, MODES, NK, TABK(100), NZT, DKRAT, DTDEP
     1,     TDEPMX, TDEP(400), NFLAG, SQBV(400), OCNDEP, RKMAX
     2,     SQN(400), NKT, IPPVEC
C
C
C     SKIP IF DESIRED SIZE OF TCLINE TABLE IS ACCEPTABLE
      IF (3 .LE. NZT  .AND.  NZT .LE. JDTCL)  GO TO 20
      WRITE(6,10) NZT,JDTCL
   10 FORMAT(5H NZT=,I4,23H IS ILLEGAL. DIMENSION=,I4)
      CALL ERRXIT
C     PRESET INTERNAL INCREMENT IN TCLINE DEPTHS
   20 DEL = DTDEP
C     IF IT WAS INPUT, USE IT TO CONSTRUCT LIST OF DEPTHS
      IF (DEL .NE. 0.)  GO TO 30
C     IF MAX DEPTH IS ALSO ZERO, LIST WAS INPUT DIRECTLY
      IF (TDEPMX .EQ. 0.)  GO TO 50
C     COMPUTE INCREMENT FROM INPUT MAX, MIN AND NUMBER OF POINTS
      DEL = (TDEPMX-TDEP(1)) / FLOAT(NZT-1)
C     CONSTRUCT EQUAL INCREMENT TABLE
   30 DO 40  I=2,NZT
   40 TDEP(I) = TDEP(I-1) + DEL
C
C     SET UP LIST OF N**2(ZT)
   50 DO 60  I=1,NZT
   60 SQN(I) = SQBV(NZT-I+1)
C     JUMP IF IT WAS REALLY N**2 INPUT INTO SQBV
      IF (NFLAG .EQ. 0)  GO TO 80
C     IT WAS N (=BRUNT VAISALA FREQUENCY).  CONVERT TO N**2
      DO 70  I=1,NZT
   70 SQN(I) = SQN(I)**2
```

```
   80 RETURN
      END
      SUBROUTINE DTSUPR
C     COMPUTE, STORE EIGENFUNCTION AT TOP AND BOTTOM OF SUPERSTRUCTURE
C
      COMMON /CONTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
     1,     NODISP, NOGRID
      EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
     1,     (MODBOD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
      COMMON /OCEAN/ LIBSEA, MODES, NK, TABK(100), NZT, DKRAT, DTDEP
     1,     TDEPMX, TDEP(400), NFLAG, SQBV(400), OCNDEP, RKMAX
     2,     SQN(400), NKT, IPPVEC
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,     SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMMON // RK, ITHK, ZT(400), LWAVEC, NZT1, NZT2, ZT1D
     1,     ZTN, ITOP, IBOT, SQK, TOP1, TOP2, TOP3, TOP4, TOP7, TOP8
     2,     BOT1, BOT2, BOT3, BOT4, BOT7, BOT8, IERR, QIC(400)
     3,     NTRDT, Z, ZS, TEVAL(400), TDLDK(80), TPOBS(80)
     4,     TDPOBS(80), TDPSRC(80), TWI(80), TPSUPT(80), TPSUPB(80)
     5,     TDLDK2(80), EVEC(400,80), TEMP(400,5), DC(400)
     6,     TRIMAT(400,3), SD(400), SDL(400), SDL2(400), DL(400), D(400)
     7,     DU(400), CON(400), OMEG(400), BND(400), IEVAL(400)
     8,     EV3PST(80), PSIN2I(80)
      DIMENSION CPT(3), CPB(3), DUMMY(3)
C
C
C     RETURN IF SAME AS PREVIOUS CASE
      IF (LIBSEA+MODSEA+MODSUP .EQ. 0)  RETURN
C     GENERATE COEFFS FOR COMPUTING PSI AT TOP AND BOTTOM OF LINE
C     SOURCE/SINK ON 1ST PASS OR IF OUTSIDE OF TCLINE
      IF (ITHK .NE. 1)  GO TO 10
      INDT = 0
      INDB = 0
   10 IF (INDT .LE. 0)  CALL DTPSIC(ZS+SUPTOP, INDT, CPT, DUMMY)
      IF (INDB .LE. 0)  CALL DTPSIC(ZS+SUPBOT, INDB, CPB, DUMMY)
C     GENERATE PSI FOR ALL MODES AT TOP AND BOTTOM OF SUPERSTRUCTURE
      CALL DTPSI(TPSUPT, INDT, CPT)
      CALL DTPSI(TPSUPB, INDB, CPB)
      RETURN
      END
      SUBROUTINE DTWAKE
C     EVALUATE WAKE INTEGRAL
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,     RBSEP2, RBSTR, RBLIM
C
      COMMON /CONST/ JOK, JDMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,     JDCKSV, JDMSP, JDEDGE
C
      COMMON /CONTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
     1,     NODISP, NOGRID
      EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
     1,     (MODBOD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
      COMMON /OCEAN/ LIBSEA, MODES, NK, TABK(100), NZT, DKRAT, DTDEP
     1,     TDEPMX, TDEP(400), NFLAG, SQBV(400), OCNDEP, RKMAX
     2,     SQN(400), NKT, IPPVEC
C
      COMMON /WAKE/ IWAKE, CWAKR, CWAKX, XWAKE, WAKRAD, XWNOM
```

```
     1.      RESLVS, CWAKM
C
      COMMON // RK, ITHK, ZT(400), LWAVEC, NZT1, NZT2, ZT1D
     1,      ZTN, ITOP, IBOT, SQK, TOP1, TOP2, TOP3, TOP4, TOP7, TOP8
     2,      BOT1, BOT2, BOT3, BOT4, BOT7, BOT8, IERR, QID(400)
     3,      NTROT, Z, ZS, TEVAL(400), TDLDK(80), TPOBS(80)
     4,      TDPOBS(80), TDPSRC(80), TWI(80), TPSUPT(80), TPSUPB(80)
     5,      TDLDK2(80), EVEC(400,80), TEMP(400,5), DC(400)
     6,      TRIMAT(400,3), SD(400), SDL(400), SDL2(400), DL(400), D(400)
     7,      DU(400), CON(400), OMEG(400), BND(400), IEVAL(400)
     8,      EV3PST(80), PSIN2I(80)
      DIMENSION COEF(3), DUMMY(3)
C
C
C     RETURN IF NO CHANGE FROM PREVIOUS CASE
      IF (LIBSEA+MODSEA+MODWAK .EQ. 0)  RETURN
      CALL TIMER(6)
C     SKIP IF NOT FIRST PASS
      IF (ITHK .NE. 1)  GO TO 40
C     JUMP IF SUB IS INSIDE TCLINE
      IF (ZT(1) .LT. ZS  .AND.  ZS .LT. ZTN)  GO TO 10
      WRITE(6,5)
    5 FORMAT(49H ERROR--WAKE REQUESTED BUT BODY IS OUTSIDE TCLINE)
      CALL ERRXIT
C
C     FIND II1,II2 SUCH THAT ZT(II1) .LT. ZS .LT. ZT(II2)
   10 DO 15  II2=2,NZT
   15 IF (ZT(II2) .GT. ZS)  GO TO 20
C     NEVER FALL THROUGH ABOVE LOOP
   20 II1 = II2-1
      IF (ZT(II1) .GE. ZS)  II1 = II1-1
C     FROUDE NUMBER BASED ON N AVERAGED OVER BODY DIAMETER
C     NOTE DTAVEN IS A FUNCTION FOR AVERAGE N
      FD = 2.*PI*BODSPD/(DTAVEN(BODDIA)*BODDIA)
C     WAKE RADIUS FROM THAT FROUDE NUMBER
      WRAD = .5*CWAKR*BODDIA*FD**.25
C     AVERAGE N OVER WAKE RADIUS JUST COMPUTED
      BV = DTAVEN(2.*WRAD)
C     FROUDE NUMBER
      FD = 2.*PI*BODSPD/(BV*BODDIA)
      WAKRAD = .5*CWAKR*BODDIA*FD**.25
C     NOMINAL START OF WAKE COLLAPSE (DOES NOT INCLUDE SIZING
C     FACTOR CWAKX)
      XWNOM = FD*BODDIA
      AVESQN = BV**2
C
C     COMPUTE INTEGRAL OF AVESQN*(Z+BODDEP)*SIN(ETA*SQRT(WAKRAD**2
C     -(Z+BODDEP)**2))*PSI/ETA*DZ FROM -BODDEP-WAKRAD TO -BODDEP+WAKRAD
C
C     LOOP FOR EACH MODE
   40 DO 190  MODE=1,MODES
C     FWA-1 OF EVECTOR
      LOC1 = (MODE-1)*NZT
C     COS(THETA)**2 = (XI/RK)**2
      C2T = 1./(TEVAL(MODE)*BODSPD**2)
C     SKIP IF SPEED IS SUB-CRITICAL
      TWI(MODE) = 0.
      IF (C2T .GT. 1.)  GO TO 190
C     ETA=RK*SIN(THETA)
      ETA = RK*SQRT(1. -C2T)
C     INTEGRAND, DISPLACEMENT AT SUB DEPTH, INITIAL VALUE OF INTEGRAL
      G = 0.
```

```
      ZETPST = 0.
      SUMI = 0.
C     ARGUMENT OF SIN FUNCTION IN INTEGRAND
      SARG = WAKRAD*ETA
C     DISPLACEMENTS OF TCLINE TABLE POINTS BELOW AND ABOVE SUB DEPTH
      ZET1 = ZS - ZT(II1)
      ZET2 =-ZS + ZT(II2)
C     SET INDICES OF NEXT TCLINE POINTS FROM INITIAL VALUES
      I1 = II1
      I2 = II2
C     INPUT RESOLUTION GIVES MAX ALLOWED INCREMENT IN SARG
      DSARG = PI/RESLVS
C
C     ------START INTEGRATION LOOP--------
C     SAVE PAST VALUE OF INTEGRAND AND TENTATIVELY SET NEW SARG
   50 GPST = G
      SARG = SARG - DSARG
C
      IF (SARG .GT. 0.)  GO TO 60
C     SINE RESOLUTION PERMITS STEP TO END OF INTERVAL (K=0 ALWAYS DOES
C     THIS).  TENTATIVELY SET DISPLACEMENT TO END-OF-INTERVAL
      ZET = WAKRAD
      SARG = 0.
      GO TO 70
C     SET DISPLACEMENT CORRESPONDING TO THIS VALUE OF SARG
   60 ZET = SQRT(WAKRAD**2 - (SARG/ETA)**2)
C
C     HIT ALL POINTS IN TCLINE TABLE
   70 IF (ZET .LT. ZET1)  GO TO 80
C     HIT POINT AT ZET1 UNLESS ZET2 OCCURS FIRST
      IF (ZET2 .LT. ZET1)  GO TO 90
C     ZET1 IS FIRST.  HIT IT AND POINT TO NEXT ENTRY
      ZET = ZET1
      I1 = I1 - 1
C     SET ZET1 FOR THIS NEXT ENTRY. PRESET TO END OF INTERVAL IN CASE
C     WE WENT OFF THE TABLE.
      ZET1 = WAKRAD
      IF (I1 .GT. 0)  ZET1 = ZS - ZT(I1)
C     IF ZET2 MATCHED ZET1, FIX IT TOO
      IF (ZET2 - ZET)  110,100,110
C
C     JUMP IF SINE RESOLUTION IS CONTROLLING CRITERION
   80 IF (ZET .LT. ZET2)  GO TO 120
C     ENTRY AT I2 GIVES SMALLEST STEP.  USE ZET2
   90 ZET = ZET2
C     POINT TO NEXT ENTRY.
  100 I2 = I2 + 1
C     SET CORRESPONDING ZET2 (SET TO WAKE RADIUS IF ABOVE TCLINE)
      ZET2 = WAKRAD
      IF (I2 .LE. NZT)  ZET2 = ZT( I2) - ZS
C     RESET SARG SO IT CORRESPONDS TO THE NEW ZET
  110 SARG = ETA*SQRT(WAKRAD**2 - ZET**2)
C
C     PICK UP Z COORDINATES CORRESPONDING TO DISPLACEMENT ZET
  120 Z1 = ZS - ZET
      Z2 = ZS + ZET
C
C     GET COEFFS FOR DETERMINING EFUNCTION AT Z1
      J1 = I1 + 1
      CALL DTPSIC(Z1, J1, COEF, DUMMY)
C     COMPUTE EFUNCTION AT Z1.  NOTE Z1 NEVER ABOVE TCLINE
      IF (J1 .GE. 0)  GO TO 130
```

```
C     Z1 BELOW TCLINE
      UNZ1 = COEF*EVEC(LOC1+1)
      GO TO 140
C     Z1 WITHIN TCLINE
  130 LOC = LOC1 + J1
      UNZ1 = COEF(1)*EVEC(LOC-1) + COEF(2)*EVEC(LOC)
     1       + COEF(3)*EVEC(LOC+1)
C     GET COEFFS FOR DETERMINING EFUNCTION AT Z2
  140 J2 = I2 -1
      CALL DTPSIC(Z2, J2, COEF, DUMMY)
C     COMPUTE EFUNCTION AT Z2.  NOTE Z2 NEVER BELOW TCLINE
      IF (J2 .NE. 0)  GO TO 150
      UNZ2 = COEF*EVEC(LOC1+NZT)
      GO TO 160
C     Z2 WITHIN TCLINE
  150 LOC = LOC1 + J2
      UNZ2 = COEF(1)*EVEC(LOC-1) + COEF(2)*EVEC(LOC)
     1      + COEF(3)*EVEC(LOC+1)
C
  160 IF (RK .NE. 0.)  GO TO 170
C     INTEGRAND FOR K=0
      G = (UNZ2-UNZ1)*ZET*SQRT(WAKRAD**2-ZET**2)
      GO TO 180
C     INTEGRAND FOR K NON-ZERO
  170 G = (UNZ2 -UNZ1)*ZET*SIN(SARG)/ETA
C
C     ADD CURRENT STEP INTO INTEGRAL (TRAPEZOIDAL)
  180 SUMI = SUMI + .5*(G +GPST)*(ZET -ZETPST)
C     SAVE VALUE FOR NEXT STEP
      ZETPST = ZET
C     JUMP BACK FOR NEXT STEP
      IF (ZET .LT. WAKRAD)  GO TO 50
C
C     ADD WAKE TERM TO DISPERSION TABLES
      TWI(MODE) = AVESQN*SUMI
  190 CONTINUE
      CALL TIMER(-6)
      RETURN
      END
      SUBROUTINE DTWRIT
C     SAVE DISP TABLE, GENERATE NEW DT LIBRARY IF OCEAN CHANGED
C
      COMMON /CONTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
     1,     NODISP, NOGRIC
      EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
     1,     (MODBOD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,     NTDTAB, NTEVEC, NTTEMP
C
      COMMON /OCEAN/ LIBSEA, MODES, NK, TABK(100), NZT, DKRAT, DTDEP
     1,     TDEPMX, TDEP(400), NFLAG, SQBV(400), OCNDEP, RKMAX
     2,     SQN(400), NKT, IPPVEC
C
      COMMON // RK, ITHK, ZT(400), LWAVEC, NZT1, NZT2, ZT1D
     1,     ZTN, ITOP, IBOT, SQK, TOP1, TOP2, TOP3, TOP4, TOP7, TOP8
     2,     BOT1, BOT2, BOT3, BOT4, BOT7, BOT8, IERR, QIC(400)
     3,     NTRDT, Z, ZS, TEVAL(400), TDLDK(80), TPOBS(80)
     4,     TDPOBS(80), TDPSRC(80), TWI(80), TPSUPT(80), TPSUPB(80)
     5,     TDLDK2(80), EVEC(400,80), TEMP(400,5), DC(400)
     6,     TRIMAT(400,3), SD(400), SDL(400), SDL2(400), DL(400), D(400)
     7,     DU(400), CON(400), OMEG(400), BND(400), IEVAL(400)
```

```
     8,      EV3PST(80), PSIN2I(80)
C
C
C     SKIP IF NCT 1ST PASS
      IF (ITHK .GT. 1)  GO TO 5
C     INITIALIZE DISPERSION TABLE FILE
      REWIND NTDTAB
      WRITE(NTDTAB) MODES
C     DISP TAB FOR THIS VALUE OF K
    5 WRITE(NTDTAB) RK,(TEVAL(M),TDLDK(M),TPOBS(M),TDPOBS(M),TDPSRC(M),
     1              TWI(M),TPSUPT(M),TPSUPB(M),TDLDK2(M),M=1,MODES)
C     JUMP IF OCEAN IS SAME AS PREVIOUS CASE
      IF (MODSEA .EQ. 0)  GO TO 10
C     WRITE NEW OCEAN ON CT LIBRARY FILE
      WRITE(NTDLIB) (TEVAL(M),IEVAL(M),TDLDK2(M),M=1,MODES)
   10 RETURN
      END
      SUBROUTINE FTCON
C     FOURIER TRANSFORM AND POTENTIAL SOLUTION CONTROL
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,      RBSEP2, RBSTR, RBLIM
C
      COMMON /CCNTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
     1,      NODISP, NOGRID
      EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
     1,      (MODBOD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,      X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,      IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,      ISPHAS
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,      SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMMON /WAKE/ IWAKE, CWAKR, CWAKX, XWAKE, WAKRAD, XWNOM
     1,      RESLVS, CWAKM
C
C
C      ****SUMMARY OF APPROACH****
C      TOTAL SIGNAL SIGTOT=ST+SP WHERE ST IS THE WAVE-LIKE SIGNAL AND SP
C      IS THE POTENTIAL SOLUTION.  ST IS THE INVERSE FOURIER TRANSFORM
C      OF T.  THIS OPERATION IS DONE IN ROUTINE FTFFT.  SP IS COMPUTED
C      AND ADDED TO ST BY ROUTINE FTPOT.
C      IN GENERAL, T=SUMOVERMODES(TP*EXP(I*XI*X)+TM*EXP(-I*XI*X)) WHERE
C      I=SQRT(-1) AND TP=T1(XI) AND TM=T1(-XI).  HOWEVER IF
C      T1(-XI)=CONJG(T1(XI)) OR T1(-XI)=-CONJG(T1(XI)), THE TOTAL
C      TRANSFORM CAN BE WRITTEN T=SUMOVERMODES(2*REAL(T1*EXP(I*XI*X)))
C      OR T=SUMOVERMODES(2*I*AIMAG(T1*EXP(I*XI*X))).  ROUTINE FTNEWX
C      DOES THIS OPERATION BASED ON T1=SUMOVERSOURCES(TF) WHERE
C      TF IS THE MODE BY MODE TRANSFORM (EXCLUDING X DEPENDENCE) DUE TO
C      A PARTICULAR SOURCE.  THESE TRANSFORMS ARE GENERATED BY ROUTINE
C      FTGENO.  SPECIFICALLY, TF=V*S WHERE V DEPENDS ONLY ON THE
C      VARIABLE (SIGNAL) BEING COMPUTED AND S DEPENDS ONLY ON THE SOURCE
C      MODEL.  V IS COMPUTED IN FTVAR AND S IS COMPUTED IN FTSRC.
C
C
       CALL TIMER(7)
C      SKIP IF NCT 1ST PASS
       IF (ITHOBS .NE. 1)  GO TO 5
C      COMPUTE BODY SOURCE PARAMETERS IF BODY MODEL USED
```

```
      IF (IBODY+IPBODY .NE. 0)  CALL BODY1
C     SUPERSTRUCTURE SOURCE PARAMETERS
      IF (ISUPR+IPSUPR .NE. 0)  CALL SUPR1
C     WAKE SOURCE PARAMETERS.  NOTE 1 STATEMENT SUBROUTINES ARE DUMB.
C     START OF WAKE COLLAPSE=INPUT MULTIPLIER*NOMINAL START (SEE DTWAKE)
      IF (IWAKE .NE. 0)  XWAKE = CWAKX*XWNOM
C     SKIP IF (WAVE-LIKE) TRANSFORM SOLUTION NOT REQUESTED
    5 IF (JFFT .EQ. 0)  GO TO 10
C     READ IN DISPERSION TABLE
      CALL FTDTAB
C     PRINT/PLOT DISPERSION TABLE ON 1ST PASS
      IF (ITHOBS .EQ. 1)  CALL FTDTPP
C     GENERATE TRANSFORMS TF FOR EACH SOURCE
      CALL FTGENO
C     DISPLAY POWER SPECTRAL DENSITIES ON 1ST PASS
      IF (ITHOBS .EQ. 1)  CALL FTPSDS
C     SKIP IF NO SIGNALS ARE TO BE COMPUTED
   10 IF (NX .LE. 0)  GO TO 40
C     LOOP FOR EACH CROSS-CUT OF DATA
      DO 30  ITHX=1,NX
C     DOWN-STREAM COORDINATE
      X = XMIN + DX*FLOAT(ITHX-1)
C     SKIP IF TRANSFORM SOLUTION NOT REQUESTED
      IF (JFFT .EQ. 0)  GO TO 20
C     COMPUTE TOTAL TRANSFORM T AT THIS X
      CALL FTNEWX
C     DO INVERSE TRANSFORM FOR SIGNAL VALUES
      CALL FTFFT
C     ADD IN POTENTIAL SOLUTION IF REQUESTED
   20 IF (JPOT .NE. 0)  CALL FTPOT
C     SEND CROSS-CUT DATA TO OUTPUT PROCESSOR
      CALL FTCUTS
   30 CONTINUE
   40 CALL TIMER(-7)
      RETURN
      END
      SUBROUTINE FTCUTS
C     OUTPUT SIGNAL DATA TO PP PROCESSOR
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,     X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,     IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,     ISPHAS
C
      COMMON /NAME/ NAMES(2,10), DTNAMS(2,9)
C
      COMMON /PPCOM/
     1      LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,     VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,     IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,     IDPP, NV, ISYM
     E,     ENDPP, IBLOKS(1)
C
      COMPLEX VAR
      COMMON // ETA, DETA, IETA, NETA, MODE, MINMOD, MAXMOD, XI
     1,     RK, DLDK, PSIO, DPSIO, DPSIB, WAKI, SUPT, MAXK, LOCDT
     2,     LOCCT1(40), IFWADT, LWADT, IXTRAP, VAR, IVSYM, IBSYM
     3,     IWSYM, ISSYM, LOCDTK, JTRAN
      COMPLEX CFFT, CTEMP1, CFT, CEXD
      COMMON // CFFT(256), CTEMP1(256), CFT(256,40), CEXD(256,40)
      EQUIVALENCE (YSPACE,CFFT)
      DIMENSION YSPACE(1)
```

```
C
C
C     JUMP IF THE GRID HAS MULTIPLE DEPTHS AT CONSTANT X
      IF (NOBS .GT. 1)  GO TO 30
C
C     GRID IS MULTIPLE X AND CONSTANT DEPTH
C     SKIP IF NOT 1ST PASS
      IF (ITHX .GT. 1)  GO TO 10
C     INITIALIZE PP SPECIFICATIONS
      CALL SETID(4HCUTS, 0, 1HX, 1HY, NAMES(1,IVAR))
C     WRITE DATA RECORD FOR THIS X
   10 CALL WRTDAT(1, NY, YSPACE, 1, X)
C     SKIP IF NOT LAST PASS
      IF (ITHX .LT. NX)  GO TO 20
      VMIN = 0.
      VMAX = DY*FLOAT(NY-1)
      CALL WRTID(NY, 0,0)
   20 RETURN
C
C     Y-Z SCAN
   30 IF (ITHOBS .GT. 1)  GO TO 40
C     1ST PASS.  INITIALIZE PP SPECS
      CALL SETID(4HCUTS, 0, 5HDEPTH, 1HY, NAMES(1,IVAR))
C     MAKE FUNCTION POSITIVE TO THE LEFT
      FTODP = -FTODP
C     WRITE DATA RECORD FOR THIS DEPTH
   40 CALL WRTDAT(1, NY, YSPACE, 1, OBSDEP)
C     SKIP IF NOT LAST PASS
      IF (ITHOBS .LT. NOBS)  GO TO 50
      VMIN = 0.
      VMAX = DY*FLOAT(NY-1)
      CALL WRTID(NY, 0,0)
   50 RETURN
      END
      SUBROUTINE FTDISP
C     INTERPOLATE IN DISPERSION TABLES AT GIVEN VALUE OF ETA
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,     RBSEP2, RBSTR, RBLIM
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,     SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMMON /WAKE/ IWAKE, CWAKR, CWAKX, XWAKE, WAKRAD, XWNOM
     1,     RESLVS, CWAKM
C
      COMPLEX VAR
      COMMON // ETA, DETA, IETA, NETA, MODE, MINMOD, MAXMOD, XI
     1,     RK, DLDK, PSIO, DPSIO, DPSIB, WAKI, SUPT, MAXK, LOCDT
     2,     LOCDT1(40), IFWADT, LWADT, IXTRAP, VAR, IVSYM, IBSYM
     3,     IWSYM, ISSYM, LOCDTK, JTRAN
      COMPLEX CFTBOD, CFTSUP, CFTWAK
      COMMON // CFTBOD(256), CFTSUP(256), CFTWAK(256), TABXI(256)
     1,     TKO(40), TK(100), TETA(100,40), TDLDK(100,40), TPSIO(100,40)
     2,     TDPSIO(100,40), TDPSIB(100,40), TWAKI(100,40), TSUPT(100,40)
      EQUIVALENCE (TEMP1,CFTBOD)
      DIMENSION TEMP1(9,1)
C
C
C     FIND PROPER POSITION IN DISP TABLE.  ASSUME ETA ALWAYS INCREASES
      DO 10  I=LOCDT,LWADT
   10 IF (ETA .LE. TETA(I))  GO TO 20
```

```
C     FLAG THAT ARGUMENT EXCEEDS TABLE RANGE
      IXTRAP = 1
      RETURN
C
C     LINEAR INTERP TO FIND K AS FUNCTION OF ETA
   20 C2 = (ETA-TETA(I-1)) / (TETA(I)-TETA(I-1))
      C1 = 1.-C2
C     K INDEX WHICH CORRESPONDS TO I
      J = I-LOCDTK
      TEMP = TK(J-1)
C     IF TETA(I-1)=0, PICK UP K FROM 1ST POINT LIST
      IF (I-1 .EQ. IFWADT)  TEMP = TKO(MODE)
      RK = C1*TEMP + C2*TK(J)
C     LINEAR INTERP TO FIND REMAINING VARIABLES AS FUNCTION OF K
C     D(LAMBDA)/DK WHERE LAMBDA = 1/C**2
      DLDK = C1*TDLDK(I-1) + C2*TDLDK(I)
C     NORMALIZED EIGENFUNCTION PSI AND D(PSI)/DZ AT OBSERVATION DEPTH
      PSIO = C1*TPSIO(I-1) + C2*TPSIO(I)
      DPSIO = C1*TDPSIO(I-1) + C2*TDPSIO(I)
C     D(PSI)/DZ AT BODY DEPTH
      IF (IBODY .NE. 0)  DPSIB = C1*TDPSIB(I-1) + C2*TDPSIB(I)
C     WAKE SOURCE TERM
      IF (IWAKE .NE. 0)  WAKI = C1*TWAKI(I-1) + C2*TWAKI(I)
C     SUPERSTRUCTURE TERM = PSI(BOTTOM OF SUPER) - PSI(TOP OF SUPER)
      IF (ISUPR .NE. 0)  SUPT = C1*TSUPT(I-1) + C2*TSUPT(I)
C     SAVE CURRENT TABLE POSITION FOR NEXT ENTRY
      LOCDT = I
      RETURN
      END
      SUBROUTINE FTDTAB
C     READ DISPERSION TABLE AND PERFORM FINAL ADJUSTMENTS ON IT
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,     RBSEP2, RBSTR, RBLIM
C
      COMMON /CONST/ JDK, JDMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,     JDCKSV, JDMSP, JDEDGE
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,     NTDTAB, NTEVEC, NTTEMP
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,     X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,     IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,     ISPHAS
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,     SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMMON /WAKE/ IWAKE, CWAKR, CWAKX, XWAKE, WAKRAD, XWNOM
     1,     RESLVS, CWAKM
C
      COMPLEX VAR
      COMMON // ETA, DETA, IETA, NETA, MODE, MINMOD, MAXMOD, XI
     1,     RK, DLDK, PSIO, DPSIO, DPSIB, WAKI, SUPT, MAXK, LOCDT
     2,     LOCDT1(40), IFWADT, LWADT, IXTRAP, VAR, IVSYM, IBSYM
     3,     IWSYM, ISSYM, LOCDTK, JTRAN
      COMPLEX CFTBOD, CFTSUP, CFTWAK
      COMMON // CFTBOD(256), CFTSUP(256), CFTWAK(256), TABXI(256)
     1,     TKO(40), TK(100), TETA(100,40), TDLDK(100,40), TPSIO(100,40)
     2,     TDPSIO(100,40), TDPSIB(100,40), TWAKI(100,40), TSUPT(100,40)
      EQUIVALENCE (TEMP1,CFTBOD)
```

```
      DIMENSION TEMP1(9,1)
C
C
      CALL TIMER(8)
      SQUIN = 1./BODSPD**2
C     MODE1 AND MODEN ARE INPUT LIMITS OF DESIRED MODES.  SET UP
C     INTERNAL STORAGE AND DO-LOOP LIMITS TO SQUEEZE OUT UNUSED MODES
      MINMOD = 1
      MAXMOD = MODEN -MODE1 +1
C     CHECK MODE RANGE AGAINST AVAILABLE STORAGE
      IF (MAXMOD .LE. JDMFT)  GO TO 6
      WRITE(6,3) MODE1,MODEN,JDMFT
    3 FORMAT(29H MODE RANGE EXCEEDS DIMENSION,3I10)
      CALL ERRXIT
C     INITIALIZE FWA OF TABLE FOR EACH MODE
    6 DO 10  MODE=MINMOD,MAXMOD
   10 LOCDT1(MODE) = 0
C     DISPERSION TABLE IS ON TAPE NTDTAB
      REWIND NTDTAB
      READ(NTDTAB) MODES
C     SKIP IF DISP TABLE HAS AT LEAST AS MANY MODES AS WANTED
      IF (MODEN .LE. MODES)  GO TO 16
      WRITE(6,13) MODEN,MODES
   13 FORMAT(20H MODEN EXCEEDS MODES,2I10)
      CALL ERRXIT
C     LOOP FOR EACH ENTRY (VALUE OF K) IN DISP TAB, BUT DONT EXCEED
C     STORAGE DIMENSION
   16 DO 50  IK=1,JDK
C           READ  K,(LAMBDA(M),DLDK(M),PSIO(M),DPSIO(M),DPSIB(M),
C              WAKI(M),SUPT(M),SUPB(M),DLDK2(M),M=1,MODES)
      READ(NTDTAB) RK,((TEMP1(I,M),I=1,9),M=1,MODES)
C     SKIP OUT OF LOOP WHEN ENTIRE TABLE HAS BEEN READ
      IF (EOF,NTDTAB)  60,20
C     DATA WAS READ.  SAVE VALUE OF K
   20 TK(IK) = RK
      DO 40  MODE=MINMOD,MAXMOD
C     NOTE THAT HERE (AND EVERYWHERE ELSE IN THE FT ROUTINES), THE
C     VARIABLE -MODE- IS THE STORAGE INDEX OF THE MODE BEING
C     CONSIDERED.   NOW SET ACTUAL MODE NUMBER
      MN = MODE +MODE1 -1
C     TRANSFER VARIABLES FROM TEMP STORAGE TO DISPERSION TABLE
      TDLDK(IK,MODE) = TEMP1(2,MN)
      TPSIO(IK,MODE) = TEMP1(3,MN)
      TDPSIO(IK,MODE) = TEMP1(4,MN)
      TDPSIB(IK,MODE) = 0.
      IF (IBODY .NE. 0)  TDPSIB(IK,MODE) = TEMP1(5,MN)
C     SUPERSTRUCTURE TERM IS PSI(BOTTOM)-PSI(TOP)
      TSUPT(IK,MODE) = 0.
      IF (ISUPR .NE. 0)  TSUPT(IK,MODE) = TEMP1(8,MN) -TEMP1(7,MN)
C     SET UP TO COMPUTE ETA
      TEMP = 1. - SQUIN/TEMP1(1,MN)
      IF (TEMP .GE. 0.)  GO TO 30
C     CANT DO IT.  SET LOC OF LAST ENTRY FOR WHICH ETA IS IMAGINARY
      LOCDT1(MODE) = IK
C     SAVE LAMBDA
      TETA(IK,MODE) = TEMP1(1,MN)
      GO TO 40
   30 ETA = RK*SQRT(TEMP)
      TETA(IK,MODE) = ETA
C     FINISH THE COMPUTATION OF THE WAKE SOURCE TERM AND STORE IT
      TWAKI(IK,MODE) = 0.
      IF (IWAKE .NE. 0)  TWAKI(IK,MODE) = 2.*CWAKM*BODSPD*TEMP1(1,MN)
```

```
     1                                   *TEMP1(6,MN)
   40 CONTINUE
   50 CONTINUE
C
      IK = JDK + 1
C     SET NUMBER OF ENTRIES IN TABLE
   60 MAXK = IK -1
C
C     NOW GO BACK IN THE TABLE AND SET THE ETA=0 VALUES FOR EACH MODE
      MODE = MAXMOD
C     START LOOP FOR EACH MODE
   70 IKO = LOCDT1(MODE)
      IF (IKO .LT. MAXK-1)  GO TO 90
C     TOO FEW TABLE POINTS FOR THIS MODE.  ALSO SKIP LOWER MODES--
C     THEY ARE WORSE CASES
      MINMOD = MODE +1
      IF (MINMOD .LE. MAXMOD)  GO TO 130
      WRITE(6,80)
   80 FORMAT(31H MAX(K) IN DISP TABLE TOO SMALL)
      CALL ERRXIT
C
   90 IF (IKO .GT. 0)  GO TO 100
C     NO IMAGINARY ETA FOR THIS MODE.  SET FWA OF TABLE AND
C     CORRESPONDING K
      LOCDT1(MODE) = 1
      TKO(MODE) = TK(1)
      GO TO 120
C     IKO IS INDEX OF LAST IMAGINARY ETA FOR THIS MODE
  100 IF (TETA(IKO+1,MODE) .NE. 0.)  GO TO 110
C     1ST REAL ETA=0 (BY CHANCE).  POINT TO THAT ENTRY
      LOCDT1(MODE) = IKO +1
      TKO(MODE) = TK(IKO+1)
      GO TO 120
C     AVOID REPEATED INDEXING
  110 T1 = TK(IKO)
      T2 = TK(IKO+1)
C     BACK OUT LAMBDA FROM 1ST REAL ETA
      RL = SQUIN/(1.-(TETA(IKO+1,MODE)/T2)**2)
C     LINEARLY INTERPOLATE TO FIND K(LAMBDA=SQUIN)
C     NOTE THAT TETA(IKO,MODE) WAS USED TO SAVE LAMBDA
      TKO(MODE) = T1+(T2-T1)/(RL-TETA(IKO,MODE))*(SQUIN-TETA(IKO,MODE))
C     LINEAR INTERP COEFFICIENTS FOR POINT AT K=TKO(MODE)
      C2 = (TKO(MODE)-T1) / (T2-T1)
      C1 = 1.-C2
C     REPLACE VALUES AT IKO WITH THE ETA=0 (K=TKO) VALUES
      TDLDK(IKO,MODE) = C1*TDLDK(IKO,MODE) + C2*TDLDK(IKO+1,MODE)
      TPSIO(IKO,MODE) = C1*TPSIO(IKO,MODE) + C2*TPSIO(IKO+1,MODE)
      TDPSIO(IKO,MODE) = C1*TDPSIO(IKO,MODE) + C2*TDPSIO(IKO+1,MODE)
      IF (IBODY .NE. 0)  TDPSIB(IKO,MODE) =
     1                   C1*TDPSIB(IKO,MODE) + C2*TDPSIB(IKO+1,MODE)
      IF (ISUPR .NE. 0)  TSUPT(IKO,MODE) =
     1                   C1*TSUPT(IKO,MODE) + C2*TSUPT(IKO+1,MODE)
C     THERE IS NO VALUE OF WAKE TERM AT IKO SO EXTRAPOLATE FROM
C     POINTS IKO+1 AND IKO+2
      IF (IWAKE .NE. 0)  TWAKI(IKO,MODE) = TWAKI(IKO+1,MODE) +
     1   (TWAKI(IKO+2,MODE)-TWAKI(IKO+1,MODE)) /(TK(IKO+2)-T2)
     2   *(TKO(MODE)-T2)
      TETA(IKO,MODE) = 0.
C
  120 MODE = MODE-1
      IF (MODE .GE. MINMOD)  GO TO 70
C
```

```
 130  CALL TIMER(-8)
      RETURN
      END
      SUBROUTINE FTCTPP
C     PRINT/PLOT DISPERSION TABLE
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,     RBSEP2, RBSTR, RBLIM
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,     X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,     IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,     ISPHAS
C
      COMMON /NAME/ NAMES(2,10), DTNAMS(2,9)
C
      COMMON /PPCOM/
     1      LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,     VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,     ICCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,     IDPP, NV, ISYM
     E,     ENDPP, IBLOKS(1)
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,     SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMMON /WAKE/ IWAKE, CWAKR, CWAKX, XWAKE, WAKRAD, XWNOM
     1,     RESLVS, CWAKM
C
      COMPLEX VAR
      COMMON // ETA, DETA, IETA, NETA, MODE, MINMOD, MAXMOD, XI
     1,     RK, DLDK, PSIO, DPSIO, DPSIB, WAKI, SUPI, MAXK, LOCDT
     2,     LOCDT1(40), IFWADT, LWADT, IXTRAP, VAR, IVSYM, IBSYM
     3,     IWSYM, ISSYM, LOCDTK, JTRAN
      COMPLEX CFTBOD, CFTSUP, CFTWAK
      COMMON // CFTBOD(256), CFTSUP(256), CFTWAK(256), TABXI(256)
     1,     TKO(40), TK(100), TETA(100,40), TDLDK(100,40), TPSIO(100,40)
     2,     TDPSIO(100,40), TDPSIB(100,40), TWAKI(100,40), TSUPT(100,40)
      EQUIVALENCE (TEMP1,CFTBOD)
      DIMENSION TEMP1(9,1)
      DATA DTNAMS/5HDL/DK,1H , 6HW(OBS),1H , 10HDW/DZ(OBS),1H ,
     1           10HDW/DZ(BOD),1H , 5HTWAKE,1H , 5HTSUPR,1H ,
     2           6HLAMBDA,1H , 7HD2L/DK2,1H , 4H-Y/X,1H /
C
C
C     SKIP IF SPECIAL PRINT IS OFF
      IF (IPRDT .EQ. 0)  GO TO 100
C     LOOP FOR EACH MODE IN TABLE
      DO 90  MODE=MINMOD,MAXMOD
C     ACTUAL MODE NUMBER
      MN = MODE +MODE1 -1
      WRITE(6,50) MN
  50  FORMAT(22H1  DISPERSION RELATION/7H   MODE,I3/1H0,6X,1HK,11X,
     1      3HETA,10X,5HDL/DK,8X,6HW(OBS),7X,23HDW/DZ(OBS)   DW/DZ(BOD),
     2      3X,5HTWAKE,8X,5HTSUPR)
C     FWA OF TABLE FOR THIS MODE
      LIM = LOCDT1(MODE)
      WRITE(6,60) LIM,TKO(MODE),TETA(LIM,MODE),TDLDK(LIM,MODE),
     1    TPSIO(LIM,MODE),TDPSIO(LIM,MODE),TDPSIB(LIM,MODE),
     2    TWAKI(LIM,MODE),TSUPT(LIM,MODE)
      LIM = LIM + 1
      WRITE(6,60) (I,TK(I),TETA(I,MODE),TDLDK(I,MODE),TPSIO(I,MODE),
```

```
     1        TDPSIO(I,MODE),TDPSIB(I,MODE),TWAKI(I,MODE),
     2        TSUPT(I,MODE),I=LIM,MAXK)
   60 FORMAT(1X,I3,8E13.5)
   90 CONTINUE
C
C     LOOP FOR EACH D.T. VARIABLE WHICH CAN BE SENT TO PP PROCESSOR
  100 DO 180  ITHVAR=1,6
C     SKIP IF PP OPTION IS OFF FOR THIS VARIABLE
      IF (IPPDT(ITHVAR) .EQ. 0)  GO TO 180
C     SKIP IF DESIRED VARIABLE HAS NOT BEEN COMPUTED
      IF (ITHVAR .EQ. 4  .AND.  IBODY+IPBODY .EQ. 0)  GO TO 180
      IF (ITHVAR .EQ. 5  .AND.  IWAKE .EQ. 0)  GO TO 180
      IF (ITHVAR .EQ. 6  .AND.  ISUPR+IPSUPR .EQ. 0)  GO TO 180
C     PRESET THE PP SPECS
      CALL SETID(DTNAMS(1,ITHVAR), 1, 4HMODE, 3HETA, DTNAMS(1,ITHVAR))
C     INDICATE VARIABLE LIST IS DIFFERENT FOR EACH PARAMETER VALUE
      IVLIST = 3
C     LOOP FOR EACH MODE
      DO 170  MODE=MINMOD,MAXMOD
C     FLOAT ACTUAL MODE NUMBER
      RMODE = MODE + MODE1 -1
C     FWA OF TABLE, NUMBER OF VARIABLE, FUNCTION PAIRS TO BE WRITTEN
      I1 = LOCDT1(MODE)
      N = MAXK -I1 +1
C     JUMP ON VARIABLE TO BE DISPLAYED
      GO TO (110,120,130,140,150,160),ITHVAR
  110 CALL WRTD3(N, TETA(I1,MODE), TDLDK(I1,MODE), 1, RMODE)
      GO TO 170
  120 CALL WRTD3(N, TETA(I1,MODE), TPSIO(I1,MODE), 1, RMODE)
      GO TO 170
  130 CALL WRTD3(N, TETA(I1,MODE), TDPSIO(I1,MODE), 1, RMODE)
      GO TO 170
  140 CALL WRTD3(N, TETA(I1,MODE), TDPSIB(I1,MODE), 1, RMODE)
      GO TO 170
  150 CALL WRTD3(N, TETA(I1,MODE), TWAKI(I1,MODE), 1, RMODE)
      GO TO 170
  160 CALL WRTD3(N, TETA(I1,MODE), TSUPT(I1,MODE), 1, RMODE)
  170 CONTINUE
C     WRITE THE PP ID RECORD
      CALL WRTID(N, 0,0)
  180 CONTINUE
      RETURN
      END
      SUBROUTINE FTFFT
C
      COMMON /CONST/ JDK, JDMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,      JDCKSV, JDMSP, JDEDGE
C
      COMPLEX VAR
      COMMON // ETA, DETA, IETA, NETA, MODE, MINMOD, MAXMOD, XI
     1,      RK, DLDK, PSIO, DPSIO, DPSIB, WAKI, SUPT, MAXK, LOCDT
     2,      LOCDT1(40), IFWADT, LWADT, IXTRAP, VAR, IVSYM, IBSYM
     3,      IWSYM, ISSYM, LOCDTK, JTRAN
      COMPLEX CFFT, CTEMP1, CFT, CEXD
      COMMON // CFFT(256), CTEMP1(256), CFT(256,40), CEXD(256,40)
      EQUIVALENCE (YSPACE,CFFT)
      DIMENSION YSPACE(1)
C     GIVEN COMPLEX FUNCTION CFFT, COMPUTE INVERSE FOURIER
C     TRANSFORM=1/(2*PI) *INTEGRAL(CFFT*EXP(I*ETA*Y)*DETA) WITH
C     LIMITS MINUS TO PLUS INFINITY AND I=SQRT(-1).
C     DISCRETE EQUIVALENT IS
C     RESULT(J)=DETA/(2*PI) *SUMOVERK(CFFT(K)*EXP(I*(J-1)*(K-1)/(2*N)))
```

```
C     WHERE K LIMITS ARE 1 TO 2*N, INDEX J=1,...,2*N AND
C     DETA*DY=2*PI/(2*N).   NOTE EQUIVALENCE IS EXACT WHEN RESULT
C     AND CFFT ARE ALIASED.  TO ALIAS, ASSUME CFFT IS HERMITIAN
C     SYMMETRIC AND UNALIASED CFFT(J)=0 FOR J=N+1,...,2*N
C
C
C     RETURN IF NO SOURCES ARE ON (FFT=0)
      IF (JTRAN .EQ. 0)  RETURN
      CALL TIMER(11)
      COEF = DETA/(2.*PI)
C     HERMITEAN SYMMETRY REQUIRES IM(CFFT(1))=0
      CFFT(1) = CMPLX(COEF*REAL(CFFT(1)), 0.)
C     ALIASING DOES NOT SPECIFY RE(CFFT(N+1)).  MAKE ASSUMPTION
      CFFT(NETA+1) = (0., 0.)
C     SET INDEX FOR ALIASING
      IAL = NETA+NETA
C     JTRAN, 1=INPUT CFFT IS REAL, 2=IMAGINARY, 3=COMPLEX
      GO TO (10,30,50),JTRAN
C     INPUT CFFT IS REAL.  SPECIALIZE THE COMPLEX CASE FOR SPEED
   10 DO 20  IETA=2,NETA
      CFFT(IETA) = CMPLX(COEF*REAL(CFFT(IETA)), 0.)
      CFFT(IAL) = CFFT(IETA)
   20 IAL = IAL-1
C     FLAG TRANSFORM AS BEING REAL
      IFORM = 0
      GO TO 70
C     INPUT CFFT IS IMAGINARY.  SAME AS COMPLEX CASE EXCEPT MULTIPLY
C     BY -SQRT(-1) BECAUSE FOURT OPERATES FASTER ON REAL DATA
   30 DO 40  IETA=2,NETA
      TEMP = COEF*AIMAG(CFFT(IETA))
      CFFT(IETA) = CMPLX(-TEMP, 0.)
      CFFT(IAL) = CMPLX(TEMP, 0.)
   40 IAL = IAL-1
C     FLAG TRANSFORM AS BEING REAL
      IFORM = 0
      GO TO 70
C     INPUT CFFT IS COMPLEX
   50 DO 60  IETA=2,NETA
      CFFT(IETA) = COEF*CFFT(IETA)
      CFFT(IAL) = CONJG(CFFT(IETA))
   60 IAL = IAL-1
C     FLAG TRANSFORM AS BEING COMPLEX
      IFORM = 1
C
   70 CALL FOURT(CFFT, NETA+NETA, 1, 1, IFORM, 0)
C
C     RESULT IS ALWAYS REAL.  AVOID COMPLEX NOTATION
      IF (JTRAN .EQ. 2)  GO TO 90
      DO 80  I=1,NETA
   80 YSPACE(I) = REAL(CFFT(I))
      GO TO 110
C     TRANSFORM WAS MULTIPLIED BY -SQRT(-1) SO TAKE IM(RESULT)
   90 DO 100  I=1,NETA
  100 YSPACE(I) = AIMAG(CFFT(I))
  110 CALL TIMER(-11)
      RETURN
      END
      SUBROUTINE FTGENO
C     GENERATE TRANSFORM TF FOR EACH MODE AND SOURCE
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,      RBSEP2, RBSTR, RBLIM
```

```
C
      COMMON /CONST/ JDK, JDMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,      JDCKSV, JDMSP, JDEDGE
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,      X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,      IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,      ISPHAS
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,      SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMMON /WAKE/ IWAKE, CWAKR, CWAKX, XWAKE, WAKRAD, XWNOM
     1,      RESLVS, CWAKM
C
      COMPLEX VAR
      COMMON // ETA, DETA, IETA, NETA, MODE, MINMOD, MAXMOD, XI
     1,      RK, DLDK, PSIO, DPSIO, DPSIB, WAKI, SUPT, MAXK, LOCDT
     2,      LOCDT1(40), IFWADT, LWADT, IXTRAP, VAR, IVSYM, IBSYM
     3,      IWSYM, ISSYM, LOCDTK, JTRAN
      COMPLEX CFTBOD, CFTSUP, CFTWAK
      COMMON // CFTBOD(256), CFTSUP(256), CFTWAK(256), TABXI(256)
     1,      TKO(40), TK(100), TETA(100,40), TDLDK(100,40), TPSIO(100,40)
     2,      TDPSIO(100,40), TDPSIB(100,40), TWAKI(100,40), TSUPT(100,40)
      EQUIVALENCE (TEMP1,CFTBOD)
      DIMENSION TEMP1(9,1)
C
C
      CALL TIMER(9)
C     NUMBER OF POINTS IN TRANSFORM = NUMBER OF SIGNAL POINTS
      NETA = NY
C     LENGTH OF ALIASED TRANSFORM = 2*NETA.  FIND DETA FROM
C     RELATION  DETA*DY=2*PI/(2*NETA)
      DETA = PI/(FLOAT(NETA)*DY)
C     LOOP FOR EACH MODE
      DO 50  MODE=MINMOD,MAXMOD
C     ADDRESS-1 OF K=0 ENTRY IN DISP TABLE FOR THIS MODE
      LOCDTK = (MODE-1)*JDK
C     ADDRESS OF ETA=0 ENTRY IN DISP TABLE FOR THIS MODE
      IFWADT = LOCDTK + LOCDT1(MODE)
C     INITIALIZE CURRENT POSITION
      LOCDT = IFWADT
C     LWA OF THIS MODE IN DISP TABLE
      LWADT = LOCDTK + MAXK
C     PRESET TO ETA-WITHIN-TABLE.  CAN NOT FALL OFF FRONT, ONLY END
      IXTRAP = 0
C
C     LOOP FOR EACH ETA
      DO 30  IETA=2,NETA
      ETA = DETA*FLOAT(IETA-1)
C     INTERPOLATE IN DISPERSION TABLES USING ETA AS ARGUMENT
      CALL FTDISP
C     SKIP IF WITHIN TABLE
      IF (IXTRAP .EQ. 0)  GO TO 20
C     FELL OFF END.  ZERO OUT REMAINDER OF TRANSFORM
      DO 10  I=IETA,NETA
      TABXI(I) = 0.
      CFTBOD(I) = (0.,0.)
      CFTSUP(I) = (0.,0.)
   10 CFTWAK(I) = (0.,0.)
      GO TO 40
   20 XI = SQRT(RK**2 -ETA**2)
```

```
      TABXI(IETA) = XI
C     COMPUTE V WHICH DEPENDS ONLY ON THE OUTPUT SIGNAL VARIABLE
      CALL FTVAR
C     INCLUDE SOURCE FACTOR(S) AND STORE
      CALL FTSRC
   30 CONTINUE
C
C     EQUATIONS BLOW FOR ETA=0.  EXTRAPOLATE FOR THAT POINT
   40 IF (IBODY .NE. 0)  CFTBOD(1) = 2.*CFTBOD(2) - CFTBOD(3)
      IF (IWAKE .NE. 0)  CFTWAK(1) = 2.*CFTWAK(2) - CFTWAK(3)
      IF (ISUPR .NE. 0)  CFTSUP(1) = 2.*CFTSUP(2) - CFTSUP(3)
C     WRITE TRANSFORMS ON FILE
   50 CALL FTWRIT
      CALL TIMER(-9)
      RETURN
      END
      SUBROUTINE FTNEWX
C     GENERATE TRANSFORM FOR CURRENT VALUE OF X
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,      RBSEP2, RBSTR, RBLIM
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,      NTDTAB, NTEVEC, NTTEMP
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,      X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,      IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,      ISPHAS
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,      SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMMON /WAKE/ IWAKE, CWAKR, CWAKX, XWAKE, WAKRAD, XWNOM
     1,      RESLVS, CWAKM
C
      COMPLEX VAR
      COMMON // ETA, DETA, IETA, NETA, MODE, MINMOD, MAXMOD, XI
     1,      RK, DLDK, PSIO, DPSIO, DPSIB, WAKI, SUPT, MAXK, LOCDT
     2,      LOCDT1(40), IFWADT, LWADT, IXTRAP, VAR, IVSYM, IBSYM
     3,      IWSYM, ISSYM, LOCDTK, JTRAN
      COMPLEX CFFT, CTEMP1, CFT, CEXD
      COMMON // CFFT(256), CTEMP1(256), CFT(256,40), CEXD(256,40)
      EQUIVALENCE (YSPACE,CFFT)
      DIMENSION YSPACE(1)
      DIMENSION ID(4), ISYM(4), ISTAT(4)
C
C
      CALL TIMER(10)
C     IS-- 1=XI, 2=BODY, 3=WAKE, 4=SUPERSTRUCTURE
C     ISTAT(IS)-- -1=TURN ON SOURCE IS, 0=IS IS OFF, 1=IS ALREADY ON
C     ON 1ST PASS, INITIALIZE ALL SOURCES TO OFF
      IF (ITHX .NE. 1)  GO TO 20
      DO 10  IS=1,4
   10 ISTAT(IS) = 0
C     SHOW ALL SOURCES ARE OFF
      JTRAN = 0
      DO 15  MODE=MINMOD,MAXMOD
      DO 15  IETA=1,NETA
   15 CFT(IETA,MODE) = (0., 0.)
C     DECIDE WHICH SOURCES TO TURN ON
C     IF BODY REQUESTED, TURN IT ON THE 1ST PASS (IS=2 FOR BODY)
```

```
   20 IF (IBODY .NE. 0  .AND.  ITHX .EQ. 1)  ISTAT(2) = -1
C     IF SUPERSTRUCT REQUESTED, TURN IT ON THE 1ST PASS (IS=4 FOR SUP)
      IF (ISUPR .NE. 0  .AND.  ITHX .EQ. 1)  ISTAT(4) = -1
C     IF WAKE REQUESTED, TURN IT ON WHEN X EXCEEDS XWAKE (BUT DONT
C     TURN IT ON IF IT IS ALREADY ON)  (IS=3 FOR WAKE)
      IF (IWAKE .NE. 0  .AND.  X .GE. XWAKE  .AND.  ISTAT(3) .NE. 1)
     1                                             ISTAT(3) = -1
C
C     SKIP IF ALL SOURCES ARE OFF
      IF (JTRAN .EQ. 0)  GO TO 70
C     STEP (THE SOURCES WHICH ARE ALREADY ON) THROUGH DX
      DO 50  MODE=MINMOD,MAXMOD
      DO 50  IETA=1,NETA
   50 CFT(IETA,MODE) = CFT(IETA,MODE)*CEXD(IETA,MODE)
C
C     LOOP FOR EACH SOURCE
   70 DO 80  IS=2,4
C     JUMP IF SOURCE IS TO BE TURNED ON NOW
   80 IF (ISTAT(IS) .EQ. -1)  GO TO 90
C     NO SOURCE IS TO BE TURNED ON NOW
      GO TO 210
C
C     READ IN TRANSFORMS FOR EACH SOURCE
   90 REWIND NTTEMP
C     IS=ID(RECNO) INDICATES CONTENTS OF RECORD NUMBER=RECNO
C     ISYM(RECNO) INDICATES SYMMETRY OF TRANSFORM,
C     1=HERMITEAN ANTI-SYMMETRIC, 0=HS, -1=NO SYMMETRY
      READ(NTTEMP) ID,ISYM
      DO 190  MODE=MINMOD,MAXMOD
      DO 95  IETA=1,NETA
   95 CTEMP1(IETA) = (0., 0.)
C     LOOP FOR EACH SOURCE AND XI
      DO 180  IREC=1,4
C     JUMP IF GOING TO READ A SOURCE
      IF (ID(IREC) .NE. 1)  GO TO  150
C     READ XI
      READ(NTTEMP) (YSPACE(IETA),IETA=1,NETA)
C     INSERT (TRANSFORM(S) JUST TURNED ON) INTO EXISTING TRANSFORMS
      DO 100  IETA=1,NETA
      ARG = YSPACE(IETA)*X
  100 CFT(IETA,MODE) = CFT(IETA,MODE)
     1                  + CTEMP1(IETA)*CMPLX(COS(ARG), SIN(ARG))
C     SKIP IF COS, SIN(XI*DX) HAVE ALREADY BEEN STORED
      IF (ISTAT(1) .EQ. 1)  GO TO 140
      DO 130  IETA=1,NETA
      ARG = YSPACE(IETA)*DX
  130 CEXD(IETA,MODE) = CMPLX(COS(ARG), SIN(ARG))
C     XI RECORD SIGNALS END OF SOURCE RECORDS FOR THIS MODE
  140 GO TO 190
C     READ SOURCE RECORD
  150 READ(NTTEMP) (CFFT(IETA),IETA=1,NETA)
C     SKIP IF NOT TURNING ON THIS SOURCE--GET SOURCE NUMBER, TEST STATUS
      IS = ID(IREC)
      IF (ISTAT(IS) .NE. -1)  GO TO 180
C     JTRAN SHOWS CHARACTER OF NET TRANSFORM
C     1=REAL, 2=IMAGINARY, 3=COMPLEX (3 NOT IMPLEMENTED)
      IF (ISYM(IREC) .EQ. 1)  JTRAN = JTRAN .OR. 2
      IF (ISYM(IREC) .EQ. 0)  JTRAN = JTRAN .OR. 1
      IF (ISYM(IREC) .EQ. -1)  JTRAN = JTRAN .OR. 3
C     ADD THIS SOURCE INTO SUM OF EMERGING SOURCES
      DO 160  IETA=1,NETA
  160 CTEMP1(IETA) = CTEMP1(IETA) + CFFT(IETA)
```

```
  180 CONTINUE
  190 CONTINUE
C
C     RESET SOURCE STATUS FROM (TURN ON) TO (ON)
      DO 200  IS=1,4
  200 IF (ISTAT(IS) .EQ. -1)  ISTAT(IS) = 1
C     SHOW EXP(I*XI*DX) HAS BEEN STORED
      ISTAT(1) = 1
C
C
C     SKIP IF ALL SOURCES ARE OFF
  210 IF (JTRAN .EC. 0)  GO TO 300
C     TEST FOR TRANSFORM REAL, IMAGINARY, OR COMPLEX
      GO TO (220,250),JTRAN
C     TRANSFORM IS REAL
  220 DO 240  IETA=1,NETA
      TEMP = 0.
      DO 230  MODE=MINMOD,MAXMOD
  230 TEMP = TEMP + REAL(CFT(IETA,MODE))
  240 CFFT(IETA) = CMPLX(2.*TEMP, 0.)
      GO TO 320
C     TRANSFORM IS IMAGINARY
  250 DO 270  IETA=1,NETA
      TEMP = 0.
      DO 260  MODE=MINMOD,MAXMOD
  260 TEMP = TEMP + AIMAG(CFT(IETA,MODE))
  270 CFFT(IETA) = CMPLX(0., 2.*TEMP)
      GO TO 320
C
  300 DO 310  IETA=1,NETA
  310 CFFT(IETA) = (0., 0.)
C
  320 CALL TIMER(-10)
      RETURN
      END
      SUBROUTINE FTPOT
C     ADD POTENTIAL SOLUTION TO DISTURBANCE
C
      COMMON /CONST/ JDK, JDMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,      JDCKSV, JDMSP, JDEDGE
C
      COMMON /CONTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
     1,      NODISP, NOGRID
      EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
     1,      (MODBOD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,      X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,      IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,      ISPHAS
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,      RBSEP2, RBSTR, RBLIM
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,      SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMPLEX VAR
      COMMON // ETA, DETA, IETA, NETA, MODE, MINMOD, MAXMOD, XI
     1,      RK, CLCK, PSIO, DPSIO, DPSIB, WAKI, SUPT, MAXK, LOCDT
     2,      LOCDT1(40), IFWADT, LWADT, IXTRAP, VAR, IVSYM, IBSYM
     3,      IWSYM, ISSYM, LOCDTK, JTRAN
```

```
      COMPLEX CFFT, CTEMP1, CFT, CEXD
      COMMON // CFFT(256), CTEMP1(256), CFT(256,40), CEXD(256,40)
      EQUIVALENCE (YSPACE,CFFT)
      DIMENSION YSPACE(1)
      DIMENSION ZDI(4)
C
C
      CALL TIMER(12)
      IF (JFFT .NE. 0)   GO TO 6
C     NO WAVES--JUST POTENTIAL SOLN.  INITIALIZE OUTPUT ARRAY.
      DO 4  I=1,NY
    4 YSPACE(I) = 0.
C     SKIP IF OUTSIDE SPECIFIED RANGE FOR POTENTIAL SOLN
    6 IF (X .GT. XPMAX)    GO TO 530
C     INDEX OF MAX Y COORDINATE
      LIM = MIN0(IFIX(YPMAX/DY)+1, NY)
C     SKIP IF BODY POTENTIAL OPTION OFF
      IF (IPBODY .EQ. 0)   GO TO 290
C     X COORDINATE OF OBSERVATION POINT WRT SOURCE, SINK
      X1 = X + RBSEP2
      X2 = X - RBSEP2
      X1S = X1**2
      X2S = X2**2
C     UBIQUITOUS FACTORS
      CON = RBSTR/(4.*PI)
      CONU = CON/BODSPD
C     Z COORDINATE OF OBS POINT WRT BODY, IMAGE
      ZDI(1) = -OBSDEP + BODDEP
      ZDI(2) = -OBSDEP - BODDEP
C     LOOP FOR BODY, IMAGE
      DO 210  ID=1,2
C     PICK UP RELATIVE Z COORDINATE
      ZD = ZDI(ID)
      ZDS = ZD**2
C     PRESET Y COORDINATE
      Y = 0.
C     LOOP FOR EACH Y
      DO 200  I=1,LIM
      YS = Y**2
C     SQUARE OF TRANSVERSE DISTANCE TO OBS POINT
      TS = ZDS + YS
C     DISTANCE TO OBS POINT FROM SOURCE, SINK
      R1 = SQRT(X1S+TS)
      R2 = SQRT(X2S+TS)
C
C     JUMP ON VARIABLE TO COMPUTE EFFECT OF SOURCE+SINK AT THIS DEPTH
      GO TO (10,20,30,40,50,60,70,80,90,100),IVAR
C     (U) X-VELOCITY
   10 YSPACE(I) = YSPACE(I) + CON*(X1/R1**3-X2/R2**3)
      GO TO 200
C     (V) Y-VELOCITY
   20 YSPACE(I) = YSPACE(I) + CON*(Y/R1**3-Y/R2**3)
      GO TO 200
C     (DELTA-X) X-DISPLACEMENT
   30 YSPACE(I) = YSPACE(I) - CONU*(1./R1-1./R2)
      GO TO 200
C     (DELTA-Y) Y-DISPLACEMENT
C     CONTRIBUTION IS ZERO FOR TS=0.
   40 IF (TS .EQ. 0.)  GO TO 200
      YSPACE(I) = YSPACE(I) + CONU*Y/TS*(X1/R1-X2/R2)
      GO TO 200
C     (DELTA-Z) VERTICAL DISPLACEMENT
```

```
C       CONTRIBUTION IS ZERO FOR TS=0.
    50 IF (TS .EQ. 0.)  GO TO 200
       YSPACE(I) = YSPACE(I) + CONU*ZD/TS*(X1/R1-X2/R2)
       GO TO 200
C      (EPSILON-X)  X-STRAIN
    60 YSPACE(I) = YSPACE(I) + CONU*(X1/R1**3-X2/R2**3)
       GO TO 200
C      (EPSILON-Y) Y-STRAIN
    70 IF (TS .EQ. 0.)  GO TO 75
       TEMP = 1. -2.*YS/TS
       YSPACE(I) = YSPACE(I) + CONU*(X1/R1*(TEMP-YS/R1**2)
     1                 -X2/R2*(TEMP-YS/R2**2))/TS
       GO TO 200
C      LIMIT OF ABOVE AS TS GOES TO 0.
    75 YSPACE(I) = YSPACE(I) + CONU*(-.5/X1S+.5/X2S)
       GO TO 200
C      (GAMMA-XY)
    80 YSPACE(I) = YSPACE(I) + 2.*CONU*(Y/R1**3-Y/R2**3)
       GO TO 200
C      (SIGMA) STRAIN RATE
    90 YSPACE(I) = YSPACE(I) - CON*((1.-3.*ZDS/R1**2)/R1**3
     1                 -(1.-3.*ZDS/R2**2)/R2**3)
       GO TO 200
C      (W) VERTICAL VELOCITY
   100 YSPACE(I) = YSPACE(I) + CON*(ZD/R1**3-ZD/R2**3)
   200 Y = Y + DY
   210 CONTINUE
C
C
C      SKIP IF SUPERSTRUCTURE NOT TO BE COMPUTED
   290 IF (IPSUPR .EQ. 0)  GO TO 530
C      X COORDINATE OF OBS POINT WRT SOURCE, SINK
       X1 = X - SUPMID + SUSEP2
       X2 = X - SUPMID - SUSEP2
       X1S = X1**2
       X2S = X2**2
C      UBIQUITOUS FACTORS
       CON = SUSTR/(4.*PI)
       CONU = CON/BODSPD
C      Z COORDINATE OF OBS POINT WRT BOTTOM, BOTTOM IMAGE, TOP,
C      TOP IMAGE OF SUPERSTRUCTURE
       ZDI(1) = -OBSDEP + BODDEP - SUPBOT
       ZDI(2) = -OBSDEP - BODDEP + SUPBOT
       ZDI(3) = -OBSDEP + BODDEP - SUPTOP
       ZDI(4) = -OBSDEP - BODDEP + SUPTOP
C      LOOP FOR BOTTOM, THEN TOP OF SUPER
       ID = 1
       DO 520  IBT=1,2
C      LOOP FOR SUPER, IMAGE
       DO 510  ISI=1,2
C      PICK UP RELATIVE Z COORDINATE
       ZD = ZDI(ID)
       ZDS = ZD**2
C      PRESET
       Y = 0.
C      LOOP FOR EACH Y
       DO 500  I=1,LIM
       YS = Y**2
C      SQUARE OF TRANSVERSE DISTANCE TO OBS POINT
       TS = ZDS + YS
C      DISTANCE TO OBS POINT FROM SOURCE, SINK
       R1 = SQRT(X1S+TS)
```

```
      R2 = SQRT(X2S+TS)
C     JUMP ON VARIABLE TO COMPUTE EFFECT OF SOURCE+SINK AT THIS DEPTH
      GO TO (310,320,330,340,350,360,370,380,390,400),IVAR
C     (U)
  310 YSPACE(I) = YSPACE(I) - CON*(X1/R1/(ZD+R1)-X2/R2/(ZD+R2))
      GO TO 500
C     (V)
  320 YSPACE(I) = YSPACE(I) - CON*(Y/R1/(ZD+R1)-Y/R2/(ZD+R2))
      GO TO 500
C     (DELTA-X)
  330 YSPACE(I) = YSPACE(I) - CONU*ALOG((ZD+R1)/(ZD+R2))
      GO TO 500
C     (DELTA-Y)
C     CONTRIBUTION IS ZERO FOR Y=0.
  340 IF (Y .EQ. 0.)  GO TO 500
      TEMP = SQRT(TS)
      YSPACE(I) = YSPACE(I) - CONU*(ASIN((TS+ZD*R1)/TEMP/(ZD+R1))
     1                -ASIN((TS+ZD*R2)/TEMP/(ZD+R2)))
      GO TO 500
C     (DELTA-Z)
  350 YSPACE(I) = YSPACE(I) - CONU*ALOG((X1+R1)/(X2+R2))
      GO TO 500
C     (EPSILON-X)
  360 YSPACE(I) = YSPACE(I) - CONU*(X1/R1/(ZD+R1)-X2/R2/(ZD+R2))
      GO TO 500
C     (EPSILON-Y)
C     CONTRIBUTION IS ZERO FOR TS=0
  370 IF (TS .EQ. 0.)  GO TO 500
      YSPACE(I) = YSPACE(I) - CONU*(X1/R1*(-1./(ZD+R1)+ZD/TS)
     1                -X2/R2*(-1./(ZD+R2)+ZD/TS))
      GO TO 500
C     (GAMMA-XY)
  380 YSPACE(I) = YSPACE(I) - 2.*CONU*(Y/R1/(ZD+R1)-Y/R2/(ZD+R2))
      GO TO 500
C     (SIGMA)
  390 YSPACE(I) = YSPACE(I) - CON*(ZD/R1**3-ZD/R2**3)
      GO TO 500
C     (W)
  400 YSPACE(I) = YSPACE(I) - CON*(1./R1-1./R2)
  500 Y = Y + DY
  510 ID = ID + 1
      CON = -CON
      CONU = -CONU
  520 CONTINUE
  530 CALL TIMER(-12)
      RETURN
      END
      SUBROUTINE FTPSDS
C     OUTPUT PSD DATA TO PP PROCESSOR
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,     X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,     IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,     ISPHAS
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPIO, NTPDAT, NTPLOT
     1,     NTDTAB, NTEVEC, NTTEMP
C
      COMMON /NAME/ NAMES(2,10), DTNAMS(2,9)
C
      COMMON /PPCOM/
     1      LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
```

```
     2,     VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,     IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,     IDPP, NV, ISYM
     5,     ENDPP, IBLOKS(1)
C
      COMPLEX VAR
      COMMON // ETA, DETA, IETA, NETA, MODE, MINMOD, MAXMOD, XI
     1,     RK, DLDK, PSIO, DPSIO, DPSIB, WAKI, SUPI, MAXK, LOCDT
     2,     LOCDT1(40), IFWADT, LWADT, IXTRAP, VAR, IVSYM, IBSYM
     3,     IWSYM, ISSYM, LOCDTK, JTRAN
      COMPLEX CFTBOD, CFTSUP, CFTWAK
      COMMON // CFTBOD(256), CFTSUP(256), CFTWAK(256), TABXI(256)
     1,     TKO(40), TK(100), TETA(100,40), TDLDK(100,40), TPSIO(100,40)
     2,     TDPSIO(100,40), TDPSIB(100,40), TWAKI(100,40), TSUPT(100,40)
      EQUIVALENCE (TEMP1,CFTBOD)
      DIMENSION TEMP1(9,1)
      REAL NAMES
      DIMENSION NAMPSD(2,3), IDPSD(3), ID(4), JSYM(4)
      DATA NAMPSD/9HPSD(BODY),1H ,9HPSD(WAKE),1H ,10HPSD(SUPER),1H /
      DATA IDPSD/4HBPSD, 4HWPSD, 4HSPSD/
C
C
C     RETURN IF PSDS ARE NOT TO BE DISPLAYED
      IF (IPPPSD .EQ. 0)  RETURN
C     RECORD NUMBER OF SOURCE TO PP NEXT
      IPPREC = 1
   10 REWIND NTTEMP
      READ(NTTEMP) ID,JSYM
C     ID(I) = SOURCE NUMBER IS OF RECORD I
C     IS-- 1=XI (WHICH IS LAST RECORD OF MODE), 2=BODY, 3=WAKE, 4=SUPER
      IS = ID(IPPREC)
C     INITIALIZE PP SPECS FOR SOURCE IS
      CALL SETID(IDPSD(IS-1), 1, 4HMODE, 3HETA, NAMPSD(1,IS-1))
C     LOOP FOR EACH MODE
      DO 50  MODE=MINMOD,MAXMOD
      ITHREC = 1
C     READ RECORD NUMBER ITHREC FOR THIS MODE
C     USE CFTBOD AS TEMP STORAGE
   20 READ(NTTEMP) (CFTBOD(IETA),IETA=1,NETA)
C     SKIP IF THIS IS NOT THE SOURCE WE WANT
      IF (ITHREC .NE. IPPREC)  GO TO 40
C
C     (SEE COMMENTS IN FTCON).  ASSUME HERE THAT TP=+OR-TM.  THEN
C     ONLY TP WAS WRITTEN ON THE FILE.   PSD=(2*TP)**2
      DO 30  IETA=1,NETA
   30 TEMP1(IETA) = 4.*(REAL(CFTBOD(IETA))**2 + AIMAG(CFTBOD(IETA))**2)
C     COMPUTE AND FLOAT ACTUAL MODE NUMBER
      RMODE = MODE + MODE1 -1
C     WRITE THE PSD DATA FOR THE PP PROCESSOR
      CALL WRTDAT(1, NETA, TEMP1, 1, RMODE)
C
C     BUMP RECORD NUMBER AND LOOP BACK FOR NEXT SOURCE
   40 ITHREC = ITHREC + 1
      IF (ID(ITHREC) .NE. 1)  GO TO 20
C     LAST RECORD OF EACH MODE IS XI
      READ(NTTEMP) (TEMP1(IETA),IETA=1,NETA)
   50 CONTINUE
C     WRAP UP PP SPECS AND WRITE THEM
      VMIN = 0.
      VMAX = DETA*FLOAT(NETA-1)
C     NOTE THAT NAMES HAS BEEN MADE REAL
      TITLE(1) = NAMES(1,IVAR)
```

```
      TITLE(2) = NAMES(2,IVAR)
      CALL WRTID(NETA, 0,0)
C     MOVE UP TO NEXT RECORD, LOOP IF IT IS ANOTHER SOURCE
      IPPREC = IPPREC + 1
      IF (ID(IPPREC) .NE. 1)  GO TO 10
      RETURN
      END
      SUBROUTINE FTSRC
C     COMPUTE TRANSFORM POINT FOR CURRENT ETA, MODE
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,      RBSEP2, RBSTR, RBLIM
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,      SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMMON /WAKE/ IWAKE, CWAKR, CWAKX, XWAKE, WAKRAD, XWNUM
     1,      RESLVS, CWAKM
C
      COMPLEX VAR
      COMMON // ETA, DETA, IETA, NETA, MODE, MINMOD, MAXMOD, XI
     1,      RK, DLDK, PSIO, DPSIO, DPSIB, WAKI, SUPT, MAXK, LOCDT
     2,      LOCDT1(40), IFWADT, LWADT, IXTRAP, VAR, IVSYM, IBSYM
     3,      IWSYM, ISSYM, LOCDTK, JTRAN
      COMPLEX CFTBOD, CFTSUP, CFTWAK
      COMMON // CFTBOD(256), CFTSUP(256), CFTWAK(256), TABXI(256)
     1,      TKO(40), TK(100), TETA(100,40), TDLDK(100,40), TPSIO(100,40)
     2,      TDPSIO(100,40), TDPSIB(100,40), TWAKI(100,40), TSUPT(100,40)
      EQUIVALENCE (TEMP1,CFTBOD)
      DIMENSION TEMP1(9,1)
      COMPLEX S
C
C
C     NOTE THAT S=INTEGRAL(F*PSI*DZ).  IBSYM,IWSYM,ISSYM INDICATE THE
C     SYMMETRY OF T1 (SEE FTCON COMMENTS) FOR THE BODY, WAKE, SUPER WHEN
C     XI CHANGES SIGN-- 1=HERMITEAN ANTI-SYMMETRIC, 0=HERMITEAN
C     SYMMETRIC, -1=NO SYMMETRY
C
C     SKIP IF BODY OFF
      IF (IBODY .EQ. 0)  GO TO 100
      IF (IBODY .EQ. 2)  GO TO 20
C     RANKINE BODY
C     RBSTR=SOURCE STRENGTH, RBSEP2=1/2 SOURCE TO SINK SEPARATION
      S = CMPLX(0., -2.*RBSTR*DPSIB*SIN(XI*RBSEP2))
      CFTBOD(IETA) = S*VAR
      IBSYM = IVSYM
      GO TO 100
C     DIPOLE BODY.  RBLIM=LIM(RBSTR*RBSEP2)
   20 S = CMPLX(0., -2.*RBLIM*DPSIB*XI)
      CFTBOD(IETA) = S*VAR
      IBSYM = IVSYM
C
C     SKIP IF WAKE IS OFF
  100 IF (IWAKE .EQ. 0)  GO TO 200
      S = CMPLX(-WAKI*COS(XI*XWAKE), WAKI*SIN(XI*XWAKE))
      CFTWAK(IETA) = S*VAR
      IWSYM = IVSYM
C
C     SKIP IF SUPERSTRUCTURE IS OFF
  200 IF (ISUPR .EQ. 0)  GO TO 300
      IF (ISUPR .EQ. 2)  GO TO 220
C     OVAL SUPERSTRUCTURE.  SUSTR=SOURCE STRENGTH, SUSEP2=1/2 SOURCE
```

```
C     TO SINK SEPARATION, SUPT=PSI(BOT)-PSI(TOP)
      TEMP = 2.*SUSTR*SUPT*SIN(XI*SUSEP2)
C     SUPMID=X COORDINATE OF MIDDLE OF SUPERSTRUCTURE
      S = CMPLX(TEMP*SIN(XI*SUPMID), TEMP*COS(XI*SUPMID))
      CFTSUP(IETA) = S*VAR
      ISSYM = IVSYM
      GO TO 300
C     CIRCULAR SUPER.  SULIM=LIM(SUSTR*SUSEP2)
  220 TEMP = 2.*SULIM*SUPT*XI
      S = CMPLX(TEMP*SIN(XI*SUPMID), TEMP*COS(XI*SUPMID))
      CFTSUP(IETA) = S*VAR
      ISSYM = IVSYM
C
  300 RETURN
      END
      SUBROUTINE FTVAR
C     COMPUTE THE VARIABLE-DEPENDENT (BUT SOURCE-INDEPENDENT) PART OF
C     THE TRANSFORM
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,     RBSEP2, RBSTR, RBLIM
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,     X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,     IVAR, IPROT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,     ISPHAS
C
      COMMON /NAME/ NAMES(2,10), DTNAMS(2,9)
C
      COMPLEX VAR
      COMMON // ETA, DETA, IETA, NETA, MODE, MINMOD, MAXMOD, XI
     1,     RK, DLDK, PSIO, DPSIO, DPSIB, WAKI, SUPT, MAXK, LOCDT
     2,     LOCDT1(40), IFWADT, LWADT, IXTRAP, VAR, IVSYM, IBSYM
     3,     IWSYM, ISSYM, LOCDTK, JTRAN
      DATA NAMES/  20HX-VELOCITY (U)      , 20HY-VELOCITY (V)
     1,            20HX-DISPLACE (DELTA-X), 20HY-DISPLACE (DELTA-Y)
     2,            20HZ-DISPLACE (DELTA-Z), 20HX-STRAIN (EPSILON-X)
     3,            20HY-STRAIN (EPSILON-Y), 20HSHEAR STRN (GAMMAXY)
     4,            20HDILATATION (SIGMA)  , 20HZ-VELOCITY (W)       /
C
C
C     IVSYM, 1=VAR IS HERMITEAN ANTI-SYMMETRIC IN XI, 0=H. SYMMETRIC,
C     -1=NO SYMMETRY IN XI (THIS CASE HAS NOT BEEN IMPLEMENTED)
      TEMP = 2.*XI*(2.*RK/DLDK*(ETA/(BODSPD*XI**2))**2 +1.)
      TEMP = 1./TEMP
      GO TO (10,20,30,40,50,60,70,80,90,100),IVAR
C
C     (U) DOWN TRACK VELOCITY DISTURBANCE
   10 VAR = CMPLX(TEMP*DPSIO*XI/RK**2, 0.)
      IVSYM = 0
      GO TO 200
C
C     (V) CROSS TRACK VELOCITY
   20 VAR = CMPLX(TEMP*DPSIO*ETA/RK**2, 0.)
      IVSYM = 1
      GO TO 200
C
C     (DELTA-X) DOWN TRACK DISPLACEMENT
   30 VAR = CMPLX(0., -TEMP*DPSIO/(BODSPD*RK**2))
      IVSYM = 0
      GO TO 200
C
```

```
C     (DELTA-Y) CROSS TRACK DISPLACEMENT
   40 VAR = CMPLX(0., -TEMP*DPSIO*ETA/(BODSPD*XI*RK**2))
      IVSYM = 1
      GO TO 200
C
C     (DELTA-Z) VERTICAL DISPLACEMENT
   50 VAR = CMPLX(-TEMP*PSIO/(BODSPD*XI), 0.)
      IVSYM = 0
      GO TO 200
C
C     (EPSILON-X) DOWN TRACK STRAIN
   60 VAR = CMPLX(TEMP*DPSIO*XI/(BODSPD*RK**2), 0.)
      IVSYM = 0
      GO TO 200
C
C     (EPSILON-Y) CROSS TRACK STRAIN
   70 VAR = CMPLX(TEMP*DPSIO*ETA**2/(BODSPD*XI*RK**2), 0.)
      IVSYM = 0
      GO TO 200
C
C     (GAMMA-XY) SHEARING STRAIN IN HORIZONTAL PLANE
   80 VAR = CMPLX(TEMP*DPSIO*2.*ETA/(BODSPD*RK**2), 0.)
      IVSYM = 1
      GO TO 200
C
C     (SIGMA) HORIZONTAL PLANE DILATATION
   90 VAR = CMPLX(0., TEMP*DPSIO)
      IVSYM = 0
      GO TO 200
C
C     (W) VERTICAL VELOCITY
  100 VAR = CMPLX(0., -TEMP*PSIO)
      IVSYM = 0
C
C
  200 RETURN
      END
      SUBROUTINE FTWRIT
C     WRITE TRANSFORMS FOR EACH SOURCE ON FILE NTTEMP
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,     RBSEP2, RBSTR, RBLIM
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,     NTDTAB, NTEVEC, NTTEMP
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,     SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMMON /WAKE/ IWAKE, CWAKR, CWAKX, XWAKE, WAKRAD, XWNOM
     1,     RESLVS, CWAKM
C
      COMPLEX VAR
      COMMON // ETA, DETA, IETA, NETA, MODE, MINMOD, MAXMOD, XI
     1,     RK, DLDK, PSIO, DPSIO, DPSIB, WAKI, SUPT, MAXK, LOCDT
     2,     LOCDT1(40), IFWADT, LWADT, IXTRAP, VAR, IVSYM, IBSYM
     3,     IWSYM, ISSYM, LOCDTK, JTRAN
      COMPLEX CFTBOD, CFTSUP, CFTWAK
      COMMON // CFTBOD(256), CFTSUP(256), CFTWAK(256), TABXI(256)
     1,     TKO(40), TK(100), TETA(100,40), TDLDK(100,40), TPSIO(100,40)
     2,     TDPSIO(100,40), TDPSIB(100,40), TWAKI(100,40), TSUPT(100,40)
      EQUIVALENCE (TEMP1,CFTBOD)
```

```
      DIMENSION TEMP1(9,1)
      DIMENSION ID(4), ISYM(4)
C
C
C     ID(NREC) IDENTIFIES THE CONTENTS OF RECORD NUMBER NREC
C     SETTINGS ARE 1=XI, 2=BODY, 3=WAKE, 4=SUPERSTRUCTURE
C     ISYM(NREC) INDICATES THE SYMMETRY OF THE CORRESPONDING TRANSFORM
C     SKIP IF NOT 1ST PASS
      IF (MODE .NE. MINMOD)  GO TO 60
      NREC = 0
      IF (IBODY .EQ. 0)  GO TO 10
      NREC = NREC+1
      ID(NREC) = 2
      ISYM(NREC) = IBSYM
C
   10 IF (IWAKE .EQ. 0)  GO TO 20
      NREC = NREC+1
      ID(NREC) = 3
      ISYM(NREC) = IWSYM
C
   20 IF (ISUPR .EQ. 0)  GO TO 50
      NREC = NREC+1
      ID(NREC) = 4
      ISYM(NREC) = ISSYM
C     XI MUST BE LAST
   50 NREC = NREC+1
      ID(NREC) = 1
      ISYM(NREC) = 0
      REWIND NTTEMP
      WRITE(NTTEMP) ID,ISYM
C
C     WRITE TRANSFORMS IN SAME ORDER AS SOURCES ABOVE
   60 IF (IBODY .EQ. 0)  GO TO 70
      WRITE(NTTEMP) (CFTBOD(IETA),IETA=1,NETA)
   70 IF (IWAKE .EQ. 0)  GO TO 80
      WRITE(NTTEMP) (CFTWAK(IETA),IETA=1,NETA)
   80 IF (ISUPR .EQ. 0)  GO TO 110
      WRITE(NTTEMP) (CFTSUP(IETA),IETA=1,NETA)
  110 WRITE(NTTEMP) (TABXI(IETA),IETA=1,NETA)
      RETURN
      END
      SUBROUTINE INCON
C     INPUT CONTROL ROUTINE
C
      COMMON /CONTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
     1,     NODISP, NOGRID
      EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
     1,     (MODBOD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
      COMMON // LENCK, NBLOKS, ITHCOM, IDENT, ITHTYP, INTYPE
     1,     LSTCHK(1000), LSTSAV(10000)
C
C     *****SUMMARY OF APPROACH*****
C     PROGRAM INPUT MAY COME FROM 3 SOURCES
C       1. INPUT FILE (NAMELIST DECK IMMEDIATELY FOLLOWING AN INP
C          CONTROL CARD)
C       2. DATA LIBRARY FILE (LIB CARD IN INPUT STREAM SPECIFIES WHICH
C          NAMELIST DECK TO READ FROM LIBRARY FILE)
C       3. DISPERSION TABLE FILE (HANDLED BY DT ROUTINES, NOT HERE)
C
C     THE CHECKLIST IS A LIST OF INPUT VARIABLES WITH ONE OR MORE
C     FLAGS SPECIFIED FOR EACH VARIABLE.  THE PROGRAM SAVES ALL
```

```
C     CHECKLIST VARIABLES BEFORE READING INPUT AND COMPARES VALUES
C     AFTER INPUT, SETTING FLAGS INDICATING CHANGES.  IN MULTI-CASE
C     JOBS, THIS PERMITS THE PROGRAM TO DETERMINE WHAT RESULTS CAN BE
C     CARRIED OVER FROM THE PREVIOUS CASE AND WHAT MUST BE RECOMPUTED.
C
C     OUTPUT FORMAT SPECIFICATIONS CAN BE INPUT FOR EACH SET OF DATA
C     SENT TO THE PRINT/PLOT (PP) PROCESSOR.  SPECS FOR ALL PP SETS
C     ARE INPUT TO A SINGLE SET OF VARIABLES.  TWO OF THOSE VARIABLES
C     (IOPP AND IOCUR) IDENTIFY THE PARTICULAR PP SET TO WHICH THE
C     SPECS APPLY.  BEFORE READING ANY PP SPEC, NULL VALUES ARE
C     INSERTED IN THE PP INPUT VARIABLES.  AFTER READING, THE VALUES
C     ARE MOVED TO A SAVE ARRAY UNTIL END OF INPUT PROCESSING WHEN
C     THEY ARE ALL WRITTEN ON A FILE.  AT THE START OF THE NEXT CASE,
C     THIS FILE IS READ SO THAT PP SPECS WILL ACCUMULATE FROM CASE
C     TO CASE.
C
C
      CALL TIMER(1)
C     BUMP CASE NUMBER (=0 ON 1ST CALL TO INCON)
      ICASE = ICASE + 1
C     ONCE PER RUN INITIALIZATIONS
      IF (ICASE .EQ. 1)  CALL  INRUN1
C     RESTORE PRINT/PLOT DATA
      CALL INRSPP
C     SET UP CHECK LIST
      CALL INCKO
C     SAVE CHECK LIST VARIABLES BEFORE INPUT
      CALL INSVCK
C
C     READ INPUT PROCESSOR COMMAND
   10 CALL INRCOM
      GO TO (20,30,40,50), ITHCOM
C     READ INPUT DATA FROM TAPE 5
   20 CALL INRDAT(5)
      GO TO 10
C
C     READ LIBRARY DATA
   30 CALL INLIB
      GO TO 10
C
C     END OF RUN--NO RETURN
   40 CALL ENDRUN
C
C     END OF CASE INPUT. COMPARE VARIABLES AFTER INPUT WITH
C     VALUES SAVED BEFORE INPUT, SET APPROPRIATE CHECK LIST FLAGS
   50 CALL INCHK
C     SAVE P/P DATA
      CALL INSVPP
      CALL TIMER(-1)
C     RETURN TO EXECUTE THE CASE
      RETURN
      END
      SUBROUTINE INCHK
C     COMPARE VARIABLES IN CHECKLIST WITH THOSE SAVED PREVIOUSLY
C
      COMMON /CONTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
     1,     NODISP, NOGRID
      EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
     1,     (MODBOD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
      COMMON // LENCK, NBLOKS, ITHCOM, IDENT, ITHTYP, INTYPE
     1,     LSTCHK(1000), LSTSAV(1000)
```

```
      DIMENSION IBASE(1)
C
C
C     INDEX INTO CHECKLIST VARIABLE STORAGE
      ISAV = 0
C     DIVISORS TO SHIFT 6 AND 12 OCTAL DIGITS
      ISHF6 = 8**6
      ISHF12 = 8**12
C     INITIALLY, SET FOR ALL CHECKLIST FLAGS OFF
      IFLAG = 0
C
C     LOOP FOR EACH ENTRY IN CHECKLIST
      DO 80  ITHCK=1,LENCK
C     PICK UP TEST, DIMENSION, LOC OF NEXT ENTRY IN CHKLIST.
C     REMOVE FLAG BITS
      ITEST = LSTCHK(ITHCK) .AND. 77777777777777B
C     MASK FOR ADDRESS, DROP TEST AND DIMENSION
      LOC = ITEST .AND. 777777B
C     CONVERT ABSOLUTE ADDRESS TO IBASE INDEX
      LOC = LOC -LOCF(IBASE) +1
C     SHIFT OUT ADDRESS
      IDIM = ITEST/ISHF6
C     MASK FOR DIMENSION, REMOVE TEST
      IDIM = IDIM .AND. 777777B
C     SHIFT OUT DIMENSION AND ADDRESS, RETAIN TEST
      ITEST = ITEST/ISHF12
      GO TO (10,20,30,40,50,60), ITEST
C
   10 DO 15  I=1,IDIM
      ISAV = ISAV+1
      IF (IBASE(LOC) .NE. LSTSAV(ISAV))  GO TO 70
   15 LOC = LOC+1
      GO TO 80
C
   20 DO 25  I=1,IDIM
      ISAV = ISAV+1
      IF (IBASE(LOC) .LT. LSTSAV(ISAV))  GO TO 70
   25 LOC = LOC+1
      GO TO 80
C
   30 DO 35  I=1,IDIM
      ISAV = ISAV+1
      IF (IBASE(LOC) .LE. LSTSAV(ISAV))  GO TO 70
   35 LOC = LOC+1
      GO TO 80
C
   40 DO 45  I=1,IDIM
      ISAV = ISAV+1
      IF (IBASE(LOC) .EQ. LSTSAV(ISAV))  GO TO 70
   45 LOC = LOC+1
      GO TO 80
C
   50 DO 55  I=1,IDIM
      ISAV = ISAV+1
      IF (IBASE(LOC) .GE. LSTSAV(ISAV))  GO TO 70
   55 LOC = LOC+1
      GO TO 80
C
   60 DO 65  I=1,IDIM
      ISAV = ISAV+1
      IF (IBASE(LOC) .GT. LSTSAV(ISAV))  GO TO 70
   65 LOC = LOC+1
```

```
      GO TO 80
C
C     TEST IS SATISFIED.  MASK TO SAVE FLAGS, REMOVE OTHER STUFF
   70 ITEMP = LSTCHK(ITHCK) .AND. 77777770000000000000B
C     SET TO TURN ON ALL FLAGS ASSOCIATED WITH THIS VARIABLE
      IFLAG = IFLAG .OR. ITEMP
   80 CONTINUE
C
C
C     SET INDIVIDUAL FLAGS BASED ON BITS IN IFLAG
      MASK = 4000000000000B
      DO 100  I=1,20
C     SHIFT THE ON BIT LEFT BY 1
      MASK = MASK + MASK
C     PICK OUT CORRESPONDING BIT IN IFLAG
      ITEMP = MASK .AND. IFLAG
      ICKFLG(I) = 0
  100 IF (ITEMP .NE. 0)  ICKFLG(I) = 1
      RETURN
      END
      SUBROUTINE INCKO
C     SET UP CHECKLIST
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,     RBSEP2, RBSTR, RBLIM
C
      COMMON /CONST/ JDK, JDMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,     JDCKSV, JDMSP, JDEDGE
C
      COMMON /CONTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
     1,     NCDISP, NOGRIC
      EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
     1,     (MODBOD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,     X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,     IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,     ISPHAS
C
      COMMON /OCEAN/ LIBSEA, MODES, NK, TABK(100), NZT, DKRAT, DTDEP
     1,     TDEPMX, TDEP(400), NFLAG, SQBV(400), OCNDEP, RKMAX
     2,     SQN(400), NKT, IPPVEC
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,     SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMMON /WAKE/ IWAKE, CWAKR, CWAKX, XWAKE, WAKRAD, XWNOM
     1,     RESLVS, CWAKM
C
      COMMON // LENCK, NBLOKS, ITHCOM, IDENT, ITHTYP, INTYPE
     1,     LSTCHK(1000), LSTSAV(10000)
      INTEGER EQ,GE,GT
      DATA NE,LT,LE,EQ,GE,GT/1,2,3,4,5,6/
C
C
C     TURN ON A DIFFERENT BIT FOR EACH CHECKLIST FLAG
      ION = 4000000000000B
      DO 10  I=1,20
C     SHIFT THE ON BIT LEFT BY 1
      ION = ION + ION
   10 ICKFLG(I) = ION
C     INITIALIZE NUMBER OF ENTRIES IN CHECKLIST
```

```
      LENCK = 0
C
C     SET UP CHECKLIST.  EACH CALL TO INCK1 INSERTS 1 ENTRY
C     INTO CHECKLIST
C     CALL INCK1(VARBLE,DIMEN,TEST,FLAGS)
C     OPERATION IS
C     FLAGS = 0
C     IF (VARBLE(AFTERINPUT) .TEST. VARBLE(BEFOREINPUT)) FLAGS = 1
      CALL INCK1(NK,1,NE,MODSEA)
      CALL INCK1(DKRAT,1,NE,MODSEA)
      CALL INCK1(RKMAX,1,NE,MODSEA)
      CALL INCK1(TABK,JDK,NE,MODSEA)
      CALL INCK1(OCNDEP,1,NE,MODSEA)
      CALL INCK1(MODES,1,GT,MODSEA)
      CALL INCK1(NZT,1,NE,MODSEA)
      CALL INCK1(TDEP,JDTCL,NE,MODSEA)
      CALL INCK1(SQBV,JDTCL,NE,MODSEA)
      CALL INCK1(DTDEP,1,NE,MODSEA)
      CALL INCK1(TDEPMX,1,NE,MODSEA)
      CALL INCK1(NFLAG,1,NE,MODSEA)
      CALL INCK1(OBSDEP,1,NE,MODOBS)
      CALL INCK1(BODDEP,1,NE,MODBOD+MODSUP+MODWAK)
      CALL INCK1(SUPTOP,1,NE,MODSUP)
      CALL INCK1(SUPBOT,1,NE,MODSUP)
      CALL INCK1(IBODY,1,GT,MODBOD)
      CALL INCK1(ISUPR,1,GT,MODSUP)
      CALL INCK1(IWAKE,1,GT,MODWAK)
      CALL INCK1(BODSPD,1,NE,MODWAK)
      CALL INCK1(BODDIA,1,NE,MODWAK)
      CALL INCK1(CWAKR,1,NE,MODWAK)
      CALL INCK1(RESLVS,1,NE,MODWAK)
      RETURN
      END
      SUBROUTINE INCK1(VAR,IDIM,ITEST,IFLAGS)
C     INSERT 1 ENTRY INTO CHECKLIST
C
      COMMON /CONST/ JDK, JDMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,       JDCKSV, JDMSP, JDEDGE
C
      COMMON // LENCK, NBLOKS, ITHCUM, IDENT, ITHTYP, INTYPE
     1,       LSTCHK(1000), LSTSAV(10000)
      DATA I6,I12/1000000B,1000000000000B/
C
C
C     BUMP NUMBER OF ENTRIES
      LENCK = LENCK+1
      IF (LENCK .LE. JDCKL)  GO TO 20
      WRITE(6,10) JDCKL
   10 FORMAT(11H DIMENSION=,I6,22H OF CHECKLIST EXCEEDED)
      CALL ERRXIT
C     THE OCTAL REPRESENTATION OF THE ENTRY IS FFFFFFFTDDDDDDLLLLLL
C     WHERE F ARE FLAG BITS, T IS TEST TO APPLY, D IS
C     DIMENSION, L IS LOC OF VARIABLE
   20 LSTCHK(LENCK) = LOCF(VAR) + I6*IDIM + I12*ITEST + IFLAGS
      RETURN
      END
      SUBROUTINE INLIB
C     READ DESIRED DATA SET FROM DATA LIBRARY FILE
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,       NTDTAB, NTEVEC, NTTEMP
C
```

```
      COMMON // LENCK, NBLOKS, ITHCOM, IDENT, ITHTYP, INTYPE
     1,     LSTCHK(1000), LSTSAV(10000)
C
C
C     NAMELIST DECKS ARE SEPARATED BY CARDS OF THE FORM *T,I WHERE
C     * IS IN CC1, T=(1 CHAR DATA TYPE), I=(10 CHAR ID)
      REWIND NTILIB
C     LOOP THROUGH CARDS IN LIBRARY LOOKING FOR * IN CC1
   10 READ(NTILIB,20) ICC1,ITREF,IDREF
   20 FORMAT(2A1,1X,A10)
      IF (ICC1 .NE. 1H*)  GO TO 10
C     FOUND A DECK SEPARATOR.  LAST CARD ON LIB FILE IS *END
      IF (ITREF .NE. 1HE)  GO TO 40
      WRITE(6,30)  INTYPE,IDENT
   30 FORMAT(29H LIBRARY DATA SET NOT FOUND--,A1,1H,,A10)
      CALL ERRXIT
C     CHECK FOR PROPER TYPE AND ID
   40 IF (ITREF .NE. INTYPE)  GO TO 10
      IF (IDREF .NE. IDENT)  GO TO 10
C     EVERYTHING MATCHES.  READ NAMELIST DECK FROM LIBRARY
      CALL INRDAT(NTILIB)
      RETURN
      END
      SUBROUTINE INMVPP
C     MOVE INPUT PP BLOCK TO IBLOKS ARRAY
C
      COMMON /CONST/ JDK, JOMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,     JDCKSV, JDMSP, JOEDGE
C
      COMMON // LENCK, NBLOKS, ITHCOM, IDENT, ITHTYP, INTYPE
     1,     LSTCHK(1000), LSTSAV(10000)
      COMMON//
     1      LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,     VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,     IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,     IDPP, NV, ISYM
     E,     ENDPP, IBLOKS(1)
      DIMENSION IDBLOK(1),DUMMY(1),IDPP(1),IOCUR(1)
      EQUIVALENCE (DUMMY,LENPP),(IDBLOK,DUMMY(2))
C
C
C     INTERPRET BLANK IN IDPP AS A NO-ENTRY
      IF (IDPP .EQ. 1H )  IDPP = NULL
C     INTERPRET 0 IN IOCUR AS A NO-ENTRY.  ENTRY TO IOCUR PERMITTED
C     ONLY IF IDPP WAS INPUT
      IF (IOCUR .EQ. 0  .OR.  IDPP .EQ. NULL)  IOCUR = NULL
      LOC1 = LENPP+1
C     SKIP IF THERE ARE NOT YET ANY PP BLOCKS IN IBLOKS ARRAY
      IF (NBLOKS .EQ. 0)  GO TO 20
C     CHECK WHETHER THIS BLOCK HAS BEEN INPUT BEFORE (SEE IF
C     INPUT ID AND OCCURANCE NUMBER MATCH WITH THOSE IN IBLOKS)
C     LOOP FOR EACH BLOCK STORED IN IBLOKS
      DO 10  I=1,NBLOKS
      IF (IDPP .EQ. IDPP(LOC1)  .AND.  IOCUR .EQ. IOCUR(LOC1))  GO TO 40
   10 LOC1 = LOC1 +LENPP
C
C     NO MATCH.  THIS IS A NEW BLOCK.  MOVE IT IN BACK OF ANY
C     BLOCKS ALREADY STORED
   20 DO 30  I=1,LENPP
      IDBLCK(LOC1) = IDBLOK(I)
   30 LOC1 = LOC1 +1
C     BUMP NUMBER OF BLOCKS STORED
```

```
      NBLOKS = NBLOKS +1
      RETURN
C
C     BLOCK WAS ALREADY INPUT.  OVERLAY OLD DATA WITH THE NEW INPUTS
   40 DO 50   I=1,LENPP
      IF (IDBLOK(I) .NE. NULL)  IDBLOK(LOC1) = IDBLOK(I)
   50 LOC1 = LOC1 +1
      RETURN
      END
      SUBROUTINE INRCOM
C     READ INPUT PROCESSOR CONTROL CARD COMMAND
C
      COMMON // LENCK, NBLOKS, ITHCOM, IDENT, ITHTYP, INTYPE
     1,     LSTCHK(1000), LSTSAV(10000)
      DIMENSION LSTCOM(4),LSTTYP(4)
      DATA LSTCOM/3HINP, 3HLIB, 3HEND, 3HRUN/
      DATA LSTTYP/1HO, 1HS, 1HG, 1HP/
C
C
C     READ COMMAND, DATA-TYPE, IDENTIFIER (IC1,IC2 ARE COMMAS)
      READ(5,10) KOMAND,IC1,INTYPE,IC2,IDENT
   10 FORMAT(A3,3A1,A10)
      WRITE(6,20) KOMAND,IC1,INTYPE,IC2,IDENT
   20 FORMAT(1X,A3,3A1,A10)
C     MATCH INPUT COMMAND WITH LIST OF POSSIBLES, GET ITHCOM=COMMAND NUM
      DO 30   ITHCOM=1,4
   30 IF (KOMAND .EQ. LSTCOM(ITHCOM))  GO TO 50
      WRITE(6,40)
   40 FORMAT(46H ILLEGAL INPUT PROCESSOR COMMAND ON ABOVE CARD)
      CALL ERRXIT
C
C     SKIP IF THIS COMMAND DOES NOT HAVE A DATA TYPE
   50 IF (ITHCOM .GT. 2)  GO TO 80
C     MATCH INPUT DATA-TYPE WITH LIST OF POSSIBLES, GET ITHTYP=TYPE NUM
      DO 60   ITHTYP=1,4
   60 IF (INTYPE .EQ. LSTTYP(ITHTYP))  GO TO 80
      WRITE(6,70)
   70 FORMAT(32H ILLEGAL DATA-TYPE ON ABOVE CARD)
      CALL ERRXIT
C
   80 RETURN
      END
      SUBROUTINE INRDAT(IFILE)
C     READ NAMELIST DATA FROM LOGICAL TAPE IFILE
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,     RBSEP2, RBSTR, RBLIM
C
      COMMON /CONST/ JDK, JDMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,     JDCKSV, JDMSP, JDEDGE
C
      COMMON /CONTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
     1,     NODISP, NOGRID
      EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
     1,     (MODBOD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,     X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,     IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,     ISPHAS
C
      COMMON /OCEAN/ LIBSEA, MODES, NK, TABK(100), NZT, DKRAT, DTDEP
```

```
     1,     TDEPMX, TDEP(400), NFLAG, SQBV(400), OCNDEP, RKMAX
     2,     SGN(400), NKT, IPPVEC
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,     SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMMON /WAKE/ IWAKE, CWAKR, CWAKX, XWAKE, WAKRAD, XWNOM
     1,     RESLVS, CWAKM
C
      COMMON // LENCK, NBLOKS, ITHCOM, IDENT, ITHTYP, INTYPE
     1,     LSTCHK(1000), LSTSAV(10000)
      COMMON//
     1      LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,     VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,     IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,     IDPP, NV, ISYM
     E,     ENDPP, IBLOKS(1)
      DIMENSION IDBLOK(1),DUMMY(1)
      EQUIVALENCE (DUMMY,LENPP),(IDBLOK,DUMMY(2))
      NAMELIST/OCEAN/ IPRDT, IPPDT, LIBSEA, MODES, NK, TABK, NZT
     1,    DKRAT, DTDEP, TDEPMX, TDEP, NFLAG, SQBV, OCNDEP
     2,     NODISP, IPPVEC, RKMAX, IPREDG
      NAMELIST/SOURCE/ BODDEP, BODDIA, BODLEN, IBODY, BODSPD, IPBODY
     1,    ISUPR, SUPTOP, SUPBOT, IPSUPR, SUPMID, SUPDIA, SUPLEN
     2,    IWAKE, CWAKR, CWAKX, RESLVS, CWAKM
      NAMELIST/GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS, DX, XMIN
     1,    NX, DY, NY, MODE1, MODEN, IVAR, XPMAX, YPMAX, NOGRID
     2,    IPPPSD, YMIN, ISPHAS
      NAMELIST/PP/ PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX, VLEN
     1,    FNAME, FMIN, FMAX, FLEN, FTODP, TITLE, IOCUR, IPLTYP
     2,    IPLOT, IPRINT, IEDIT, NOPP, IDPP, ISYM
C
C
C     GO READ DESIRED TYPE OF DATA
      GO TO (10,20,30,40),ITHTYP
C
C     TYPE1--O
   10 READ(IFILE,OCEAN)
      GO TO 100
C
C     TYPE2--S
   20 READ(IFILE,SOURCE)
      GO TO 100
C
C     TYPE 3--G
   30 READ(IFILE,GRID)
      GO TO 100
C
C     TYPE 4--P
C     INSERT NO-ENTRY VALUES IN P/P INPUT BLOCK
   40 DO 50  I=1,LENPP
   50 IDBLOK(I) = NULL
      READ(IFILE,PP)
C     MOVE INPUT PP BLOCK TO IBLOKS ARRAY
      CALL INMVPP
C
  100 RETURN
      END
      SUBROUTINE INRSPP
C     RESTORE PRINT/PLOT SPECIFICATIONS FROM PREVIOUS CASE
C
      COMMON /CCNTRL/ ICASE, ICKFLG(20), JDISP, JFFT, JPOT
```

```
     1,      NODISP, NOGRID
      EQUIVALENCE (MODSEA,ICKFLG(1)), (MODOBS,ICKFLG(2))
     1,      (MODBCD,ICKFLG(3)), (MODWAK,ICKFLG(4)), (MODSUP,ICKFLG(5))
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,      NTDTAB, NTEVEC, NTTEMP
C
      COMMON // LENCK, NBLOKS, ITHCOM, IDENT, ITHTYP, INTYPE
     1,      LSTCHK(1000), LSTSAV(10000)
      COMMON//
     1       LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,      VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,      IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,      IDPP, NV, ISYM
     E,      ENDPP, IBLOKS(1)
C
C
      IF (ICASE .GT. 1)  GO TO 10
C     NOTHING TO RESTORE ON FIRST CASE
C     INITIALIZE LENGTH OF SPEC BLOCK, NUMBER OF BLOCKS
      LENPP = LOCF(ENDPP) - LOCF(LENPP)
      NBLOKS = 0
      RETURN
C
   10 REWIND NTPDEF
      READ(NTPDEF) NBLOKS,LENPP
C     RETURN IF THERE WERE NO SPECS
      IF (NBLOKS .LE. 0)  RETURN
C     RESTORE PP SPECIFICATIONS FROM PREVIOUS CASE
      DO 20  I=1,NBLOKS
      LIM2 = I*LENPP
      LIM1 = LIM2 -LENPP +1
   20 READ(NTPDEF) (IBLOKS(J),J=LIM1,LIM2)
      RETURN
      END
      SUBROUTINE INRUN1
C     ONCE PER RUN INITIALIZATIONS
C
      COMMON /CONST/ JDK, JDMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,      JDCKSV, JDMSP, JDEDGE
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,      NTDTAB, NTEVEC, NTTEMP
C
      COMMON/PPCOM/
     1       LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,      VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,      IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,      IDPP, NV, ISYM
     E,      ENDPP, IBLOKS(1)
C
C
C     DIMENSIONS
C     K (WAVE NUMBER ARRAY)
      JDK = 100
C     MAX MODE NUMBER (DT ROUTINES)
      JDMODE = 80
C     MAX MODE RANGE FOR FT ROUTINES
      JDMFT = 40
C     MAX MODE RANGE FOR SP ROUTINES
      JDMSP = 41
C     MAX NUMBER OF ENTRIES IN WAVE FAMILY EDGE TABLES
```

```
      JDEDGE = 20
C     NUMBER OF POINTS IN THERMOCLINE
      JDTCL = 400
C     DIMENSION OF CHECKLIST
      JDCKL = 1000
C     DIMENSION OF CHECKLIST VARIABLE SAVE STORAGE
      JDCKSV = 10000
C
C     NO-ENTRY VALUE (TO CHECK WHETHER A VARIABLE WAS INPUT)
      NULL = 4HNULL
C
      PI = 3.14159265359
C
C     TAPE UNIT ASSIGNMENTS
C     INPUT DATA LIBRARY
      NTILIB = 1
C     DISPERSION LIBRARY
      NTDLIB = 2
C     DISPERSION TABLE--NOTE ASSIGNMENTS FOR NTDTAB AND NTTEMP ARE
C     SWITCHED EACH ENTRY TO DTCON
      NTDTAB = 3
C     TRANSFORMS
      NTTEMP = 4
C     (INPUT=5, OUTPUT=6)
C     EIGENVECTORS
      NTEVEC = 7
C     P/P DEFINITIONS (INPUT IMAGE)
      NTPDEF = 8
C     P/P DATA FILE
      NTPDAT = 9
C     P/P ID FILE
      NTPID = 10
C     PLOT OUTPUT
      NTPLOT = 50
C
C     TELL ROUTINE SETID IT IS OK TO START A NEW PP SET
      IDPP = 1H
      RETURN
      END
      SUBROUTINE INSVCK
C     SAVE THE VARIABLES IN THE CHECKLIST IN ARRAY LSTSAV
C
      COMMON /CONST/ JDK, JDMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,     JDCKSV, JDMSP, JDEDGE
C
      COMMON // LENCK, NBLOKS, ITHCOM, IDENT, ITHTYP, INTYPE
     1,     LSTCHK(1000), LSTSAV(10000)
      DIMENSION IBASE(1)
C
C
C     INDEX TO CHECKLIST VARIABLES STORAGE
      ISAV = 0
C     LOOP FOR EACH ENTRY IN CHECKLIST
      DO 20  ITHCK=1,LENCK
C     PICK UP DIMENSION, LOC OF NEXT ENTRY.  MASK OUT OTHER STUFF
      IDIM = LSTCHK(ITHCK) .AND. 777777777777B
C     MASK TO DROP DIMENSION, PICK UP ADDRESS
      LOC = IDIM .AND. 777777B
C     CONVERT ABSOLUTE ADDRESS TO IBASE INDEX
      LOC = LOC -LOCF(IBASE) +1
C     SHIFT OUT ADDRESS TO GET DIMENSION
      IDIM = IDIM/1000000B
```

```
C     SAVE EACH WORD
      DO 10  I=1,IDIM
      ISAV = ISAV +1
      LSTSAV(ISAV) = IBASE(LOC)
   10 LOC = LOC +1
   20 CONTINUE
C     COMPARE NUMBER OF VARIABLES SAVED WITH DIMENSION OF SAVE ARRAY
      IF (ISAV .LE. JDCKSV)  GO TO 40
      WRITE(6,30) ISAV,JDCKSV
   30 FORMAT(41H CHECKLIST VARIABLE SAVE STORAGE EXCEEDED,2I10)
      CALL ERRXIT
   40 RETURN
      END
      SUBROUTINE INSVPP
C     SAVE PRINT/PLOT SPECIFICATIONS FOR THIS CASE
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,      NTDTAB, NTEVEC, NTTEMP
C
      COMMON // LENCK, NBLOKS, ITHCOM, IDENT, ITHTYP, INTYPE
     1,      LSTCHK(1000), LSTSAV(10000)
      COMMON//
     1       LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,      VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,      IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,      IDPP, NV, ISYM
     E,      ENDPP, IBLOKS(1)
C
C
      REWIND NTPDEF
C     NUMBER OF BLOCKS, LENGTH OF EACH
      WRITE(NTPDEF) NBLOKS,LENPP
      IF (NBLOKS .LE. 0)  GO TO 20
      DO 10  I=1,NBLCKS
      LIM2 = I*LENPP
      LIM1 = LIM2 -LENPP +1
   10 WRITE(NTPDEF) (IBLOKS(I),I=LIM1,LIM2)
   20 RETURN
      END
      SUBROUTINE PPCON
C     PRINT/PLOT CONTROL
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,      NTDTAB, NTEVEC, NTTEMP
C
      COMMON // VLIST(2050), FLIST(2050), NOPROC, IPVAL, NBLOKS
     1,      IPPEND, JEDIT, MATCH, ISKIPP, LIMLO, LIMHI, PVAL, LENDAT
     2,      LIMV1, LIMVE, ITHOUT, HT, PVALS(1000), FMAXS(1000)
     3,      VFMAXS(1000), FMINS(1000), VFMINS(1000), FIS(1000)
     4,      SQFIS(1000), IPBUF(1024), RANGEP, FIRSTP, DELTAP
     5,      FIRSTV, DELTAV, FIRSTF, DELTAF
      COMMON //
     1       LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,      VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,      IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,      IDPP, NV, ISYM
     E,      ENDPP, IBLOKS(1)
C
C
      CALL TIMER(13)
C     LOAD ALL P/P SPECIFICATIONS WHICH WERE INPUT
      CALL PPSPEC
```

```
C     RETURN IF NO P/P FOR THIS CASE
      IF (IPPEND .NE. 0)  GO TO 40
C
C     READ PPFB FROM PROGRAM TAPE, SET UP VLIST IF APPROPRIATE
   10 CALL PPTDEF
C     JUMP TO TERMINATE P/P PROCESSING FOR THIS CASE
      IF (IPPEND .NE. 0)  GO TO 30
C     ENFORCE INPUT SPECS TO YIELD NET P/P DEFINITION
      CALL PPIDEF
      IF (IPPEND .NE. 0)  GO TO 30
C     SET LIMITS, SCALES, ETC.   DRAW AXES IF PLOTTING
      CALL PPSET
C
C     LOOP FOR EACH DATA RECORD (EACH PARAMETER VALUE) IN THIS PP SET
      DO 20  IPVAL=1,NP
C     READ DATA RECORD AND DECIDE WHETHER TO PROCESS IT
      CALL PPDATA
C     SKIP IF DATA IS NOT TO BE PROCESSED
      IF (NOPROC .NE. 0)  GO TO 20
C     PRINT
      IF (IPRINT .NE. 0)  CALL PPRINT
C     PLOT
      IF (IPLOT .NE. 0)  CALL PPLOT
   20 CONTINUE
C
C     SET ORIGIN FOR NEXT PLOT.  NOTE NEW ORIGIN EMPTIES PLOT
C     BUFFER WHICH PERMITS OVERLAY OF BUFFER AFTER ANY PLOT COMPLETED
      IF (IPLOT .NE. 0  .AND.  ISKIPP .EQ. 0)  CALL PPLORG
C     DO SUMMARY PRINT IF REQUESTED
      IF (IPRINT .GE. 2  .AND.  ISKIPP .EQ. 0)  CALL PPSUM
C     LOOP FOR NEXT PP SET
      GO TO 10
C
C
C     TERMINATE PP PROCESSING FOR THIS CASE
   30 REWIND NTPID
      REWIND NTPDAT
   40 CALL TIMER(-13)
      RETURN
      END
      SUBROUTINE PPAXIS(XO, YO, LABEL, LENLAB, AXLEN, ROT, FIRST, DELTA)
C     DRAW AND LABEL AXES
C     NOTE CALLING SEQ IS SAME AS CALCOMPS AXIS ROUTINE
C     ALSO NOTE THIS IS NOT AS GENERAL AS AXIS (AXIS MAY REPLACE
C     ANY CALL TO PPAXIS BUT NOT VICE-VERSA)
C
C
C     LENGTH OF AXIS LABEL
      LLAB = IABS(LENLAB)
C     CHARACTER SIZE
      HT = .105
C     DIRECTION COSINES OF AXIS--PRESET FOR X AXIS
      CX = 1.
      CY = 0.
C     SKIP IF THIS IS X AXIS
      IF (LENLAB .LT. 0)  GO TO 10
C     Y AXIS
      CX = 0.
      CY = 1.
C     LOC OF OUTSIDE OF TIC MARK WRT INSIDE
   10 DXTIC = CX*(0.)      + CY*(-HT/2.)
      DYTIC = CX*(-HT/2.) + CY*(0.)
```

```
C     LOC OF START OF SCALE WRT INSIDE OF TIC MARK
      DXSCA = CX*(-8.*HT+.025) + CY*(-10.*HT-.025)
      DYSCA = CX*(-1.5*HT) + CY*(0.)
C     LOC OF START OF AXIS NAME WRT END OF AXIS
      DXNAM = CX*(-10.*HT) + CY*(-10.*HT)
      DYNAM = CX*(-3.*HT)  + CY*(1.5*HT)
C     INCHES ALONG AXIS NEEDED TO WRITE SCALE
      DAMIN = CX*(9.*HT)  + CY*(HT+.1)
C     CURRENT POSITION ALONG AXIS
      XLOC = X0
      YLOC = Y0
C     CURRENT LENGTH OF AXIS
      ALEN = 0.
C     MOVE PEN TO ORIGIN
      CALL PLOT(X0, Y0, 3)
C
C     LOOP FOR EACH INCH OF AXIS--MAX LENGTH 100
      DO 30  ITHTIC=1,100
C     DRAW TIC MARK
      CALL PLOT(XLOC+DXTIC, YLOC+DYTIC, 2)
C     SKIP SCALE IF NOT ENOUGH ROOM TO DRAW IT
      IF (ALEN+DAMIN .GT. AXLEN)  GO TO 20
C     CONVERT SCALE NUMBER TO DISPLAY CODE
      SCALE = PPBCI(DELTA*ALEN+FIRST)
C     DRAW IT
      CALL SYMBOL(XLOC+DXSCA, YLOC+DYSCA, HT, SCALE, 0., 10)
C     BRING PEN (UP) BACK TO INSIDE TIC
   20 CALL PLOT(XLOC, YLOC, 3)
C     GET AXIS LENGTH AT NEXT TIC
      ALEN = AMIN1(AXLEN, ALEN+1.)
C     EXTEND AXIS TO NEXT TIC
      XLOC = X0 + CX*ALEN
      YLOC = Y0 + CY*ALEN
      CALL PLOT(XLOC, YLOC, 2)
   30 IF (ALEN .GE. AXLEN)  GO TO 40
C
C     DRAW TIC AND SCALE FOR END OF AXIS
   40 CALL PLOT(XLOC+DXTIC, YLOC+DYTIC, 2)
      SCALE = PPBCI(DELTA*ALEN+FIRST)
      CALL SYMBOL(XLOC+DXSCA, YLOC+DYSCA, HT, SCALE, 0., 10)
C     DRAW NAME OF AXIS
      CALL SYMBOL(XLOC+DXNAM, YLOC+DYNAM, HT, LABEL, 0., LLAB)
C     FIND POSITION OF SCALE VALUE=ZERO ON AXIS
      ZLOC = -FIRST/DELTA
C     SKIP IF SCALE=0 IS OFF (OR AT END) OF AXIS
      IF (ZLOC .LE. 0.  .OR.  ZLOC .GE. ALEN)  GO TO 50
C     MARK THE AXIS AT SCALE=0
      ZX = X0 + CX*ZLOC
      ZY = Y0 + CY*ZLOC
      CALL PLOT(ZX, ZY, 3)
      CALL PLOT(ZX+CY*HT/2., ZY+CX*HT/2., 2)
   50 RETURN
      END
      FUNCTION PPBCI(RNUM)
C     CONVERT RNUM TO (MAX) 9 CHARACTER DISPLAY CODE
C
C
      ANUM = ABS(RNUM)
C     USE E FORMAT FOR LARGE NUMBERS
      IF (ANUM .GE. 100000.)  GO TO 10
C     USE E FORMAT FOR SMALL NUMBERS
      IF (0. .LT. ANUM  .AND.  ANUM .LT. .01)  GO TO 10
```

```
C     USE I FORMAT FOR INTEGERS
      INUM = RNUM
      IF (ABS(RNUM-FLOAT(INUM)) .LT. .00001)  GO TO 20
C     CHOOSE BETWEEN 2 F FORMATS TO MAINTAIN 3 DIGIT ACCURACY
      IF (ANUM .LT. 10.)  GO TO 30
      GO TO 40
C
C
   10 ENCODE(10,15,BCI) RNUM
   15 FORMAT(E10.2)
      GO TO 50
C
   20 ENCODE(10,25,BCI) INUM
   25 FORMAT(I10)
      GO TO 50
C
   30 ENCODE(10,35,BCI) RNUM
   35 FORMAT(F10.5)
      GO TO 50
C
   40 ENCODE(10,45,BCI) RNUM
   45 FORMAT(F10.2)
C
C
   50 PPBCI = BCI
      RETURN
      END
      SUBROUTINE PPDATA
C     READ NEXT DATA RECORD, CHECK LIMITS
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,     NTDTAB, NTEVEC, NTTEMP
C
      COMMON // VLIST(2050), FLIST(2050), NOPROC, IPVAL, NBLOKS
     1,     IPPEND, JEDIT, MATCH, ISKIPP, LIMLO, LIMHI, PVAL, LENDAT
     2,     LIMV1, LIMVE, ITHOUT, FT, PVALS(1000), FMAXS(1000)
     3,     VFMAXS(1000), FMINS(1000), VFMINS(1000), FIS(1000)
     4,     SQFIS(1000), IPBUF(1024), RANGEP, FIRSTP, DELTAP
     5,     FIRSTV, DELTAV, FIRSTF, DELTAF
      COMMON //
     1      LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,     VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,     IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,     IDPP, NV, ISYM
     E,     ENDPP, IBLOKS(1)
C
C
C     PRESET FLAG SO DATA READ HERE WILL BE PROCESSED
      NOPROC = 0
C     TEST DATA RECORD FORMAT
      IF (IVLIST .EQ. 3)  GO TO 10
C     VLIST IS FIXED.  READ PARAMETER VALUE, LOWER AND UPPER
C     LIMITS, FUNCTION VALUES
      READ(NTPDAT) PVAL,L1,L2,(FLIST(I),I=L1,L2)
C     SET FWA AND LWA OF DATA TO BE PROCESSED.  DATA MUST LIE WITHIN
C     (VMIN,VMAX) WINDOW AND HAVE A FUNCTION VALUE
      LIMV1 = MAX0(LIMLO,L1)
      LIMVE = MIN0(LIMHI,L2)
C     SKIP IF NO DATA (DONT TRY TO PROCESS JUST ONE POINT)
      IF (LIMV1 .GE. LIMVE)  GO TO 100
      LENDAT = LIMVE -LIMV1 +1
      GO TO 20
```

```
C     VLIST CHANGES WITH PARAMETER.  READ PARAMETER VALUE, LENGTH
C     OF V/FLIST, VARIABLES AND FUNCTIONS
   10 READ(NTPDAT) PVAL,LIMHI,(VLIST(I),FLIST(I),I=1,LIMHI)
      LIMLO = 1
C
C     JUMP IF THIS PP SET IS TO BE SKIPPED
   20 IF (ISKIPP .NE. 0)  GO TO 100
C     JUMP IF PARAMETER VALUE IS OUTSIDE DESIRED RANGE
      IF (PVAL .LT. PMIN  .OR.  PVAL .GT. PMAX)  GO TO 100
C     JUMP IF STORAGE LIMIT WOULD BE EXCEEDED
      IF (ITHOUT .GE. 1000)  GO TO 100
C     IF A NEW VLIST WAS JUST READ, FIND FWA AND LWA OF DATA IN
C     RANGE VMIN TO VMAX
      IF (IVLIST .EQ. 3)  CALL PPVLIM(NOPROC)
C     JUMP IF NO DATA IN THAT RANGE
      IF (NOPROC .NE. 0)  GO TO 100
C
C     BUMP NUMBER OF RECORDS (PARAMETER VALUES) PROCESSED
      ITHOUT = ITHOUT + 1
C     SAVE PARAMETER VALUE IN DISPLAY CODE
      PVALS(ITHOUT) = PPBCI(PVAL)
      RETURN
C
C
C     DO NOT PROCESS THIS DATA RECORD
  100 NOPROC = 1
      RETURN
      END
      SUBROUTINE PPIDEF
C     OVERLAY PROGRAM TAPE PPFB WITH INPUT PPFB (IF ANY)
C
      COMMON /CONST/ JDK, JDMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,      JDCKSV, JDMSP, JDEDGE
C
      COMMON // VLIST(2050), FLIST(2050), NOPROC, IPVAL, NBLOKS
     1,      IPPEND, JEDIT, MATCH, ISKIPP, LIMLO, LIMHI, PVAL, LENDAT
     2,      LIMV1, LIMVE, ITHOUT, FT, PVALS(1000), FMAXS(1000)
     3,      VFMAXS(1000), FMINS(1000), VFMINS(1000), FIS(1000)
     4,      SCFIS(1000), IPBUF(1024), RANGEP, FIRSTP, DELTAP
     5,      FIRSTV, DELTAV, FIRSTF, DELTAF
      COMMON //
     1       LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,      VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,      IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,      IDPP, NV, ISYM
     E,      ENDPP, IBLOKS(1)
      DIMENSION IOCUR(1), IDPP(1), DUMMY(1), IDBLOK(1)
      EQUIVALENCE (DUMMY,LENPP), (IDBLOK,DUMMY(2))
C
C
      MATCH = 0
C     JUMP IF NO PPFBS WERE INPUT (NOMINAL VALUES WILL STAND)
      IF (NBLOKS .EQ. 0)  GO TO 80
C     SCAN INPUT PPFBS FOR FOLLOWING PURPOSES
C     1. TERMINATE PP PROCESSOR IF EDITING (IE PROCESS ONLY DATA FOR
C        WHICH THERE IS AN INPUT PPFB) AND ALL INPUT PPFBS HAVE BEEN
C        DONE
C     2. DECREMENT OCCURANCE NUMBER OF ALL INPUT PPFBS WHICH HAVE AN
C        ID THAT MATCHES CURRENT DATA
C     3. CHECK FOR AN INPUT PPFB WHICH APPLIES TO THE CURRENT DATA
C        BY SETTING MATCH--BIT 0=MATCH WITH NULL ID, BIT 1=MATCH WITH
C        ID, BIT 2=MATCH WITH ID+OCCURANCE
```

```
      IF (JEDIT .NE. 0)  IPPEND = 1
      LOC = 1
      DO 40  ITHBLK=1,NBLOKS
      LOC = LOC + LENPP
      IF (IOCUR(LOC) .EQ. NULL  .OR.  IOCUR(LOC) .GT. 0)  IPPEND = 0
      IF (IDPP(LOC) .EQ. NULL)  MATCH = MATCH .OR. 1
      IF (IDPP(LOC) .NE. IDPP)  GO TO 40
      IF (IOCUR(LOC) .NE. NULL)  GO TO 30
      MATCH = MATCH .OR. 2
      GO TO 40
   30 IOCUR(LOC) = IOCUR(LOC) -1
      IF (IOCUR(LOC) .EQ. 0)  MATCH = MATCH .OR. 4
   40 CONTINUE
      IF (IPPEND .NE. 0)  RETURN
C
C     SKIP IF NO INPUT PPFB FOR CURRENT DATA
      IF (MATCH .EQ. 0)  GO TO 80
C     FIND THE INPUT PPFB WHICH APPLIES TO THIS DATA
      LOC = 1
      DO 50  ITHBLK=1,NBLOKS
      LOC = LOC + LENPP
      IF (IDPP(LOC) .EQ. NULL  .AND.  MATCH .EQ. 1)  GO TO 60
      IF (IDPP(LOC) .NE. IDPP)  GO TO 50
      IF (IOCUR(LOC) .EQ. NULL  .AND.  MATCH .LT. 4)  GO TO 60
      IF (IOCUR(LOC) .EQ. 0)  GO TO 60
   50 CONTINUE
C     NEVER FALL THROUGH ABOVE LOOP
C     OVERLAY PROGRAM PPFB WITH INPUT PPFB
   60 DO 70  I=1,LENPP
   70 IF (IDBLOK(I+LOC-1) .NE. NULL)  IDBLOK(I) = IDBLOK(I+LOC-1)
   80 RETURN
      END
      SUBROUTINE PPLORG
C     MOVE PEN ORG TO ORG OF NEXT PLOT AXES
C
      COMMON /PRTPLT/ IPLTON, XAORG, YAORG, XPORG, YPORG
C
      COMMON // VLIST(2050), FLIST(2050), NOPROC, IPVAL, NBLOKS
     1,      IPPEND, JEDIT, MATCH, ISKIPP, LIMLO, LIMHI, PVAL, LENDAT
     2,      LIMV1, LIMVE, ITHOUT, HT, PVALS(1000), FMAXS(1000)
     3,      VFMAXS(1000), FMINS(1000), VFMINS(1000), FIS(1000)
     4,      SQFIS(1000), IPBUF(1024), RANGEP, FIRSTP, DELTAP
     5,      FIRSTV, DELTAV, FIRSTF, DELTAF
      COMMON //
     1       LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,      VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,      IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,      IDPP, NV, ISYM
     E,      ENDPP, IBLOKS(1)
C
C
C     JUMP IF PLOT JUST FINISHED WAS A MULTI-TRACE PLOT
      IF (IPLTYP .NE. 0)  GO TO 10
C     RASTER
      CALL PLOT(PLEN+4.-XPORG, -YPORG, -3)
      GO TO 20
C     MULTI-TRACE
   10 CALL PLOT(VLEN+4.-XPORG, -YPORG, -3)
C
   20 XPORG = 0.
      YPORG = 0.
      RETURN
```

```
      END
      SUBROUTINE PPLOT
C     DRAW THE PLOT FOR THE NEXT PARAMETER VALUE
C
      COMMON /PRTPLT/ IPLTON, XAORG, YAORG, XPORG, YPORG
C
      COMMON // VLIST(2050), FLIST(2050), NOPROC, IPVAL, NBLOKS
     1,      IPPEND, JEDIT, MATCH, ISKIPP, LIMLO, LIMHI, PVAL, LENDAT
     2,      LIMV1, LIMVE, ITHOUT, HT, PVALS(1000), FMAXS(1000)
     3,      VFMAXS(1000), FMINS(1000), VFMINS(1000), FIS(1000)
     4,      SQFIS(1000), IPBJF(1024), RANGEP, FIRSTP, DELTAP
     5,      FIRSTV, DELTAV, FIRSTF, DELTAF
      COMMON //
     1       LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,      VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,      IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,      IDPP, NV, ISYM
     E,      ENDPP, IBLOKS(1)
C
C
C     INSERT SCALING PARAMETERS INTO VARIABLE, FUNCTION LISTS
      SAVV1 = VLIST(LIMVE+1)
      SAVV2 = VLIST(LIMVE+2)
      VLIST(LIMVE+1) = FIRSTV
      VLIST(LIMVE+2) = DELTAV
      FLIST(LIMVE+1) = FIRSTF
      FLIST(LIMVE+2) = DELTAF
C     JUMP FOR MULTI-TRACE PLOT
      IF (IPLTYP .NE. 0)  GO TO 10
C     RASTER PLOT
C     MOVE PEN ORIGIN TO ORIGIN OF THIS LINE
      X = (PVAL-FIRSTP) / DELTAP
      CALL PLOT(X-XPORG, -YPORG, -3)
      XPORG = X
      YPORG = 0.
C     DRAW THE LINE
      CALL LINE(FLIST(LIMV1), VLIST(LIMV1), LENDAT, 1, 0, 0)
      GO TO 20
C
C     PRESET TO NO SYMBOLS
   10 LINTYP = 0
C     IF SYMBOLS DESIRED, PUT ONE AT 1ST AND LAST POINT OF TRACE
      IF (ISYM .NE. 0)  LINTYP = LENDAT -1
C     DRAW THE TRACE
      CALL LINE(VLIST(LIMV1), FLIST(LIMV1), LENDAT, 1, LINTYP,ITHOUT-1)
C     JUMP IF NOT LABELING THE TRACES
      IF (ISYM .EQ. 0)  GO TO 20
C     VERTICAL LOCATION TO WRITE PARAMETER VALUE VS. SYMBOL
      YPOS = FLEN - 2.*HT*(ITHOUT+1)
      IF (YPOS .LT. 0.)  GO TO 20
      CALL SYMBOL(VLEN, YPOS, HT, PVALS(ITHOUT), 0., 10)
      CALL SYMBOL(VLEN+11.5*HT, YPOS+.5*HT, HT, ITHOUT-1, 0., -1)
      CALL PLOT(VLEN+10.5*HT, YPOS, 3)
      CALL PLOT(VLEN+10.5*HT, YPOS+2.*HT, 2)
C     RESTORE INDEP. VARIABLE VALUES CLOBBERED BY SCALE FACTORS
   20 VLIST(LIMVE+1) = SAVV1
      VLIST(LIMVE+2) = SAVV2
      RETURN
      END
      SUBROUTINE PPRINT
C     PRINT DATA FOR CURRENT PARAMETER VALUE
C
```

```
      COMMON // VLIST(2050), FLIST(2050), NOPROC, IPVAL, NBLOKS
     1,     IPPEND, JEDIT, MATCH, ISKIPP, LIMLO, LIMHI, PVAL, LENDAT
     2,     LIMV1, LIMVE, ITHOUT, HT, PVALS(1000), FMAXS(1000)
     3,     VFMAXS(1000), FMINS(1000), VFMINS(1000), FIS(1000)
     4,     SQFIS(1000), IPBUF(1024), RANGEP, FIRSTP, DELTAP
     5,     FIRSTV, DELTAV, FIRSTF, DELTAF
      COMMON //
     1      LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,     VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,     IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,     IDPP, NV, ISYM
     E,     ENDPP, IBLOKS(1)
C
C
C     SKIP IF NOT PRINTING SUMMARY
      IF (IPRINT .LT. 2)  GO TO 15
C     COLLECT AND SAVE STATISTICS FOR THIS PARAMETER VALUE
C     INITIALIZE INTEGRAL, SQUARE INTEGRAL, AND LOC OF MAX, MIN
      FI = 0.
      SQFI = 0.
      MAX = LIMV1
      MIN = LIMV1
C     LOOP THROUGH THAT PART OF DATA BEING PROCESSED
      LIM = LIMV1 + 1
      DO 10  I=LIM,LIMVE
C     PICK UP FUNCTION VALUE TO AVOID INDEXING
      F = FLIST(I)
C     LOC OF MAX, LOC OF MIN
      IF (FLIST(MAX) .LT. F)  MAX = I
      IF (FLIST(MIN) .GT. F)  MIN = I
      HAFDV = .5 * (VLIST(I)-VLIST(I-1))
      FI = FI + (F+FLIST(I-1)) *HAFDV
   10 SQFI = SQFI + (F**2+FLIST(I-1)**2) *HAFDV
C     SAVE EXTREMA INFORMATION
      FMAXS(ITHOUT) = FLIST(MAX)
      VFMAXS(ITHOUT) = VLIST(MAX)
      FMINS(ITHOUT) = FLIST(MIN)
      VFMINS(ITHOUT) = VLIST(MIN)
C     SAVE INTEGRAL, SQUARE INTEGRAL
      FIS(ITHOUT) = FI
      SQFIS(ITHOUT) = SQFI
C
C     JUMP IF NOT PRINTING ALL POINTS
   15 IF (IPRINT .NE. 1  .AND.  IPRINT .NE. 3)  GO TO 100
C     FIRST LINE OF PAGE
      WRITE(6,20) PVALS(ITHOUT),PNAME,TITLE,IDPP
   20 FORMAT(1H1,A10,1H=,A10,20X,2A10,20X,A10)
C     2 FORMATS.  JUMP IF PRINTING VARIABLE/FUNCTION
      IF (IVLIST .EQ. 3)  GO TO 60
C     JUST PRINTING FUNCTION
      WRITE(6,30) FNAME
   30 FORMAT(1H ,2A10/)
      DO 50  I1=LIMV1,LIMVE,8
      IE = MIN0(LIMVE,I1+7)
      WRITE(6,40) I1,(FLIST(I),I=I1,IE)
   40 FORMAT(1H ,I4,8E13.5)
   50 CONTINUE
      GO TO 100
C
   60 WRITE(6,70) VNAME,FNAME
   70 FORMAT(1H ,A10,1H ,2A10/)
      DO 90  I1=LIMV1,LIMVE,4
```

```
      IE = MINO(LIMVE,I1+3)
      WRITE(6,80) I1,(VLIST(I),FLIST(I),I=I1,IE)
   80 FORMAT(1H ,I4,4(E14.4,E12.4))
   90 CONTINUE
C
  100 RETURN
      END
      SUBROUTINE PPSET
C     SCALING, SETUPS FOR THIS PP SET.  DRAW AXES IF PLOTTING
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,     NTDTAB, NTEVEC, NTTEMP
C
      COMMON /PRTPLT/ IPLTON, XAORG, YAORG, XPORG, YPORG
C
      COMMON // VLIST(2050), FLIST(2050), NOPROC, IPVAL, NBLOKS
     1,     IPPEND, JEDIT, MATCH, ISKIPP, LIMLO, LIMHI, PVAL, LENDAT
     2,     LIMV1, LIMVE, ITHOUT, HT, PVALS(1000), FMAXS(1000)
     3,     VFMAXS(1000), FMINS(1000), VFMINS(1000), FIS(1000)
     4,     SQFIS(1000), IPBUF(1024), RANGEP, FIRSTP, DELTAP
     5,     FIRSTV, DELTAV, FIRSTF, DELTAF
      COMMON //
     1      LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,     VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,     IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,     IDPP, NV, ISYM
     E,     ENDPP, IBLOKS(1)
C
C
C     PRESET TO PROCESS THIS PP SET
      ISKIPP = 0
C     INITIALIZE COUNT OF NUMBER OF RECORDS OUTPUT
      ITHOUT = 0
C     JUMP IF NOTHING WOULD BE ACCOMPLISHED BY PROCESSING THIS DATA
      IF (IPRINT .EQ. 0   .AND.  IPLOT .EQ. 0)  GO TO 200
C     JUMP IF EDITING IS ON (PP ONLY DATA FOR WHICH A PPFB WAS INPUT)
C     AND NO PPFB WAS INPUT
      IF (JEDIT .NE. 0   .AND.  MATCH .EQ. 0)  GO TO 200
C
C     SKIP IF VLIST IS NOT FIXED
      IF (IVLIST .GT. 2)  GO TO 40
C     FIND FWA AND LWA OF DATA WITHIN RANGE VMIN TO VMAX
      CALL PPVLIM(ISKIPP)
      LIMLO = LIMV1
      LIMHI = LIMVE
C     JUMP IF NO DATA WITHIN THAT RANGE
      IF (ISKIPP .NE. 0)  GO TO 200
C     SKIP IF NOT PRINTING
      IF (IPRINT .NE. 1   .AND.  IPRINT .NE. 3)  GO TO 40
C     PRINT THE LIST OF INDEPENDENT VARIABLE VALUES
      WRITE(6,10) TITLE,IDPP,VNAME
   10 FORMAT(6H1*****,36X,2A10,20X,A10/1H ,A10/)
      DO 30  I1=LIMV1,LIMVE,8
      IE = MINO(LIMVE,I1+7)
      WRITE(6,20) I1,(VLIST(I),I=I1,IE)
   20 FORMAT(1H ,I4,8E13.5)
   30 CONTINUE
C
C     SKIP IF NOT PLOTTING THIS DATA
   40 IF (IPLOT .EQ. 0)  GO TO 100
C     CHARACTER HEIGHT IN INCHES
      HT = .105
```

```
C     SKIP IF PLOTTING HAS ALREADY BEEN OPENED
      IF (IPLTON .NE. 0)  GO TO 50
C     ONCE PER RUN INITIALIZATIONS
      CALL PLOTS(IPBUF, 1024, NTPLOT)
C     SHOW PLOTTING HAS BEEN INITIATED
      IPLTON = 1
C     LOCATION OF ORIGIN OF AXES (IE LOWER LEFT CORNER OF PLOT)
C     WRT LL CORNER OF PAGE
      XAORG = 0.
      YAORG = 8.*HT
C     MOVE PEN ORG TO AXES ORG
      CALL PLOT(XAORG, YAORG, -3)
C     LOC OF PEN ORG WRT AXES ORG
      XPORG = 0.
      YPORG = 0.
C
C     SET SCALING FOR INDEPENDENT VARIABLE
   50 FIRSTV = VMIN
      DELTAV = (VMAX-VMIN)/VLEN
C     JUMP FOR MULTI-TRACE PLOT
      IF (IPLTYP .NE. 0)  GO TO 60
C     RASTER
      IF (PMAX .NE. PMIN)  GO TO 55
C     RASTER PLOT WILL BLOW.  SWITCH TO MULTI-TRACE
      IPLTYP = 1
      GO TO 60
C     PARAMETER SCALING
   55 FIRSTP = PMIN
      DELTAP = (PMAX-PMIN)/PLEN
C     FUNCTION SCALING
      FIRSTF = 0.
      IF (FTODP .NE. 0.  .AND.  NP .GT. 1) FLEN=FTODP*PLEN/FLOAT(NP-1)
     1                                    *RANGEP/(PMAX-PMIN)
C     DONT EVER LET THE LENGTH OF THE FUNCTION AXIS EXCEED THE PLOT SIZE
      IF (ABS(FLEN) .GT. ABS(PLEN))  FLEN = SIGN(PLEN,FLEN)
      ABSMXF = AMAX1(ABS(FMAX), ABS(FMIN))
      DELTAF = ABSMXF/FLEN
      IF (FLEN .GT. 0.)  GO TO 70
C     POSITIVE DIRECTION FOR F OPPOSITE FROM USUAL.  ADJUST FOR
C     DRAWING AXIS
      FLEN = -FLEN
      GO TO 70
C
C     SCALE FOR M.T. PLOT
   60 FIRSTF = FMIN
      DELTAF = (FMAX-FMIN)/FLEN
C
C     DRAW PLOT TITLE
   70 IF (TITLE(1) .NE. 1H ) CALL SYMBOL(0., -YAORG, HT, TITLE, 0., 20)
C     SKIP FOR M.T. PLOT
      IF (IPLTYP .NE. 0)  GO TO 80
C     RASTER.  DRAW F AXIS
      CALL PPAXIS(0., -3.*HT, FNAME, -20, FLEN, 0., FIRSTF, DELTAF)
C     DRAW P AXIS, THEN VARIABLE AXIS
      CALL PPAXIS(0., 0., PNAME, -10, PLEN, 0., FIRSTP, DELTAP)
      CALL PPAXIS(0., 0., VNAME, 10, VLEN, 90., FIRSTV, DELTAV)
      GO TO 90
C
C     MULTI-TRACE.  DRAW V AXIS, THEN FUNCTION AXIS
   80 CALL PPAXIS(0., 0., VNAME, -10, VLEN, 0., FIRSTV, DELTAV)
      CALL PPAXIS(0., 0., FNAME, 20, FLEN, 90., FIRSTF, DELTAF)
C     SKIP IF NOT LABELING EACH TRACE
```

```
      IF (ISYM .EQ. 0)  GO TO 90
C     DRAW PARAMETER-KEY HEADER
      CALL SYMBOL(VLEN, FLEN, HT, PNAME, 0., 10)
      CALL SYMBOL(VLEN+11.*HT, FLEN, HT, 3HSYM, 0., 4)
      CALL PLOT(VLEN+14.*HT, FLEN-.5*HT, 3)
      CALL PLOT(VLEN, FLEN-.5*HT, 2)
      CALL PLOT(VLEN+10.5*HT, FLEN+HT, 3)
      CALL PLOT(VLEN+10.5*HT, FLEN-2.*HT, 2)
   90 CONTINUE
  100 RETURN
C
C     DO NOT PROCESS THIS PP SET
  200 ISKIPP = 1
      RETURN
      END
      SUBROUTINE PPSPEC
C     READ ALL THE PP SPECIFICATIONS WHICH WERE INPUT FOR THIS CASE
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,     NTDTAB, NTEVEC, NTTEMP
C
      COMMON // VLIST(2050), FLIST(2050), NOPROC, IPVAL, NBLOKS
     1,     IPPEND, JEDIT, MATCH, ISKIPP, LIMLO, LIMHI, PVAL, LENDAT
     2,     LIMV1, LIMVE, ITHOUT, HT, PVALS(1000), FMAXS(1000)
     3,     VFMAXS(1000), FMINS(1000), VFMINS(1000), FIS(1000)
     4,     SQFIS(1000), IPBUF(1024), RANGEP, FIRSTP, DELTAP
     5,     FIRSTV, DELTAV, FIRSTF, DELTAF
      COMMON //
     1      LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,     VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,     IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,     IDPP, NV, ISYM
     E,     ENDPP, IBLOKS(1)
      DIMENSION IEDIT(1),NOPP(1),IDPP(1),IOCUR(1)
C
C
      IPPEND = 0
      JEDIT = 0
      LENPP = LOCF(ENDPP) - LOCF(LENPP)
      REWIND NTPDEF
C     NUMBER OF BLOCKS AND LENGTH OF EACH
      READ(NTPDEF) NBLOKS,LENBLK
      IF (NBLOKS .EQ. 0)  GO TO 50
      IF (LENBLK .EQ. LENPP)  GO TO 20
      WRITE(6,10) LENPP,LENBLK
   10 FORMAT(49H PP DEFINITION FILE FORMAT INCONSISTENT WITH PROG
     1        ,6H SPECS,2I10)
      CALL ERRXIT
C     READ ALL INPUT BLOCKS
   20 LIM1 = 1
      DO 30  ITHBLK=1,NBLOKS
      LIM2 = LIM1 +LENPP -1
      READ(NTPDEF) (IBLOKS(I),I=LIM1,LIM2)
   30 LIM1 = LIM1 + LENPP
C     IEDIT-- 0=P/P ALL DATA WRITTEN, 1=P/P ONLY DATA FOR WHICH
C     THERE IS AN INPUT BLOCK.   SET JEDIT=1 IF ANY BLOCK SHOWS IEDIT=1
C     NOPP-- 0=REWIND DATA, ID FILES AND PROCEED TO P/P, 1=NO REWIND,
C     NO P/P.
      LOC = 1
      DO 40  ITHBLK=1,NBLOKS
      LOC = LOC + LENPP
      IF (NOPP(LOC) .EQ. 1)  GO TO 60
```

```
   40 IF (IEDIT(LOC) .EQ. 1)  JEDIT = 1
   50 REWIND NTPID
      REWIND NTPDAT
      RETURN
C
   60 IPPEND = 1
      RETURN
      END
      SUBROUTINE PPSUM
C     PRINT SUMMARY DATA
C
      COMMON // VLIST(2050), FLIST(2050), NOPROC, IPVAL, NBLOKS
     1,      IPPEND, JEDIT, MATCH, ISKIPP, LIMLO, LIMHI, PVAL, LENDAT
     2,      LIMV1, LIMVE, ITHOUT, HT, PVALS(1000), FMAXS(1000)
     3,      VFMAXS(1000), FMINS(1000), VFMINS(1000), FIS(1000)
     4,      SQFIS(1000), IPBUF(1024), RANGEP, FIRSTP, DELTAP
     5,      FIRSTV, DELTAV, FIRSTF, DELTAF
      COMMON //
     1       LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,      VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,      IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,      IDPP, NV, ISYM
     E,      ENDPP, IBLOKS(1)
C
C
      WRITE(6,10) FNAME,TITLE,IDPP
   10 FORMAT(10H1FUNCTION=,2A10,16X,2A10,20X,A10)
C
      WRITE(6,20) VNAME,VNAME,PNAME
   20 FORMAT(34H0       VALUE OF      MAXIMUM OF     ,A10,6X
     1,     14HMINIMUM OF     ,A10,20X,6HSQUARE/[illegible],A10,3X,8HFUNCTION
     2,     6X,6HOF MAX,10X,8HFUNCTION,6X,6HOF MIN,11X,8HINTEGRAL
     3,     5X,8HINTEGRAL)
C
      WRITE(6,30) (I,PVALS(I),FMAXS(I),VFMAXS(I),FMINS(I),VFMINS(I),
     1              FIS(I),SQFIS(I),I=1,ITHOUT)
   30 FORMAT(1H ,I4,A10,4E15.6,E15.4,E13.4)
      RETURN
      END
      SUBROUTINE PPTDEF
C     READ PPFB FOR NEXT PP DATA SET FROM PROGRAM TAPE,
C     SET UP VLIST IF APPROPRIATE
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,      NTDTAB, NTEVEC, NTTEMP
C
      COMMON // VLIST(2050), FLIST(2050), NOPROC, IPVAL, NBLOKS
     1,      IPPEND, JEDIT, MATCH, ISKIPP, LIMLO, LIMHI, PVAL, LENDAT
     2,      LIMV1, LIMVE, ITHOUT, HT, PVALS(1000), FMAXS(1000)
     3,      VFMAXS(1000), FMINS(1000), VFMINS(1000), FIS(1000)
     4,      SQFIS(1000), IPBUF(1024), RANGEP, FIRSTP, DELTAP
     5,      FIRSTV, DELTAV, FIRSTF, DELTAF
      COMMON //
     1       LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,      VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,      IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,      IDPP, NV, ISYM
     E,      ENDPP, IBLOKS(1)
      DIMENSION IDBLOK(1),DUMMY(1)
      EQUIVALENCE (DUMMY,LENPP),(IDBLOK,DUMMY(2))
C
C
```

```
C     READ IN PPFB FOR NEXT PP FROM PROGRAM TAPE
      READ(NTPID) LEN,(IDBLOK(I),I=1,LEN)
      IF (EOF,NTPID)  10,20
C     EOF ENCOUNTERED, TERMINATE PP PROCESSOR
   10 IPPEND = 1
      RETURN
C     THERE IS ANOTHER PP SET.  SET FLAG TO CONTINUE PROCESSING
   20 IPPEND = 0
C     SET ACTUAL RANGE OF PARAMETER (RANGE TO PP MAY DIFFER)
      RANGEP = PMAX -PMIN
C     SET UP LIST OF INDEPENDENT VARIABLE VALUES
      GO TO (30,50,60),IVLIST
C     LIST OF VALUES IS SAME FOR EACH PARAMETER VALUE AND IS
C     EQUAL INCREMENT FROM VMIN TO VMAX
   30 DV = (VMAX-VMIN) / FLOAT(NV-1)
      VLIST(1) = VMIN
      DO 40   I=2,NV
   40 VLIST(I) = VLIST(I-1) + DV
      GO TO 55
C     LIST OF VALUES IS SAME FOR EACH PARAMETER VALUE
C     SPACING IS ARBITRARY
   50 READ(NTPID) (VLIST(I),I=1,NV)
C     SET FWA AND LWA OF VLIST
   55 LIMLO = 1
      LIMHI = NV
      RETURN
C     VLIST VARIES WITH PARAMETER VALUES AND IS ON PP DATA FILE
C     ALONG WITH FUNCTION VALUES
   60 RETURN
      END
      SUBROUTINE PPVLIM(NODATA)
C     FIND FWA AND LWA OF DATA IN RANGE VMIN TO VMAX
C
      COMMON // VLIST(2050), FLIST(2050), NOPROC, IPVAL, NBLOKS
     1,      IPPEND, JEDIT, MATCH, ISKIPP, LIMLO, LIMHI, PVAL, LENDAT
     2,      LIMV1, LIMVE, ITHOUT, HT, PVALS(1000), FMAXS(1000)
     3,      VFMAXS(1000), FMINS(1000), VFMINS(1000), FIS(1000)
     4,      SQFIS(1000), IPBUF(1024), RANGEP, FIRSTP, DELTAP
     5,      FIRSTV, DELTAV, FIRSTF, DELTAF
      COMMON //
     1       LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,      VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,      ICCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,      IDPP, NV, ISYM
     E,      ENDPP, IBLOKS(1)
C
C
C     JUMP IF VARIABLE DECREASES
      IF (VLIST(LIMHI) .LT. VLIST(1))  GO TO 60
      DO 30  LIMV1=1,LIMHI
   30 IF (VMIN .LE. VLIST(LIMV1))  GO TO 40
      GO TO 110
   40 DO 50  I=1,LIMHI
      LIMVE = LIMHI-I+1
   50 IF (VMAX .GE. VLIST(LIMVE))  GO TO 100
      GO TO 110
C
   60 DO 70  LIMV1=1,LIMHI
   70 IF (VMAX .GE. VLIST(LIMV1))  GO TO 80
      GO TO 110
   80 DO 90  I=1,LIMHI
      LIMVE = LIMHI-I+1
```

```
   40 IF (IEDIT(LOC) .EQ. 1)  JEDIT = 1
   50 REWIND NTPID
      REWIND NTPDAT
      RETURN
C
   60 IPPEND = 1
      RETURN
      END
      SUBROUTINE PPSUM
C     PRINT SUMMARY DATA
C
      COMMON // VLIST(2050), FLIST(2050), NOPROC, IPVAL, NBLOKS
     1,      IPPEND, JEDIT, MATCH, ISKIPP, LIMLO, LIMHI, PVAL, LENDAT
     2,      LIMV1, LIMVE, ITHOUT, HT, PVALS(1000), FMAXS(1000)
     3,      VFMAXS(1000), FMINS(1000), VFMINS(1000), FIS(1000)
     4,      SQFIS(1000), IPBJF(1024), RANGEP, FIRSTP, DELTAP
     5,      FIRSTV, DELTAV, FIRSTF, DELTAF
      COMMON //
     1       LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,      VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,      IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,      IDPP, NV, ISYM
     E,      ENDPP, IBLOKS(1)
C
C
      WRITE(6,10) FNAME,TITLE,IDPP
   10 FORMAT(10H1FUNCTION=,2A10,16X,2A10,20X,A10)
C
      WRITE(6,20) VNAME,VNAME,PNAME
   20 FORMAT(34H0      VALUE OF     MAXIMUM OF    ,A10,6X
     1,     14HMINIMUM OF    ,A10,20X,6HSQUARE/7X,A10,3X,8HFUNCTION
     2,     6X,6HOF MAX,10X,8HFUNCTION,6X,6HOF MIN,11X,8HINTEGRAL
     3,     5X,8HINTEGRAL)
C
      WRITE(5,30) (I,PVALS(I),FMAXS(I),VFMAXS(I),FMINS(I),VFMINS(I),
     1            FIS(I),SQFIS(I),I=1,ITHOUT)
   30 FORMAT(1H ,I4,A10,4E15.6,E15.4,E13.4)
      RETURN
      END
      SUBROUTINE PPTDEF
C     READ PPFB FOR NEXT PP DATA SET FROM PROGRAM TAPE,
C     SET UP VLIST IF APPROPRIATE
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,      NTDTAB, NTEVEC, NTTEMP
C
      COMMON // VLIST(2050), FLIST(2050), NOPROC, IPVAL, NBLOKS
     1,      IPPEND, JEDIT, MATCH, ISKIPP, LIMLO, LIMHI, PVAL, LENDAT
     2,      LIMV1, LIMVE, ITHOUT, HT, PVALS(1000), FMAXS(1000)
     3,      VFMAXS(1000), FMINS(1000), VFMINS(1000), FIS(1000)
     4,      SQFIS(1000), IPBJF(1024), RANGEP, FIRSTP, DELTAP
     5,      FIRSTV, DELTAV, FIRSTF, DELTAF
      COMMON //
     1       LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,      VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,      IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,      IDPP, NV, ISYM
     E,      ENDPP, IBLOKS(1)
      DIMENSION IDBLCK(1),DUMMY(1)
      EQUIVALENCE (DUMMY,LENPP),(IDBLOK,DUMMY(2))
C
C
```

```
C     READ IN PPFB FOR NEXT PP FROM PROGRAM TAPE
      READ(NTPID) LEN,(IDBLOK(I),I=1,LEN)
      IF (EOF,NTPID)  10,20
C     EOF ENCOUNTERED, TERMINATE PP PROCESSOR
   10 IPPEND = 1
      RETURN
C     THERE IS ANOTHER PP SET.  SET FLAG TO CONTINUE PROCESSING
   20 IPPEND = 0
C     SET ACTUAL RANGE OF PARAMETER (RANGE TO PP MAY DIFFER)
      RANGEP = PMAX -PMIN
C     SET UP LIST OF INDEPENDENT VARIABLE VALUES
      GO TO (30,50,60),IVLIST
C     LIST OF VALUES IS SAME FOR EACH PARAMETER VALUE AND IS
C     EQUAL INCREMENT FROM VMIN TO VMAX
   30 DV = (VMAX-VMIN) / FLOAT(NV-1)
      VLIST(1) = VMIN
      DO 40  I=2,NV
   40 VLIST(I) = VLIST(I-1) + DV
      GO TO 55
C     LIST OF VALUES IS SAME FOR EACH PARAMETER VALUE
C     SPACING IS ARBITRARY
   50 READ(NTPID) (VLIST(I),I=1,NV)
C     SET FWA AND LWA OF VLIST
   55 LIMLO = 1
      LIMHI = NV
      RETURN
C     VLIST VARIES WITH PARAMETER VALUES AND IS ON PP DATA FILE
C     ALONG WITH FUNCTION VALUES
   60 RETURN
      END
      SUBROUTINE PPVLIM(NODATA)
C     FIND FWA AND LWA OF DATA IN RANGE VMIN TO VMAX
C
      COMMON // VLIST(2050), FLIST(2050), NOPROC, IPVAL, NBLOKS
     1,     IPPEND, JEDIT, MATCH, ISKIPP, LIMLO, LIMHI, PVAL, LENDAT
     2,     LIMV1, LIMVE, ITHOUT, HT, PVALS(1000), FMAXS(1000)
     3,     VFMAXS(1000), FMINS(1000), VFMINS(1000), FIS(1000)
     4,     SQFIS(1000), IPBUF(1024), RANGEP, FIRSTP, DELTAP
     5,     FIRSTV, DELTAV, FIRSTF, DELTAF
      COMMON //
     1      LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,     VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,     ICCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,     IDPP, NV, ISYM
     E,     ENDPP, IBLOKS(1)
C
C
C     JUMP IF VARIABLE DECREASES
      IF (VLIST(LIMHI) .LT. VLIST(1))  GO TO 60
      DO 30  LIMV1=1,LIMHI
   30 IF (VMIN .LE. VLIST(LIMV1))  GO TO 40
      GO TO 110
   40 DO 50  I=1,LIMHI
      LIMVE = LIMHI-I+1
   50 IF (VMAX .GE. VLIST(LIMVE))  GO TO 100
      GO TO 110
C
   60 DO 70  LIMV1=1,LIMHI
   70 IF (VMAX .GE. VLIST(LIMV1))  GO TO 80
      GO TO 110
   80 DO 90  I=1,LIMHI
      LIMVE = LIMHI-I+1
```

```
   90 IF (VMIN .LE. VLIST(LIMVE))  GO TO 100
      GO TO 110
C
C     AMOUNT OF DATA TO BE PROCESSED
  100 LENDAT = LIMVE-LIMV1+1
C     JUMP IF NO DATA.  ALSO DONT TRY TO PROCESS JUST 1 DATA POINT
      IF (LENDAT .LE. 1)  GO TO 110
      NODATA = 0
      RETURN
C     NO DATA WITHIN SPECIFIED RANGE VMIN TO VMAX
  110 NODATA = 1
      RETURN
      END
      SUBROUTINE SPCON
C     STATIONARY PHASE CONTROL
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,      RBSEP2, RBSTR, RBLIM
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,      X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,      IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,      ISPHAS
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,      SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMMON /WAKE/ IWAKE, CWAKR, CWAKX, XWAKE, WAKRAD, XWNOM
     1,      RESLVS, CWAKM
C
      COMPLEX XDEP, FSRC
      COMMON // RK, EVAL, DLDK, D2L, PSIO, DPSIO, DPSIB, WAKI, SUPT
     1,      YX, D3L, XI, ETA, INRANG, MODE, MINMOD, MAXMOD, MODES
     2,      MAXK, IFWA1, LOCDT, LOCDTK, ITHWAV, NWAVES, NWFTAB(41)
     3,      HILO, DYX, PYX, PDYX, PEVAL, PRK, FAZ, SGNFZ2, FAZ2
     4,      FAZ3, XDEP, FSRC, VAR, SIGNAL, SIGCUT(800)
      EQUIVALENCE (TEMP1,SIGCUT)
      DIMENSION TEMP1(9,1)
C
C
C     ****SUMMARY OF APPROACH****
C     TOTAL SIGNAL IS SIGTOT=SUMOVERMODES(SUMOVERWAVEFAMILIES(V*SX))
C     WHERE V DEPENDS ONLY ON THE VARIABLE (SIGNAL) BEING COMPUTED
C     AND SX = RE(S*XD) OR SX=IM(S*XD).  HERE S DEPENDS ONLY ON THE
C     SOURCE MODEL AND XD CONTAINS THE X DEPENDENCE.
C
      CALL TIMER(14)
C     SKIP IF NOT 1ST PASS OF CASE
      IF (ITHOBS .NE. 1)  GO TO 10
C     COMPUTE BODY SOURCE PARAMETERS IF BODY MODEL USED
      IF (IBODY .NE. 0)  CALL BODY1
C     SUPERSTRUCTURE SOURCE PARAMETERS
      IF (ISUPR .NE. 0)  CALL SUPR1
C     WAKE SOURCE PARAMETER
C     START OF WAKE COLLAPSE=INPUT MULTIPLIER*NOMINAL START (SEE DTWAKE)
      IF (IWAKE .NE. 0)  XWAKE = CWAKX*XWNOM
C
C     READ IN DISPERSION TABLES, SET UP WAVE FAMILY EDGE TABLES
   10 CALL SPDTAB
C     PRINT/PLOT DISPERSION RELATION ON 1ST PASS
      IF (ITHOBS .EQ. 1)  CALL SPDTPP
C
```

```
C     SKIP IF NO GRID DEFINED
      IF (NY .LT. 1  .OR.  NX .LT. 1)  GO TO 40
C     LOOP FOR EACH X (DOWNSTREAM STATION)
      DO 30  ITHX=1,NX
      X = XMIN + DX*FLOAT(ITHX-1)
C     LOOP FOR EACH Y (TRANSVERSE COORDINATE)
      DO 20  ITHY=1,NY
C     NOTE Y IS ALWAYS NEGATIVE
      Y = -(YMIN + DY*FLOAT(ITHY-1))
      YX = -Y/X
C     COMPUTE SIGNAL AT POINT DEFINED BY X, YX, OBSDEP
      CALL SPIPNT
C     ACCUMULATE AND OUTPUT (TO PP PROCESSOR) SIGNAL DATA
      CALL SPCUTS
   20 CONTINUE
   30 CONTINUE
   40 CALL TIMER(-14)
      RETURN
      END
      SUBROUTINE SPCUTS
C     INITIALIZE PP FORMAT, ACCUMULATE AND OUTPUT DATA FOR EACH CUT,
C     AND WRITE PP ID BLOCK.  THREE CASES ARE HANDLED-- X-Y GRID,
C     Z-Y GRID, Z-X GRID
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,     X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,     IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,     ISPHAS
C
      COMMON /NAME/ NAMES(2,10), DTNAMS(2,9)
      COMMON/PPCOM/
     1      LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,     VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,     IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,     IDPP, NV, ISYM
     E,     ENDPP, IBLOKS(1)
C
      COMPLEX XDEP, FSRC
      COMMON // RK, EVAL, DLDK, D2L, PSIO, DPSIO, DPSIB, WAKI, SUPT
     1,     YX, D3L, XI, ETA, INRANG, MODE, MINMOD, MAXMOD, MODES
     2,     MAXK, IFWA1, LOCDT, LOCDTK, ITHWAV, NWAVES, NWFTAB(41)
     3,     HILO, DYX, PYX, PDYX, PEVAL, PRK, FAZ, SGNFZ2, FAZ2
     4,     FAZ3, XDEP, FSRC, VAR, SIGNAL, SIGCUT(800)
      EQUIVALENCE (TEMP1,SIGCUT)
      DIMENSION TEMP1(9,1)
C
C
C     SKIP FOR Z-X GRID
      IF (NY .EQ. 1)  GO TO 30
C     X-Y OR Z-Y GRID.  SAVE SIGNAL AT CURRENT Y VALUE
      SIGCUT(ITHY) = SIGNAL
C     RETURN IF Y CUT NOT COMPLETED
      IF (ITHY .LT. NY)  RETURN
C     SKIP FOR Z-Y GRID
      IF (NOBS .GT. 1)  GO TO 20
C     X-Y GRID.  INITIALIZE PP FORMAT BEFORE WRITING 1ST CROSS CUT
      IF (ITHX .EQ. 1)  CALL SETID(4HCUTS, 0, 1HX, 2H-Y, NAMES(1,IVAR))
C     WRITE CROSS CUT
      CALL WRTDAT(1, NY, SIGCUT, 1, X)
C     RETURN IF NOT LAST PASS
      IF (ITHX .LT. NX)  RETURN
C     WRITE PP ID BLOCK
```

```
      VMIN = YMIN
      VMAX = -Y
      CALL WRTID(NY, 0,0)
      RETURN
C
C     Z-Y GRID.  INITIALIZE PP FORMAT BEFORE WRITING 1ST CROSS CUT
   20 IF (ITHOBS .EQ. 1)
     1           CALL SETID(4HCUTS, 0, 5HDEPTH, 2H-Y, NAMES(1,IVAR))
C     WRITE CROSS CUT
      CALL WRTDAT(1, NY, SIGCUT, 1, OBSDEP)
C     RETURN IF NOT LAST PASS
      IF (ITHOBS .LT. NOBS)  RETURN
      VMIN = YMIN
      VMAX = -Y
C     MAKE FUNCTION POSITIVE TO THE LEFT
      FTODP = -FTODP
      CALL WRTID(NY, 0,0)
      RETURN
C
C     Z-X GRID.  SAVE SIGNAL AT CURRENT X STATION
   30 SIGCUT(ITHX) = SIGNAL
C     RETURN IF X CUT NOT COMPLETED
      IF (ITHX .LT. NX)  RETURN
C     INITIALIZE PP FORMAT BEFORE WRITING 1ST CUT
      IF (ITHOBS .EQ. 1)
     1           CALL SETID(4HCUTS, 0, 5HDEPTH, 1HX, NAMES(1,IVAR))
C     WRITE AXIAL CUT
      CALL WRTDAT(1, NX, SIGCUT, 1, OBSDEP)
C     RETURN IF NOT LAST PASS
      IF (ITHOBS .LT. NOBS)  RETURN
C     MAKE FUNCTION POSITIVE TO THE LEFT
      FTODP = -FTODP
C     WRITE PP ID BLOCK
      VMIN = XMIN
      VMAX = X
      CALL WRTID(NX, 0,0)
      RETURN
      END
      SUBROUTINE SPDISP(JEDGE)
C     LINEAR INTERP FOR DISPERSION VARIABLES AS FUNCTION OF EVAL
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,     RBSEP2, RBSTR, RBLIM
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,     SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMMON /WAKE/ IWAKE, CWAKR, CWAKX, XWAKE, WAKRAD, XWNOM
     1,     RESLVS, CWAKM
C
      COMPLEX XDEP, FSRC
      COMMON // RK, EVAL, DLDK, D2L, PSIO, DPSIO, DPSIB, WAKI, SUPT
     1,     YX, D3L, XI, ETA, INRANG, MODE, MINMOD, MAXMOD, MODES
     2,     MAXK, IFWA1, LOCDT, LOCDTK, ITHWAV, NWAVES, NWFTAB(41)
     3,     HILO, DYX, PYX, PDYX, PEVAL, PRK, FAZ, SGNFZ2, FAZ2
     4,     FAZ3, XDEP, FSRC, VAR, SIGNAL, SIGCUT(600)
      EQUIVALENCE (TEMP1,SIGCUT)
      DIMENSION TEMP1(9,1)
      COMMON // TK(100), TEVAL(100,41), TDLDK(100,41), TD2L(100,41)
     1,     TPSIO(100,41), TDPSIO(100,41), TDPSIB(100,41), TWAKI(100,41)
     2,     TSUPT(100,41), TYX(100,41), YXEDG(20,41), EVLEDG(20,41)
     3,     LIMEDG(20,41), INDEDG(20,41)
```

```
C
C
C     SKIP IF ROUTINE IS BEING USED TO FIND WAVE FAMILY EDGES
      IF (JEDGE .NE. 0)  GO TO 10
C     CONVERT YX TO EVAL
      CALL SPEVAL
C     RETURN IF YX IS OUTSIDE RANGE OF SPECIFIED WAVE FAMILY
      IF (INRANG .EQ. 0)  RETURN
C     LOCDT IS SUCH THAT TEVAL(LOCDT-1) .LT. EVAL .LE. TEVAL(LOCDT)
C     LOCDTK IS THE CORRESPONDING INDEX FOR THE K (WAVENUMBER) LIST
C     GET LINEAR INTERP COEFFICIENTS
   10 C2 = (EVAL-TEVAL(LOCDT-1)) / (TEVAL(LOCDT)-TEVAL(LOCDT-1))
      C1 = 1. - C2
C     INTERPOLATE FOR K, D(EVAL)/DK, D2(EVAL)/DK2 WHERE EVAL=1/C**2
      RK = C1*TK(LOCDTK-1) + C2*TK(LOCDTK)
      DLDK = C1*TDLDK(LOCDT-1) + C2*TDLDK(LOCDT)
      D2L = C1*TD2L(LOCDT-1) + C2*TD2L(LOCDT)
C     THE ABOVE VARIABLES ARE SUFFICIENT FOR FINDING WAVE FAMILY
C     EDGES.  RETURN IF THAT IS WHAT THE ROUTINE IS BEING USED FOR.
      IF (JEDGE .NE. 0)  RETURN
C     COMPLETE THE DISPERSION RELATION
C     NORMALIZED EIGENFUNCTION PSI AND D(PSI)/DZ AT OBSERVATION DEPTH
      PSIO = C1*TPSIO(LOCDT-1) + C2*TPSIO(LOCDT)
      DPSIO = C1*TDPSIO(LOCDT-1) + C2*TDPSIO(LOCDT)
C     D(PSI)/DZ AT BODY DEPTH
      IF (IBODY .NE. 0)  DPSIB = C1*TDPSIB(LOCDT-1) + C2*TDPSIB(LOCDT)
C     WAKE SOURCE TERM
      IF (IWAKE .NE. 0)  WAKI = C1*TWAKI(LOCDT-1) + C2*TWAKI(LOCDT)
C     SUPERSTRUCTURE TERM = PSI(BOTTOM OF SUPER) - PSI(TOP OF SUPER)
      IF (ISUPR .NE. 0)  SUPT = C1*TSUPT(LOCDT-1) + C2*TSUPT(LOCDT)
C     COMPUTE D3(EVAL)/DK3 = D(D2L)/DK
      D3L = (TD2L(LOCDT)-TD2L(LOCDT-1)) / (TK(LOCDTK)-TK(LOCDTK-1))
      RETURN
      END
      SUBROUTINE SPDTAB
C     READ DISPERSION TABLE AND PERFORM FINAL ADJUSTMENTS ON IT
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,     RBSEP2, RBSTR, RBLIM
C
      COMMON /CONST/ JDK, JDMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,     JDCKSV, JDMSP, JDEDGE
C
      COMMON /FILES/ NTILIB, NTDLIB, NTPDEF, NTPID, NTPDAT, NTPLOT
     1,     NTDTAB, NTEVEC, NTTEMP
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,     X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,     IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,     ISPHAS
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,     SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMMON /WAKE/ IWAKE, CWAKR, CWAKX, XWAKE, WAKRAD, XWNOM
     1,     RESLVS, CWAKM
C
      COMPLEX XDEP, FSRC
      COMMON // RK, EVAL, DLDK, D2L, PSIO, DPSIO, DPSIB, WAKI, SUPT
     1,     YX, D3L, XI, ETA, INRANG, MODE, MINMOD, MAXMOD, MODES
     2,     MAXK, IFWA1, LOCDT, LOCDTK, ITHWAV, NWAVES, NWFTAB(41)
     3,     HILO, CYX, PYX, PDYX, PEVAL, PRK, FAZ, SGNFZ2, FAZ2
```

```
     4,     FAZ3, XDEP, FSRC, VAR, SIGNAL, SIGCUT(800)
      EQUIVALENCE (TEMP1,SIGCUT)
      DIMENSION TEMP1(9,1)
      COMMON // TK(100), TEVAL(100,41), TDLDK(100,41), TD2L(100,41)
     1,     TPSIO(100,41), TDPSIO(100,41), TDPSIB(100,41), TWAKI(100,41)
     2,     TSUPT(100,41), TYX(100,41), YXEDG(20,41), EVLEDG(20,41)
     3,     LIMEDG(20,41), INDEDG(20,41)
C
C
      CALL TIMER(15)
C     MODE1 AND MODEN ARE INPUT LIMITS OF DESIRED MODES.  SET UP
C     INTERNAL STORAGE AND DO-LOOP LIMITS TO SQUEEZE OUT UNUSED MODES
      MINMOD = 1
      MAXMOD = MODEN -MODE1 +1
C     CHECK MODE RANGE AGAINST AVAILABLE STORAGE
      IF (MAXMOD .LE. JDMSP)  GO TO 20
      WRITE(6,10) MODE1,MODEN,JDMSP
   10 FORMAT(29H MODE RANGE EXCEEDS DIMENSION,3I10)
      CALL ERRXIT
C     DISPERSION TABLE IS ON TAPE NTDTAB
   20 REWIND NTDTAB
      READ(NTDTAB) MODES
C     SKIP IF DISP TABLE HAS AT LEAST AS MANY MODES AS DESIRED
      IF (MODEN .LE. MODES)  GO TO 40
      WRITE(6,30) MODEN,MODES
   30 FORMAT(20H MODEN EXCEEDS MODES,2I10)
      CALL ERRXIT
C     LOOP FOR EACH ENTRY (VALUE OF K) IN DISP TABLE, BUT DONT
C     EXCEED STORAGE DIMENSION
   40 DO 70  IK=1,JDK
C          READ K,(LAMBDA(M),DLDK(M),PSIO(M),DPSIO(M),DPSIB(M),
C            WAKI(M),SUPT(M),SUPB(M),DLDK2(M),M=1,MODES)
      READ(NTDTAB) RK,((TEMP1(I,M),I=1,9),M=1,MODES)
C     SKIP OUT OF LOOP WHEN ENTIRE TABLE HAS BEEN READ
      IF (EOF,NTDTAB)  80,50
C     DATA WERE READ.  SAVE VALUE OF K
   50 TK(IK) = RK
      DO 60  MODE=MINMOD,MAXMOD
C     NOTE THAT HERE (AND EVERYWHERE ELSE IN THE SP ROUTINES), THE
C     VARIABLE -MODE- IS THE STORAGE INDEX OF THE MODE BEING
C     CONSIDERED.  NOW SET ACTUAL MODE NUMBER.
      MN = MODE +MODE1 -1
C     TRANSFER VARIABLES FROM TEMP STORAGE TO DISPERSION TABLES
      TEVAL(IK,MODE) = TEMP1(1,MN)
      TDLDK(IK,MODE) = TEMP1(2,MN)
      TPSIO(IK,MODE) = TEMP1(3,MN)
      TDPSIO(IK,MODE) = TEMP1(4,MN)
      TDPSIB(IK,MODE) = 0.
      IF (IBODY .NE. 0)  TDPSIB(IK,MODE) = TEMP1(5,MN)
      TWAKI(IK,MODE) = 0.
      IF (IWAKE .NE. 0)  TWAKI(IK,MODE) = TEMP1(6,MN)
C     SUPERSTRUCTURE TERM IS PSI(BOTTOM)-PSI(TOP)
      TSUPT(IK,MODE) = 0.
      IF (ISUPR .NE. 0)  TSUPT(IK,MODE) = TEMP1(8,MN) -TEMP1(7,MN)
      TD2L(IK,MODE) = TEMP1(9,MN)
   60 CONTINUE
   70 CONTINUE
C
      IK = JDK+1
C     SET NUMBER OF ENTRIES IN TABLE
   80 MAXK = IK-1
C
```

```
C     CONSTRUCT WAVE FAMILY EDGE TABLES AND TABLE OF STATIONARY
C     PHASE POINTS
      CALL SPWFAM
C
C     SKIP IF WAKE IS OFF
      IF (IWAKE .EQ. 0)  GO TO 130
C     FINISH COMPUTATION OF THE WAKE SOURCE TERM
C     LOOP FOR EACH MODE
      DO 110  MODE=MINMOD,MAXMOD
C     GET FWA-1 OF THIS MODE IN DISP TABLES
      IFWA1 = (MODE-1)*JDK
C     ADDRESSES OF 1ST AND LAST ENTRIES FOR THIS MODE
      LIM1 = IFWA1 +LIMEDG(1,MODE)
      LIM2 = IFWA1 + MAXK
C     LOOP FOR EACH ENTRY IN THE TABLE
      DO 90  LOCDT=LIM1,LIM2
C     FINISH COMPUTATION OF WAKE SOURCE TERM
   90 TWAKI(LOCDT) = 2.*CWAKM*BODSPD*TEVAL(LOCDT)*TWAKI(LOCDT)
C     SKIP IF 1ST EDGE IS AT K<=0
      IF (EVLEDG(1,MODE) .EQ. TEVAL(IFWA1+1))  GO TO 100
C     FILL IN SLOT PRECEDING THE 1ST TABLE ENTRY BY EXTRAPOLATION
      TWAKI(LIM1-1) = TWAKI(LIM1) + (TEVAL(LIM1-1)-TEVAL(LIM1))
     1        * (TWAKI(LIM1+1)-TWAKI(LIM1))/(TEVAL(LIM1+1)-TEVAL(LIM1))
      GO TO 110
C     FINISH COMPUTATION AT K=0 ENTRY
  100 TWAKI(IFWA1+1) = 2.*CWAKM*BODSPD*TEVAL(IFWA1+1)*TWAKI(IFWA1+1)
  110 CONTINUE
  130 CALL TIMER(-15)
      RETURN
      END
      SUBROUTINE SPDTPP
C     PRINT/PLOT DISPERSION RELATION
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,      RBSEP2, RBSTR, RBLIM
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,      X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,      IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,      ISPHAS
C
      COMMON /NAME/ NAMES(2,10), DTNAMS(2,9)
C
      COMMON/PPCOM/
     1       LENPP, PNAME, PMIN, PMAX, PLEN, VNAME, VMIN, VMAX
     2,      VLEN, FNAME(2), FMIN, FMAX, FLEN, FTODP, TITLE(2)
     3,      IOCUR, IPLTYP, IPLOT, IPRINT, IEDIT, NP, IVLIST, NOPP
     4,      IDPP, NV, ISYM
     E,      ENDPP, IBLOKS(1)
C
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,      SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMMON /WAKE/ IWAKE, CWAKR, CWAKX, XWAKE, WAKRAD, XWNOM
     1,      RESLVS, CWAKM
C
      COMPLEX XDEP, FSRC
      COMMON // RK, EVAL, DLDK, D2L, PSIO, DPSIO, DPSIB, WAKI, SUPT
     1,      YX, D3L, XI, ETA, INRANG, MODE, MINMOD, MAXMOD, MODES
     2,      MAXK, IFWA1, LOCDT, LOCDTK, ITHWAV, NWAVES, NWFTAB(41)
     3,      HILO, DYX, PYX, PDYX, PEVAL, PRK, FAZ, SGNFZ2, FAZ2
     4,      FAZ3, XDEP, FSRC, VAR, SIGNAL, SIGCUT(800)
```

```
      EQUIVALENCE (TEMP1,SIGCUT)
      DIMENSION TEMP1(9,1)
      COMMON // TK(100), TEVAL(100,41), TDLDK(100,41), TD2L(100,41)
     1,     TPSIO(100,41), TDPSIO(100,41), TDPSIB(100,41), TWAKI(100,41)
     2,     TSUPT(100,41), TYX(100,41), YXEDG(20,41), EVLEDG(20,41)
     3,     LIMEDG(20,41), INDEDG(20,41)
C
C
C     SKIP IF SPECIAL PRINT IS OFF
      IF (IPRDT .EQ. 0)  GO TO 100
C     LOOP FOR EACH MODE IN TABLE
      DO 90  MODE=MINMOD,MAXMOD
C     ACTUAL MODE NUMBER
      MN = MODE + MODE1 -1
      WRITE(5,30) MN
   30 FORMAT(22H1  DISPERSION RELATION/7H   MODE,I3/1H0,6X,1HK,9X,
     1      4H-Y/X,7X,6HLAMBDA,5X,5HDL/DK,6X,7HD2L/DK2,4X,6HW(OBS),5X,
     2      10HDW/DZ(OBS),1X,10HDW/DZ(BOD),1X,5HTWAKE,6X,5HTSUPR)
C     PRESET K INDEX OF NEXT ENTRY TO PRINT
      NEXT = 1
C     NUMBER OF WAVE FAMILIES IN THIS MODE
      NWAVES = NWFTAB(MODE)
      LIM = NWAVES + 1
      DO 80  ITHWAV=1,LIM
C     PICK UP K INDEX OF LAST ENTRY IN THIS WAVE FAMILY
      LAST = LIMEDG(ITHWAV,MODE)
      IF (ITHWAV .NE. LIM)  LAST = LAST - 1
C     SKIP IF ENTIRE WAVE FAMILY LIES BETWEEN ADJACENT TABLE POINTS
      IF (LAST .LT. NEXT)  GO TO 60
      WRITE(6,50) (I,TK(I),TYX(I,MODE),TEVAL(I,MODE),TDLDK(I,MODE)
     1,     TD2L(I,MODE), TPSIO(I,MODE), TDPSIO(I,MODE),TDPSIB(I,MODE)
     2,     TWAKI(I,MODE),TSUPT(I,MODE),I=NEXT,LAST)
   50 FORMAT(1X,I3,10E11.3)
      NEXT = LAST + 1
   60 IF (ITHWAV .NE. LIM)
     1            WRITE(6,70) YXEDG(ITHWAV,MODE),EVLEDG(ITHWAV,MODE)
   70 FORMAT(11X,4HEDGE,2E11.3)
   80 CONTINUE
   90 CONTINUE
C
C     SKIP IF NOT PRINTING WAVE FAMILY EDGE TABLE
  100 IF (IPREDG .EQ. 0)  GO TO 200
      WRITE(6,110)
  110 FORMAT(25H1  WAVE FAMILY EDGE TABLE)
C     LOOP FOR EACH MODE
      DO 140  MODE=MINMOD,MAXMOD
C     ACTUAL MODE NUMBER
      MN = MODE + MODE1 -1
      WRITE(6,120)  MN
  120 FORMAT(10H0*****MODE,I3/6X,4H-Y/X,10X,6HLAMBDA)
C     NUMBER OF WAVE FAMILIES
      NWAVES = NWFTAB(MODE)
      WRITE(6,130) (I,YXEDG(I,MODE),EVLEDG(I,MODE),I=1,NWAVES)
  130 FORMAT(1X,I2,2E14.6)
  140 CONTINUE
C
C     LOOP FOR EACH D.T. VARIABLE WHICH CAN BE SENT TO PP PROCESSOR
  200 DO 310  ITHVAR=1,9
C     SKIP IF PP OPTION IS OFF FOR THIS VARIABLE
      IF (IPPDT(ITHVAR) .EQ. 0)  GO TO 310
C     SKIP IF DESIRED VARIABLE HAS NOT BEEN COMPUTED
      IF (ITHVAR .EQ. 4  .AND.  IBODY .EQ. 0)  GO TO 310
```

```
      IF (ITHVAR .EQ. 5  .AND.  IWAKE .EQ. 0)  GO TO 310
      IF (ITHVAR .EQ. 6  .AND.  ISUPR .EQ. 0)  GO TO 310
C     PRESET THE PP SPECS
      CALL SETID(DTNAMS(1,ITHVAR), 1, 4HMODE, 1HK, DTNAMS(1,ITHVAR))
C     INDICATE VARIABLE LIST IS FIXED
      IVLIST = 2
C     LOOP FOR EACH MODE
      DO 300  MODE=MINMOD,MAXMOD
C     FLOAT ACTUAL MODE NUMBER
      RMODE = MODE + MODE1 -1
C     JUMP ON VARIABLE TO BE DISPLAYED
      GO TO (210,220,230,240,250,260,270,280,290),ITHVAR
  210 CALL WRTDAT(1, MAXK, TDLDK(1,MODE), 1, RMODE)
      GO TO 300
  220 CALL WRTDAT(1, MAXK, TPSIO(1,MODE), 1, RMODE)
      GO TO 300
  230 CALL WRTDAT(1, MAXK, TDPSIO(1,MODE), 1, RMODE)
      GO TO 300
  240 CALL WRTDAT(1, MAXK, TDPSIB(1,MODE), 1, RMODE)
      GO TO 300
  250 CALL WRTDAT(LIMEDG(1,MODE), MAXK, TWAKI(1,MODE), 1, RMODE)
      GO TO 300
  260 CALL WRTDAT(1, MAXK, TSUPT(1,MODE), 1, RMODE)
      GO TO 300
  270 CALL WRTDAT(1, MAXK, TEVAL(1,MODE), 1, RMODE)
      GO TO 300
  280 CALL WRTDAT(1, MAXK, TD2L(1,MODE), 1, RMODE)
      GO TO 300
  290 CALL WRTDAT(LIMEDG(1,MODE), MAXK, TYX(1,MODE), 1, RMODE)
  300 CONTINUE
C     WRITE THE PP ID RECORD
      CALL WRTID(MAXK, TK, 1)
  310 CONTINUE
      RETURN
      END
      SUBROUTINE SPEDGE
C     TEST FOR AND FIND WAVE FAMILY EDGES (EXTREMA OF YX)
C
      COMMON /CONST/ JDK, JDMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,     JDCKSV, JDMSP, JDEDGE
C
      COMPLEX XDEP, FSRC
      COMMON // RK, EVAL, DLDK, D2L, PSIO, DPSIO, DPSIB, WAKI, SUPT
     1,     YX, D3L, XI, ETA, INRANG, MODE, MINMOD, MAXMOD, MODES
     2,     MAXK, IFWA1, LOCDT, LOCDTK, ITHWAV, NWAVES, NWFTAB(41)
     3,     HILO, DYX, PYX, PDYX, PEVAL, PRK, FAZ, SGNFZ2, FAZ2
     4,     FAZ3, XDEP, FSRC, VAR, SIGNAL, SIGCUT(800)
      EQUIVALENCE (TEMP1,SIGCUT)
      DIMENSION TEMP1(9,1)
      COMMON // TK(100), TEVAL(100,41), TDLDK(100,41), TD2L(100,41)
     1,     TPSIO(100,41), TDPSIO(100,41), TDPSIB(100,41), TWAKI(100,41)
     2,     TSUPT(100,41), TYX(100,41), YXEDG(20,41), EVLEDS(20,41)
     3,     LIMEDG(20,41), INDEDG(20,41)
C
C
C     NOTE EDGES ARE THE EXTREMA OF YX AS A FUNCTION OF EVAL
C     HILO=1 IF LOOKING FOR A MINIMUM IN YX, HILO=-1 FOR A MAXIMUM
C     FIND EDGE IF DERIVATIVE HAS CHANGED SIGN
      IF (DYX*HILO .LT. 0.)  RETURN
C     FIND EXTREMUM BETWEEN PEVAL AND EVAL.
C     SAVE VALUES ON RIGHT SIDE OF EXTREMUM
      SYX = YX
```

```
      SDYX = DYX
      SEVAL = EVAL
      SRK = RK
C     SET RIGHT HAND POINT
      RYX = YX
      RDYX = DYX
      REVAL = EVAL
      RRK = RK
C     10 HALVING LOOPS INCREASE RESOLUTION BY FACTOR OF 1024
      DO 20  ITER=1,10
C     HALVE THE INTERVAL
      EVAL = .5*(PEVAL+REVAL)
C     INTERPOLATE FOR RK, DLDK, D2L AS FUNCTIONS OF EVAL
      CALL SPDISP(1)
C     COMPUTE YX AND DYX FROM EVAL, RK,DLDK,D2L
      CALL SPFUNC(1)
C     SKIP IF EVAL IS RIGHT OF EXTREMUM
      IF (DYX*HILO .GT. 0.)  GO TO 10
C     EVAL IS LEFT OF EXTREMUM.  REPLACE LEFT POINT
      PYX = YX
      PDYX = DYX
      PEVAL = EVAL
      PRK = RK
      GO TO 20
C     EVAL IS RIGHT OF EXTREMUM.  REPLACE RIGHT POINT
   10 RYX = YX
      RDYX = DYX
      REVAL = EVAL
      RRK = RK
   20 CONTINUE
C
      NWAVES = NWAVES+1
C     SKIP IF STORAGE NOT EXCEEDED
      IF (NWAVES .LT. JDEDGE)  GO TO 120
      WRITE(6,110) JDEDGE,MODE
  110 FORMAT(32H WAVE FAMILY EDGE TABLE EXCEEDED,2I10)
      CALL ERRXIT
C     EXTREMUM BETWEEN P AND R.  SELECT THE BETTER AND INSERT INTO
C     EDGE TABLES
  120 IF (HILO*(RYX-PYX) .LT. 0.)  GO TO 130
      YXEDG(NWAVES,MODE) = PYX
      EVLEDG(NWAVES,MODE) = PEVAL
      GO TO 140
  130 YXEDG(NWAVES,MODE) = RYX
      EVLEDG(NWAVES,MODE) = REVAL
  140 LIMEDG(NWAVES,MODE) = LOCDTK
      INDEDG(NWAVES,MODE) = LOCDTK
C     COMPLEMENT THE MIN/MAX SEARCH FLAG
      HILO = -HILO
C     SET PREVIOUS POINT = RIGHT HAND POINT...
      PYX = RYX
      PDYX = RDYX
      PEVAL = REVAL
      PRK = RRK
C     ...AND RESTORE ORIGINAL RIGHT HAND POINT
      YX = SYX
      DYX = SDYX
      EVAL = SEVAL
      RK = SRK
      RETURN
      END
      SUBROUTINE SPEVAL
```

```
C     FIND EIGENVALUE EVAL AND TABLE POSITION LOCDT FOR A GIVEN
C     STATIONARY PHASE POINT YX AND WAVE FAMILY ITHWAV
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,     RBSEP2, RBSTR, RBLIM
C
      COMMON /CONST/ JDK, JDMODE, JDTCL, PI, NULL, JDCKL, JDMFT
     1,     JDCKSV, JDMSP, JDEDGE
C
      COMPLEX XDEP, FSRC
      COMMON // RK, EVAL, DLDK, D2L, PSIO, DPSIO, DPSIB, WAKI, SUPT
     1,     YX, D3L, XI, ETA, INRANG, MODE, MINMOD, MAXMOD, MODES
     2,     MAXK, IFWA1, LOCDT, LOCDTK, ITHWAV, NWAVES, NWFTAB(41)
     3,     HILO, DYX, PYX, PDYX, PEVAL, PRK, FAZ, SGNFZ2, FAZ2
     4,     FAZ3, XDEP, FSRC, VAR, SIGNAL, SIGCUT(800)
      EQUIVALENCE (TEMP1,SIGCUT)
      DIMENSION TEMP1(9,1)
      COMMON // TK(100), TEVAL(100,41), TDLDK(100,41), TD2L(100,41)
     1,     TPSIO(100,41), TDPSIO(100,41), TDPSIB(100,41), TWAKI(100,41)
     2,     TSUPT(100,41), TYX(100,41), YXEDG(20,41), EVLEDG(20,41)
     3,     LIMEDG(20,41), INDEDG(20,41)
C
C
C     GET SINGLE INDEX EQUIVALENT OF (ITHWAV,MODE) FOR ADDRESSING
C     EDGE TABLES
      J = (MODE-1)*JDEDGE + ITHWAV
C     SKIP IF YX=-Y/X IS WITHIN THE RANGE OF THIS WAVE FAMILY
      IF ((YXEDG(J)-YX)*(YXEDG(J+1)-YX) .LE. 0.)  GO TO 10
C     SET FLAG SHOWING THIS WAVE FAMILY DOES NOT CONTRIBUTE AT YX
      INRANG = 0
      RETURN
C
C     SHOW THIS WAVE FAMILY DOES CONTRIBUTE AT YX
   10 INRANG = 1
C     SET FWA-1 OF DISPERSION TABLES FOR THIS MODE
      IFWA1 = (MODE-1)*JDK
C     SET LIM1/LIM2 = INDEX OF LOWEST/HIGHEST VALUE OF YX WITHIN
C     RANGE.  SET INC = INCREMENT IN INDEX TO INCREASE TYX(I).
C     SET SIGN OF 2ND DERIVATIVE OF PHASE FUNCTION
C     SKIP IF TABLE DECREASES
      IF (YXEDG(J) .GT. YXEDG(J+1))  GO TO 20
      LIM1 = LIMEDG(J) + IFWA1
      LIM2 = LIMEDG(J+1) -1 + IFWA1
      INC = 1
      SGNFZ2 = 1.
      GO TO 30
   20 LIM1 = LIMEDG(J+1) -1 + IFWA1
      LIM2 = LIMEDG(J) + IFWA1
      INC = -1
      SGNFZ2 = -1.
C     SKIP IF THERE ARE D.T. POINTS WITHIN RANGE
   30 IF (LIMEDG(J+1) .GT. LIMEDG(J))  GO TO 40
C     ENTIRE WAVE FAMILY LIES BETWEEN ADJACENT TABLE POINTS.  INTERP
C     BETWEEN EDGES
      EVAL = EVLEDG(J) + (EVLEDG(J+1)-EVLEDG(J))/(YXEDG(J+1)-YXEDG(J))
     1                            *(YX-YXEDG(J))
      IDT = LIM2
      GO TO 110
C
C     FIND PROPER POSITION IN TYX TABLE.  PICK UP PREVIOUS POSITION
   40 IDT = INDEDG(J) + IFWA1
      IF (YX .GE. TYX(IDT))  GO TO 70
```

```
C     SKIP IF YX LIES BETWEEN EDGE AND TABLE POINT
      IF (YX .LE. TYX(LIM1))  GO TO 60
C     TYX(LIM1) .LT. YX .LT. TYX(IDT)   FIND PROPER POSITION IN TABLE
   50 IDT = IDT -INC
      IF (YX .LT. TYX(IDT))  GO TO 50
      GO TO 90
C     INTERPOLATE BETWEEN TYX(LIM1) AND YXEDG(J+(1-INC)/2)
   60 I = J + (1-INC)/2
      DIV = YXEDG(I) - TYX(LIM1)
      IF (DIV .EQ. 0.)  DIV = 1.
      EVAL = EVLEDG(I) + (EVLEDG(I)-TEVAL(LIM1))/DIV *(YX-YXEDG(I))
      IDT = LIM1 -INC
C     SAVE POSITION IN TABLE
      INDEDG(J) = LIM1 -IFWA1
      GO TO 110
C
C     SKIP IF YX LIES BETWEEN EDGE AND TABLE POINT
   70 IF (YX .GE. TYX(LIM2))  GO TO 100
C     TYX(IDT) .LE. YX .LT. TYX(LIM2)   FIND PROPER POSITION IN TABLE
   80 IDT = IDT +INC
      IF (YX .GE. TYX(IDT))  GO TO 80
      IDT = IDT -INC
C     INTERP BETWEEN TYX(IDT) AND TYX(IDT+INC)
   90 EVAL = TEVAL(IDT) + (TEVAL(IDT+INC)-TEVAL(IDT))
     1                 /(TYX(IDT+INC)-TYX(IDT)) *(YX-TYX(IDT))
C     SAVE POSITION IN TABLE
      INDEDG(J) = IDT -IFWA1
      GO TO 110
C     INTERPOLATE BETWEEN TYX(LIM2) AND YXEDG(J+(1+INC)/2)
  100 I = J + (1+INC)/2
      DIV = YXEDG(I) - TYX(LIM2)
      IF (DIV .EQ. 0.)  DIV = 1.
      EVAL = EVLEDG(I) + (EVLEDG(I)-TEVAL(LIM2))/DIV *(YX-YXEDG(I))
      IDT = LIM2
C     SAVE POSITION IN TABLE
      INDEDG(J) = LIM2-IFWA1
C     AT THIS POINT EVAL IS BETWEEN TEVAL(IDT) AND TEVAL(IDT+INC)
C     INCLUSIVE.  SET LOCDT SO THAT TEVAL(LOCDT-1) .LE. EVAL .LE.
C     TEVAL(LOCDT)
  110 LOCDT = IDT + (1+INC)/2
C     SET K INDEX CORRESPONDING TO LOCDT
      LOCDTK = LOCDT -IFWA1
C     SKIP IF THIS IS NOT A TRANSVERSE WAVE
      IF (ITHWAV .NE. 1)  GO TO 120
      IF (EVLEDG(J) .EQ. TEVAL(IFWA1+1))  GO TO 120
C     FOLLOWING PROCEDURE SHOULD IMPROVE ACCURACY OF EVAL FOR A
C     TRANSVERSE WAVE.  FIRST INTERPOLATE FOR RK, DLDK AS FUNCTION OF
C     EVAL
      C2 = (EVAL-TEVAL(LOCDT-1))/(TEVAL(LOCDT)-TEVAL(LOCDT-1))
      C1 = 1. -C2
      RK = C1*TK(LOCDTK-1) + C2*TK(LOCDTK)
      DLDK = C1*TDLDK(LOCDT-1) + C2*TDLDK(LOCDT)
      T1 = .5*RK*DLDK/EVAL
      TEMP = .5*(1.-T1)/YX *(1.-SQRT(1.-4.*T1*(YX/(1.-T1))**2))
      EVAL = (TEMP**2+1.)/BODSPD**2
C     FORCE THIS TO BE WITHIN KNOWN LIMITS
      IF (EVAL .LT. TEVAL(LOCDT-1))  EVAL = TEVAL(LOCDT-1)
      IF (EVAL .GT. TEVAL(LOCDT))  EVAL = TEVAL(LOCDT)
C
  120 RETURN
      END
      SUBROUTINE SPFUNC(JEDGE)
```

```
C     COMPUTE STATIONARY PHASE FUNCTIONS
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,      RBSEP2, RBSTR, RBLIM
C
      COMPLEX XDEP, FSRC
      COMMON // RK, EVAL, DLDK, D2L, PSIO, DPSIO, DPSIB, WAKI, SUPT
     1,      YX, D3L, XI, ETA, INRANG, MODE, MINMOD, MAXMOD, MODES
     2,      MAXK, IFWAL, LOCDT, LOCDTK, ITHWAV, NWAVES, NWFTAB(41)
     3,      HILO, DYX, PYX, PDYX, PEVAL, PRK, FAZ, SGNFZ2, FAZ2
     4,      FAZ3, XDEP, FSRC, VAR, SIGNAL, SIGCUT(800)
      EQUIVALENCE (TEMP1,SIGCUT)
      DIMENSION TEMP1(9,1)
C
C
C     X AND Y COMPONENTS OF WAVE NUMBER RK
      XI = RK/(BODSPD*SQRT(EVAL))
      ETA = SQRT (RK**2-XI**2)
C     SOME TEMPORARIES
      D1 = DLDK*RK/EVAL
      D2 = D2L*RK**2/EVAL
      T1 = (XI/ETA)**2
      T2 = XI/RK
C     D(XI)/DK AND D(ETA)/DK
      XI1 = T2*(1.-.5*D1)
      ETA1 = ETA/RK *(1.+.5*T1*D1)
C     D2(XI)/DK2 AND D2(ETA)/DK2
      XI2 = -.5*T2/RK *(D2+D1*(2.-1.5*D1))
      ETA2 = .5*T2**2/ETA *(D2+D1*(2.-D1*(2.+.5*T1)))
C     SKIP IF PHASE FUNCTION AND ITS DERIVATIVES ARE REQUIRED
      IF (JEDGE .EQ. 0)  GO TO 10
C     ROUTINE IS BEING USED AS PART OF THE PROCESS OF FINDING WAVE
C     FAMILY EDGES.  COMPUTE STATIONARY PHASE POINT YX=D(XI)/D(ETA) AND
C     D(YX)/D(EVAL)=D(YX)/DK *D(K)/D(EVAL)
      YX = XI1/ETA1
      DYX = (ETA1*XI2-XI1*ETA2)/ETA1**2 /DLDK
      RETURN
C
   10 D3 = D3L*RK**3/EVAL
C     D3(XI)/DK3 AND D3(ETA)/DK3
      XI3 = -.5*T2/RK**2 *(D3+D2*( 3.-4.5*D1)+D1**2*(-4.5+3.75*D1))
      ETA3 = .5*T2**2/(ETA*RK) *(D3+D2*(3.-(6.+1.5*T1)*D1)
     1                         +D1**2*(-6.-1.5*T1+(6.+3.*T1+.75*T1**2)*D1))
C     PHASE FUNCTION
      FAZ = XI - XI1/ETA1 *ETA
C     D2(FAZ)/D(ETA)2 AND D3(FAZ)/D(ETA)3
      FAZ2 = (ETA1*XI2-XI1*ETA2)/ETA1**3
      FAZ3 = (ETA1*(ETA1*XI3-3.*ETA2*XI2-XI1*ETA3)+3.*XI1*ETA2**2)
     1                                                  /ETA1**5
      RETURN
      END
      SUBROUTINE SPSRC
C     COMPUTE SOURCE FUNCTION FSRC
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,      RBSEP2, RBSTR, RBLIM
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,      X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,      IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,      ISPHAS
C
```

```
      COMMON /SUPER/ ISUPR, SUPTOP, SUPBOT, IPSUPR, SUSTR, SUSEP2
     1,     SUPMID, SULIM, SUPDIA, SUPLEN
C
      COMMON /WAKE/ IWAKE, CWAKR, CWAKX, XWAKE, WAKRAD, XWNOM
     1,     RESLVS, CWAKM
C
      COMPLEX XDEP, FSRC
      COMMON // RK, EVAL, DLDK, D2L, PSIO, DPSIO, DPSIB, WAKI, SUPT
     1,     YX, D3L, XI, ETA, INRANG, MODE, MINMOD, MAXMOD, MODES
     2,     MAXK, IFWA1, LOCDT, LOCDTK, ITHWAV, NWAVES, NWFTAB(41)
     3,     HILO, DYX, PYX, PDYX, PEVAL, PRK, FAZ, SGNFZ2, FAZ2
     4,     FAZ3, XDEP, FSRC, VAR, SIGNAL, SIGCUT(800)
      EQUIVALENCE (TEMP1,SIGCUT)
      DIMENSION TEMP1(9,1)
C
C
      FSRC = (0., 0.)
C     SKIP IF BODY OFF
      IF (IBODY .EQ. 0)  GO TO 100
      IF (IBODY .EQ. 2)  GO TO 20
C     RANKINE BODY
C     RBSTR=SOURCE STRENGTH, RBSEP2=1/2 SOURCE TO SINK SEPARATION
      FSRC = CMPLX(0., -2.*RBSTR*DPSIB*SIN(XI*RBSEP2))
      GO TO 100
C     DIPOLE BODY.  RBLIM=LIM(RBSTR*RBSEP2)
   20 FSRC = CMPLX(0., -2.*RBLIM*DPSIB*XI)
C
C     SKIP IF WAKE IS OFF
  100 IF (IWAKE .EQ. 0)  GO TO 200
C     SKIP IF WAKE NOT ON YET
      IF (X .LT. XWAKE)  GO TO 200
      FSRC = FSRC + WAKI*CMPLX(-COS(XI*XWAKE), SIN(XI*XWAKE))
C
C     SKIP IF SUPERSTRUCTURE IS OFF
  200 IF (ISUPR .EQ. 0)  GO TO 300
      IF (ISUPR .EQ. 2)  GO TO 220
C     OVAL SUPERSTRUCTURE.  SUSTR=SOURCE STRENGTH, SUSEP2=1/2 SOURCE
C     TO SINK SEPARATION, SUPT=PSI(BOT)-PSI(TOP)
      TEMP = 2.*SUSTR*SUPT*SIN(XI*SUSEP2)
C     SUPMID = X COORDINATE OF MIDDLE OF SUPERSTRUCTURE
      FSRC = FSRC + TEMP*CMPLX(SIN(XI*SUPMID), COS(XI*SUPMID))
      GO TO 300
C     CIRCULAR SUPER.   SULIM=LIM(SUSTR*SUSEP2)
  220 TEMP = 2.*SULIM*SUPT*XI
      FSRC = FSRC + TEMP*CMPLX(SIN(XI*SUPMID), COS(XI*SUPMID))
C
  300 RETURN
      END
      SUBROUTINE SPVAR
C     COMPUTE THE VARIABLE-DEPENDENT (BUT SOURCE-INDEPENDENT) PART
C     OF THE SIGNAL, THEN PUT EVERYTHING TOGETHER
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,     RBSEP2, RBSTR, RBLIM
C
      COMMON /GRID/ OBSDEP, NOBS, COBS, OBSMAX, TABOBS(100), ITHOBS
     1,     X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,     IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,     ISPHAS
C
      COMPLEX XDEP, FSRC
      COMMON // RK, EVAL, DLDK, D2L, PSIO, DPSIO, DPSIB, WAKI, SUPT
```

```
     1,     YX, D3L, XI, ETA, INRANG, MODE, MINMOD, MAXMOD, MODES
     2,     MAXK, IFWA1, LOCDT, LOCDTK, ITHWAV, NWAVES, NWFTAB(41)
     3,     HILO, DYX, PYX, PDYX, PEVAL, PRK, FAZ, SGNFZ2, FAZ2
     4,     FAZ3, XDEP, FSRC, VAR, SIGNAL, SIGOUT(800)
      EQUIVALENCE (TEMP1,SIGOUT)
      DIMENSION TEMP1(9,1)
C
C
      TEMP = 2.*XI*(2.*RK/DLDK*(ETA/(BODSPD*XI**2))**2+1.)
      TEMP = (2./ABS(X*FAZ3))**(1./3.) /TEMP
      GO TO (10,20,30,40,50,60,70,80,90,100),IVAR
C
C     (U) DOWNTRACK VELOCITY DISTURBANCE
   10 VAR = DPSIC * XI/RK**2
      GO TO 200
C
C     (V) CROSS TRACK VELOCITY
   20 VAR = DPSIO * ETA/RK**2
      GO TO 200
C
C     (DELTA-X) DOWN TRACK DISPLACEMENT
   30 VAR = DPSIO /(BODSPD*RK**2)
      GO TO 210
C
C     (DELTA-Y) CROSS TRACK DISPLACEMENT
   40 VAR = DPSID * ETA/(BODSPD*XI*RK**2)
      GO TO 210
C
C     (DELTA-Z) VERTICAL DISPLACEMENT
   50 VAR = -PSIO /(BODSPD*XI)
      GO TO 200
C
C     (EPSILON-X) DOWN TRACK STRAIN
   60 VAR = DPSIC * XI/(BODSPD*RK**2)
      GO TO 200
C
C     (EPSILON-Y) CROSS TRACK STRAIN
   70 VAR = DPSIO * ETA**2/(BODSPD*XI*RK**2)
      GO TO 200
C
C     (GAMMA-XY) SHEARING STRAIN IN HORIZONTAL PLANE
   80 VAR = DPSIC * 2.*ETA/(BODSPD*RK**2)
      GO TO 200
C
C     (SIGMA) HORIZONTAL PLANE DILATATION
   90 VAR = -DPSIC
      GO TO 210
C
C     (W) VERTICAL VELOCITY
  100 VAR = PSIC
      GO TO 210
C
C
C     PUT IT ALL TOGETHER
  200 SIGNAL = SIGNAL + TEMP*VAR*REAL(FSRC*XDEP)
      RETURN
  210 SIGNAL = SIGNAL + TEMP*VAR*AIMAG(FSRC*XDEP)
      RETURN
      END
      SUBROUTINE SPWFAM
C     CONSTRUCT WAVE FAMILY EDGE TABLES AND TABLE OF STATIONARY PHASE
C     POINTS (-Y/X)
```

```
C
      COMMON /BODY/ IBODY, IPBODY, BODDEP, BODDIA, BODLEN, BODSPD
     1,     RBSEP2, RBSTR, RBLIM
C
      COMMON /CONST/ JDK, JDMODE, JDTOL, PI, NULL, JDCKL, JDMFT
     1,     JDCKSV, JDMSP, JDEDGE
C
      COMPLEX XDEP, FSRC
      COMMON // RK, EVAL, DLDK, D2L, PSIO, DPSIO, DPSIB, WAKI, SUPT
     1,     YX, D3L, XI, ETA, INRANG, MODE, MINMOD, MAXMOD, MODES
     2,     MAXK, IFWA1, LOCDT, LOCDTK, ITHWAV, NWAVES, NWFTAB(41)
     3,     HILO, DYX, PYX, PDYX, PEVAL, PRK, FAZ, SGNFZ2, FAZ2
     4,     FAZ3, XDEP, FSRC, VAR, SIGNAL, SIGCUT(800)
      EQUIVALENCE (TEMP1,SIGCUT)
      DIMENSION TEMP1(9,1)
      COMMON // TK(100), TEVAL(100,41), TDLDK(100,41), TD2L(100,41)
     1,     TPSIO(100,41), TDPSIO(100,41), TDPSIB(100,41), TWAKI(100,41)
     2,     TSUPT(100,41), TYX(100,41), YXEDG(20,41), EVLEDG(20,41)
     3,     LIMEDG(20,41), INDEDG(20,41)
C
C
      CALL TIMER(16)
      SQUIN = 1./BODSPD**2
      MODE = MAXMOD
C     START LOOP THROUGH ALL MODES FROM MAX TO MIN
C     GET FWA-1 OF THIS MODE IN DISP TABLES
   10 IFWA1 = (MODE-1)*JDK
C     CHECK FOR (TRANSVERSE WAVE), (CRITICAL SPEED), (DIVERGING WAVE)
      IF (TEVAL(IFWA1+1)-SQUIN)  50,20,40
   20 WRITE(6,30) BODSPD
   30 FORMAT(54H STATIONARY PHASE INADEQUATE.  BODY AT CRITICAL SPEED=,
     1    E13.6)
      CALL ERRXIT
C
C     DIVERGING WAVE.
C     SET VALUE OF -Y/X AND ITS DERIVATIVE WRT EVAL
   40 YX = 1./SQRT(TEVAL(IFWA1+1)*BODSPD**2-1.)
      DYX = -1.5*BODSPD**2 *YX**3
C     EIGENVALUE AND WAVENUMBER CORRESPONDING TO YX
      EVAL = TEVAL(IFWA1+1)
      RK = TK(1)
C     INDEX OF NEXT TABLE POINT
      LOCDTK = 2
C     FILL IN 1ST ENTRY OF STATIONARY PHASE POINT TABLE (NOT USED,
C     BUT LOOKS NICE ON PRINT OF DISP TABLES)
      TYX(IFWA1+1) = YX
C     THIS EDGE IS A MAX OF YX.  SET FLAG TO LOOK FOR A MIN
      HILO = 1.
      GO TO 90
C
C     TRANSVERSE WAVE.  FIND INNER EDGE (AT EVAL=1/BODSPD**2)
   50 DO 60  LOCDTK=1,MAXK
      LOCDT = IFWA1+LOCDTK
C     PUT 0 IN STATIONARY PHASE POINT TABLE SO PRINT WILL LOOK NICE
      TYX(LOCDT) = 0.
   60 IF (TEVAL(LOCDT) .GT. SQUIN)  GO TO 80
C     1ST EDGE IS BEYOND RANGE OF DISP TABLE.  ALSO SKIP LOWER
C     MODES--THEY ARE WORSE CASES
      MINMOD = MODE + 1
      IF (MINMOD .LE. MAXMOD)  GO TO 130
      WRITE(6,70)
   70 FORMAT(31H MAX(K) IN DISP TABLE TOO SMALL)
```

```
      CALL ERRXIT
C     SET VALUE OF -Y/X AT INNER EDGE
   80 YX = 0.
C     FAKE D(YX)/D(EVAL).  (YX=0 IS AN ABSOLUTE MINIMUM SO LOGIC TO
C     FIND EXTREMA WILL NEVER USE DYX)
      DYX = 1.
C     EIGENVALUE AND WAVENUMBER CORRESPONDING TO YX
      EVAL = SQUIN
      RK = TK(LOCDTK-1) + (TK(LOCDTK)-TK(LOCDTK-1))
     1          /(TEVAL(LOCDT)-TEVAL(LOCDT-1)) *(EVAL-TEVAL(LOCDT-1))
C     THIS EDGE IS A MIN OF YX.  SET FLAG TO LOOK FOR A MAX
      HILO = -1.
C
C
C     INSERT 1ST EDGE DATA INTO EDGE TABLES
   90 YXEDG(1,MODE) = YX
      EVLEDG(1,MODE) = EVAL
C     SET D.T. INDEX OF 1ST POINT BEYOND EDGE AND PRESET POSITION
C     SAVER (USED BY SPEVAL)
      LIMEDG(1,MODE) = LOCDTK
      INDEDG(1,MODE) = LOCDTK
C     INITIALIZE COUNT OF NUMBER OF WAVE FAMILIES
      NWAVES = 1
C
C     LOOP FOR REMAINING TABLE POINTS
      LIM = LOCDTK
      DO 100  LOCDTK = LIM,MAXK
C     SAVE PAST VALUES OF YX, DYX, EVAL, RK
      PYX = YX
      PDYX = DYX
      PEVAL = EVAL
      PRK = RK
C     DT ADDRESS CORRESPONDING TO LOCDTK
      LOCDT = IFWA1 + LOCDTK
C     PICK UP RK, EVAL, D(EVAL)/DK, D2(EVAL)/DK2 AT CURRENT TABLE POINT
      RK = TK(LOCDTK)
      EVAL = TEVAL(LOCDT)
      DLDK = TDLDK(LOCDT)
      D2L = TD2L(LOCDT)
C     COMPUTE STATIONARY PHASE POINT YX AND D(YX)/D(EVAL)
      CALL SPFUNC(1)
C     FILL IN TABLE OF YX
      TYX(LOCDT) = YX
C     TEST FOR AND FIND WAVE FAMILY EDGE(S) BETWEEN PAST AND CURRENT
C     POINTS
      CALL SPEDGE
  100 CONTINUE
C     SKIP IF LAST POINT IN D.T. IS NOT AN EDGE
      IF (TYX(LOCDT) .NE. YXEDG(NWAVES))  GO TO 110
C     NUMBER OF WAVE FAMILIES IS 1 LESS THAN NUMBER OF EDGES
      NWAVES = NWAVES -1
      GO TO 120
C     USE LAST POINT OF D.T. AS AN EDGE
  110 YXEDG(NWAVES+1,MODE) = YX
      EVLEDG(NWAVES+1,MODE) = EVAL
      LIMEDG(NWAVES+1,MODE) = MAXK
      INDEDG(NWAVES+1,MODE) = MAXK
C     SET UP TABLE OF NUMBER OF WAVE FAMILIES IN EACH MODE
  120 NWFTAB(MODE) = NWAVES
C
      MODE = MODE -1
      IF (MODE .GE. MINMOD)  GO TO 10
```

```
  130 CALL TIMER(-16)
      RETURN
      END
      SUBROUTINE SPXDEP
C     X DEPENDENCE OF SIGNAL
C
      COMMON /GRID/ OBSDEP, NOBS, DOBS, OBSMAX, TABOBS(100), ITHOBS
     1,     X, DX, XMIN, NX, ITHX, Y, DY, YMIN, NY, ITHY, MODE1, MODEN
     2,     IVAR, IPRDT, IPPDT(9), XPMAX, YPMAX, IPPPSD, IPREDG
     3,     ISPHAS
C
      COMPLEX XDEP, FSRC
      COMMON // RK, EVAL, DLDK, D2L, PSIO, DPSIO, DPSIB, WAKI, SUPT
     1,     YX, D3L, XI, ETA, INRANG, MODE, MINMOD, MAXMOD, MODES
     2,     MAXK, IFWA1, LOCDT, LOCDTK, ITHWAV, NWAVES, NWFTAB(41)
     3,     HILO, DYX, PYX, PDYX, PEVAL, PRK, FAZ, SGNFZ2, FAZ2
     4,     FAZ3, XDEP, FSRC, VAR, SIGNAL, SIGCUT(800)
      EQUIVALENCE (TEMP1,SIGCUT)
      DIMENSION TEMP1(9,1)
      COMPLEX CAIRY,EXPIB
C
C
C     NEGATIVE OF AIRY FUNCTION ARGUMENT
      A = FAZ2**2 *(.5*X/FAZ3**2)**(2./3.)
C     GET AIRY FUNCTIONS AIRYA=AI(-A) AND AIRYB=BI(-A)
      CALL AIRY(AIRYA, AIRYB, -A)
C     SET UP COMPLEX FORM OF AIRY FUNCTION
      CAIRY = CMPLX(AIRYA, AIRYB*SGNFZ2)
C
      B = X*(FAZ +FAZ2**3/(3.*FAZ3**2))
      EXPIB = CMPLX(COS(B), SIN(B))
C
      XDEP = CAIRY*EXPIB
      RETURN
      END
      SUBROUTINE SP1PNT
C     COMPUTE SIGNAL AT FIELD POINT DEFINED BY X, YX=-Y/X, OBSDEP
C
      COMPLEX XDEP, FSRC
      COMMON // RK, EVAL, DLDK, D2L, PSIO, DPSIO, DPSIB, WAKI, SUPT
     1,     YX, D3L, XI, ETA, INRANG, MODE, MINMOD, MAXMOD, MODES
     2,     MAXK, IFWA1, LOCDT, LOCDTK, ITHWAV, NWAVES, NWFTAB(41)
     3,     HILO, DYX, PYX, PDYX, PEVAL, PRK, FAZ, SGNFZ2, FAZ2
     4,     FAZ3, XDEP, FSRC, VAR, SIGNAL, SIGCUT(800)
      EQUIVALENCE (TEMP1,SIGCUT)
      DIMENSION TEMP1(9,1)
C
C
      CALL TIMER(17)
C     -SIGNAL- IS RUNNING SUM OF CONTRIBUTION FROM EACH MODE AND
C     WAVE FAMILY
      SIGNAL = 0.
C     LOOP FOR EACH MODE
      DO 20  MODE=MINMOD,MAXMOD
C     PICK UP NUMBER OF WAVE FAMILIES FOR THIS MODE
      NWAVES = NWFTAB(MODE)
      DO 10  ITHWAV=1,NWAVES
C     INTERPOLATE IN DISPERSION TABLES AT STATIONARY PHASE POINT
      CALL SPDISP(0)
C     SKIP IF YX IS OUTSIDE THE RANGE OF THIS WAVE FAMILY
      IF (INRANG .EQ. 0)  GO TO 10
C     COMPUTE XI, ETA, PHASE FUNCTION FAZ AND ITS DERIVATIVES FAZ2,FAZ3
```

```
      CALL SPFUNC(0)
C     COMPUTE X DEPENDENCE OF SIGNAL (XDEP)
      CALL SPXDEP
C     COMPUTE SOURCE FUNCTION (FSRC)
      CALL SPSRC
C     COMPUTE VARIABLE-DEPENDENT PART OF SIGNAL, COMBINE WITH XDEP,
C     FSRC AND ADD IT INTO -SIGNAL-
      CALL SPVAR
   10 CONTINUE
   20 CONTINUE
      CALL TIMER(-17)
      RETURN
      END
      SUBROUTINE TRID(C,A,B,D,X,N,MSF)
C     LAST MODIFICATION 3/22/68-K.E.M.
C     TRID IS A TRI-DIAGONAL MATRIX LINEAR EQUATION SOLVER.
C     IF MX=D IS SUCH AN EQJATION, THEN THE I TH ROW OF M IS
C       M(I)=(0,0,...,0,C(I),A(I),BCI),0,...,0,0), WHERE C(1)=B(N)=0.0
C       C,A,B,D AND X ARE VECTORS OF LENGTH N
C       MSF=0 IF M IS NONSINGULAR
C          =1 IF M IS SINGULAR
      DIMENSION C(N),A(N),B(N),D(N),X(N)
      NN=N
      NM=NN-1
C     SCALE RCWS
      DO 30 I=1,NN
      T=AMAX1(ABS(A(I)),ABS(B(I)),ABS(C(I)))
      IF(T) 120,120,20
 20   A(I)=A(I)/T
      B(I)=B(I)/T
      C(I)=C(I)/T
 30   D(I)=D(I)/T
C     ELININATE
      DO 90 I=1,NM
      IF(ABS(A(I))-ABS(C(I+1))) 60,40,40
 40   IF(A(I)) 50,120,50
 50   C(I)=A(I)
      A(I)=B(I)
      B(I)=0.0
      GO TO 80
 60   IF(C(I+1)) 70,120,70
 70   C(I)=C(I+1)
      C(I+1)=A(I)
      A(I)=A(I+1)
      A(I+1)=B(I)
      B(I)=B(I+1)
      B(I+1)=0.0
      T=D(I)
      D(I)=D(I+1)
      D(I+1)=T
 80   T=C(I+1)/C(I)
      A(I+1)=A(I+1)-T*A(I)
      B(I+1)=B(I+1)-T*B(I)
 90   D(I+1)=D(I+1)-T*D(I)
C     BACK SUBSTITUTE
      IF(A(NN)) 100,120,100
 100  X(NN)=D(NN)/A(NN)
      X(NM)=(D(NM)-A(NM)*X(NN))/C(NM)
      DO 110 J=2,NM
      I=NN-J
 110  X(I)=(D(I)-A(I)*X(I+1)-B(I)*X(I+2))/C(I)
C     NORMAL EXIT
```

```
      MSF=0
      RETURN
C     SINGULAR MATRIX EXIT
 120  MSF=1
      RETURN
      END
      SUBROUTINE FIGI(NM,N,T,D,E,E2,IERR)
C
      INTEGER I,N,NM,IERR
      REAL T(NM,3),D(N),E(N),E2(N)
C     REAL SQRT
C
C     GIVEN A NONSYMMETRIC TRIDIAGONAL MATRIX SUCH THAT THE PRODUCTS
C     OF CORRESPONDING PAIRS OF OFF-DIAGONAL ELEMENTS ARE ALL
C     NON-NEGATIVE, THIS SUBROUTINE REDUCES IT TO A SYMMETRIC
C     TRIDIAGONAL MATRIX WITH THE SAME EIGENVALUES.  IF, FURTHER,
C     A ZERO PRODUCT ONLY OCCURS WHEN BOTH FACTORS ARE ZERO,
C     THE REDUCED MATRIX IS SIMILAR TO THE ORIGINAL MATRIX.
C
C     ON INPUT-
C
C        NM MUST BE SET TO THE ROW DIMENSION OF TWO-DIMENSIONAL
C          ARRAY PARAMETERS AS DECLARED IN THE CALLING PROGRAM
C          DIMENSION STATEMENT,
C
C        N IS THE ORDER OF THE MATRIX,
C
C        T CONTAINS THE INPUT MATRIX.  ITS SUBDIAGONAL IS
C          STORED IN THE LAST N-1 POSITIONS OF THE FIRST COLUMN,
C          ITS DIAGONAL IN THE N POSITIONS OF THE SECOND COLUMN,
C          AND ITS SUPERDIAGONAL IN THE FIRST N-1 POSITIONS OF
C          THE THIRD COLUMN.  T(1,1) AND T(N,3) ARE ARBITRARY.
C
C     ON OUTPUT-
C
C        T IS UNALTERED,
C
C        D CONTAINS THE DIAGONAL ELEMENTS OF THE SYMMETRIC MATRIX,
C
C        E CONTAINS THE SUBDIAGONAL ELEMENTS OF THE SYMMETRIC
C          MATRIX IN ITS LAST N-1 POSITIONS.  E(1) IS NOT SET,
C
C        E2 CONTAINS THE SQUARES OF THE CORRESPONDING ELEMENTS OF E.
C          E2 MAY COINCIDE WITH E IF THE SQUARES ARE NOT NEEDED,
C
C        IERR IS SET TO
C          ZERO       FOR NORMAL RETURN,
C          N+I        IF T(I,1)*T(I-1,3) IS NEGATIVE,
C          -(3*N+I)   IF T(I,1)*T(I-1,3) IS ZERO WITH ONE FACTOR
C                     NON-ZERO.  IN THIS CASE, THE EIGENVECTORS OF
C                     THE SYMMETRIC MATRIX ARE NOT SIMPLY RELATED
C                     TO THOSE OF  T  AND SHOULD NOT BE SOUGHT.
C
C     QUESTIONS AND COMMENTS SHOULD BE DIRECTED TO B. S. GARBOW,
C     APPLIED MATHEMATICS DIVISION, ARGONNE NATIONAL LABORATORY
C
C     ------------------------------------------------------------------
C
      IERR = 0
C
      DO 100 I = 1, N
         IF (I .EQ. 1) GO TO 90
```

```
         E2(I) = T(I,1) * T(I-1,3)
         IF (E2(I)) 1000, 60, 80
   60    IF (T(I,1) .EQ. 0.0 .AND. T(I-1,3) .EQ. 0.0) GO TO 80
C     ********** SET ERROR -- PRODUCT OF SOME PAIR OF OFF-DIAGONAL
C                ELEMENTS IS ZERO WITH ONE MEMBER NON-ZERO **********
         IERR = -(3 * N + I)
   80    E(I) = SQRT(E2(I))
   90    D(I) = T(I,2)
  100 CONTINUE
C
      GO TO 1001
C     ********** SET ERROR -- PRODUCT OF SOME PAIR OF OFF-DIAGONAL
C                ELEMENTS IS NEGATIVE **********
 1000 IERR = N + I
 1001 RETURN
C     ********** LAST CARD OF FIGI **********
      END
      SUBROUTINE RATQR(N,EPS1,D,E,E2,M,W,IND,BD,TYPE,IDEF,IERR)
C
      INTEGER I,J,K,M,N,II,JJ,K1,IDEF,IERR,JDEF
      REAL D(N),E(N),E2(N),W(N),BD(N)
      REAL F,P,Q,R,S,EP,QP,ERR,TOT,EPS1,DELTA,MACHEP
C     REAL ABS,AMIN1
      INTEGER IND(M)
      LOGICAL TYPE
C
C     THIS SUBROUTINE IS A TRANSLATION OF THE ALGOL PROCEDURE RATQR,
C     NUM. MATH. 11, 264-272(1968) BY REINSCH AND BAUER.
C     HANDBOOK FOR AUTO. COMP., VOL.II-LINEAR ALGEBRA, 257-265(1971).
C
C     THIS SUBROUTINE FINDS THE ALGEBRAICALLY SMALLEST OR LARGEST
C     EIGENVALUES OF A SYMMETRIC TRIDIAGONAL MATRIX BY THE
C     RATIONAL QR METHOD WITH NEWTON CORRECTIONS.
C
C     ON INPUT-
C
C        N IS THE ORDER OF THE MATRIX,
C
C        EPS1 IS A THEORETICAL ABSOLUTE ERROR TOLERANCE FOR THE
C          COMPUTED EIGENVALUES.  IF THE INPUT EPS1 IS NON-POSITIVE,
C          OR INDEED SMALLER THAN ITS DEFAULT VALUE, IT IS RESET
C          AT EACH ITERATION TO THE RESPECTIVE DEFAULT VALUE,
C          NAMELY, THE PRODUCT OF THE RELATIVE MACHINE PRECISION
C          AND THE MAGNITUDE OF THE CURRENT EIGENVALUE ITERATE.
C          THE THEORETICAL ABSOLUTE ERROR IN THE K-TH EIGENVALUE
C          IS USUALLY NOT GREATER THAN K TIMES EPS1,
C
C        D CONTAINS THE DIAGONAL ELEMENTS OF THE INPUT MATRIX,
C
C        E CONTAINS THE SUBDIAGONAL ELEMENTS OF THE INPUT MATRIX
C          IN ITS LAST N-1 POSITIONS.  E(1) IS ARBITRARY,
C
C        E2 CONTAINS THE SQUARES OF THE CORRESPONDING ELEMENTS OF E.
C          E2(1) IS ARBITRARY,
C
C        M IS THE NUMBER OF EIGENVALUES TO BE FOUND,
C
C        IDEF SHOULD BE SET TO 1 IF THE INPUT MATRIX IS KNOWN TO BE
C          POSITIVE DEFINITE, TO -1 IF THE INPUT MATRIX IS KNOWN TO
C          BE NEGATIVE DEFINITE, AND TO 0 OTHERWISE,
C
C        TYPE SHOULD BE SET TO .TRUE. IF THE SMALLEST EIGENVALUES
```

```
C          ARE TO BE FOUND, AND TO .FALSE. IF THE LARGEST EIGENVALUES
C          ARE TO BE FOUND.
C
C     ON OUTPUT-
C
C        EPS1 IS UNALTERED UNLESS IT HAS BEEN RESET TO ITS
C          (LAST) DEFAULT VALUE,
C
C        D AND E ARE UNALTERED (UNLESS W OVERWRITES D),
C
C        ELEMENTS OF E2, CORRESPONDING TO ELEMENTS OF E REGARDED
C          AS NEGLIGIBLE, HAVE BEEN REPLACED BY ZERO CAUSING THE
C          MATRIX TO SPLIT INTO A DIRECT SUM OF SUBMATRICES.
C          E2(1) IS SET TO 0.0 IF THE SMALLEST EIGENVALUES HAVE BEEN
C          FOUND, AND TO 2.0 IF THE LARGEST EIGENVALUES HAVE BEEN
C          FOUND.  E2 IS OTHERWISE UNALTERED (UNLESS OVERWRITTEN BY BD),
C
C        W CONTAINS THE M ALGEBRAICALLY SMALLEST EIGENVALUES IN
C          ASCENDING ORDER, OR THE M LARGEST EIGENVALUES IN
C          DESCENDING ORDER.  IF AN ERROR EXIT IS MADE BECAUSE OF
C          AN INCORRECT SPECIFICATION OF IDEF, NO EIGENVALUES
C          ARE FOUND.  IF THE NEWTON ITERATES FOR A PARTICULAR
C          EIGENVALUE ARE NOT MONOTONE, THE BEST ESTIMATE OBTAINED
C          IS RETURNED AND IERR IS SET.  W MAY COINCIDE WITH D,
C
C        IND CONTAINS IN ITS FIRST M POSITIONS THE SUBMATRIX INDICES
C          ASSOCIATED WITH THE CORRESPONDING EIGENVALUES IN W --
C          1 FOR EIGENVALUES BELONGING TO THE FIRST SUBMATRIX FROM
C          THE TOP, 2 FOR THOSE BELONGING TO THE SECOND SUBMATRIX, ETC.,
C
C        BD CONTAINS REFINED BOUNDS FOR THE THEORETICAL ERRORS OF THE
C          CORRESPONDING EIGENVALUES IN W.  THESE BOUNDS ARE USUALLY
C          WITHIN THE TOLERANCE SPECIFIED BY EPS1.  BD MAY COINCIDE
C          WITH E2,
C
C        IERR IS SET TO
C          ZERO       FOR NORMAL RETURN,
C          6*N+1      IF  IDEF  IS SET TO 1 AND  TYPE  TO .TRUE.
C                     WHEN THE MATRIX IS NOT POSITIVE DEFINITE, OR
C                     IF  IDEF  IS SET TO -1 AND  TYPE  TO .FALSE.
C                     WHEN THE MATRIX IS NOT NEGATIVE DEFINITE,
C          5*N+K      IF SUCCESSIVE ITERATES TO THE K-TH EIGENVALUE
C                     ARE NOT MONOTONE INCREASING, WHERE K REFERS
C                     TO THE LAST SUCH OCCURRENCE.
C
C     NOTE THAT SUBROUTINE TQL1 OR IMTQL1 IS GENERALLY FASTER THAN
C     RATQR, IF MORE THAN N/4 EIGENVALUES ARE TO BE FOUND.  ALSO,
C     BISECT IS GENERALLY FASTER IF THE EIGENVALUES ARE CLUSTERED.
C
C     QUESTIONS AND COMMENTS SHOULD BE DIRECTED TO B. S. GARBOW,
C     APPLIED MATHEMATICS DIVISION, ARGONNE NATIONAL LABORATORY
C
C     ------------------------------------------------------------------
C
C     ********** MACHEP IS A MACHINE DEPENDENT PARAMETER SPECIFYING
C                THE RELATIVE PRECISION OF FLOATING POINT ARITHMETIC.
C
C                **********
      MACHEP = 2.**(-47)
C
      IERR = 0
      JDEF = IDEF
```

```
C     ********** COPY D ARRAY INTO W **********
      DO 20 I = 1, N
   20 W(I) = D(I)
C
      IF (TYPE) GO TO 40
      J = 1
      GO TO 400
   40 ERR = 0.0
      S = 0.0
C     ********** LOOK FOR SMALL SUB-DIAGONAL ENTRIES AND DEFINE
C                INITIAL SHIFT FROM LOWER GERSCHGORIN BOUND.
C                COPY E2 ARRAY INTO BD **********
      TOT = W(1)
      Q = 0.0
      J = 0
C
      DO 100 I = 1, N
         P = Q
         IF (I .EQ. 1) GO TO 60
         IF (P .GT. MACHEP * (ABS(D(I)) + ABS(D(I-1)))) GO TO 80
   60    E2(I) = 0.0
         J = J + 1
   80    BD(I) = E2(I)
         IND(I) = J
         Q = 0.0
         IF (I .NE. N) Q = ABS(E(I+1))
         TOT = AMIN1(W(I)-P-Q,TOT)
  100 CONTINUE
C
      IF (JDEF .EQ. 1 .AND. TOT .LT. 0.0) GO TO 140
C
      DO 110 I = 1, N
  110 W(I) = W(I) - TOT
C
      GO TO 160
  140 TOT = 0.0
C
  160 DO 360 K = 1, M
C     ********** NEXT QR TRANSFORMATION **********
  180    TOT = TOT + S
         DELTA = W(N) - S
         I = N
         F = ABS(MACHEP*TOT)
         IF (EPS1 .LT. F) EPS1 = F
         IF (DELTA .GT. EPS1) GO TO 190
         IF (DELTA .LT. (-EPS1)) GO TO 1000
         GO TO 300
C     ********** REPLACE SMALL SUB-DIAGONAL SQUARES BY ZERO
C                TO REDUCE THE INCIDENCE OF UNDERFLOWS **********
  190    IF (K .EQ. N) GO TO 210
         K1 = K + 1
         DO 200 J = K1, N
            IF (BD(J) .LE. (MACHEP*(W(J)+W(J-1))) ** 2) BD(J) = 0.0
  200    CONTINUE
C
  210    F = BD(N) / DELTA
         QP = DELTA + F
         P = 1.0
         IF (K .EQ. N) GO TO 260
         K1 = N - K
C     ********** FOR I=N-1 STEP -1 UNTIL K DO -- **********
         DO 240 II = 1, K1
```

```
            I = N - II
            Q = W(I) - S - F
            R = Q / QP
            P = P * R + 1.0
            EP = F * R
            W(I+1) = QP + EP
            DELTA = Q - EP
            IF (DELTA .GT. EPS1) GO TO 220
            IF (DELTA .LT. (-EPS1)) GO TO 1000
            GO TO 300
  220       F = BD(I) / Q
            QP = DELTA + F
            BD(I+1) = QP * EP
  240    CONTINUE
C
  260    W(K) = QP
         S = QP / P
         IF (TOT + S .GT. TOT) GO TO 180
C     ********** SET ERROR -- IRREGULAR END OF ITERATION.
C                DEFLATE MINIMUM DIAGONAL ELEMENT **********
         IERR = 5 * N + K
         S = 0.0
         DELTA = QP
C
         DO 280 J = K, N
            IF (W(J) .GT. DELTA) GO TO 280
            I = J
            DELTA = W(J)
  280    CONTINUE
C     ********** CONVERGENCE **********
  300    IF (I .LT. N) BD(I+1) = BD(I) * F / QP
         II = IND(I)
         IF (I .EQ. K) GO TO 340
         K1 = I - K
C     ********** FOR J=I-1 STEP -1 UNTIL K DO -- **********
         DO 320 JJ = 1, K1
            J = I - JJ
            W(J+1) = W(J) - S
            BD(J+1) = BD(J)
            IND(J+1) = IND(J)
  320    CONTINUE
C
  340    W(K) = TOT
         ERR = ERR + ABS(DELTA)
         BD(K) = ERR
         IND(K) = II
  360 CONTINUE
C
      IF (TYPE) GO TO 1001
      F = BD(1)
      E2(1) = 2.0
      BD(1) = F
      J = 2
C     ********** NEGATE ELEMENTS OF W FOR LARGEST VALUES **********
  400 DO 500 I = 1, N
  500 W(I) = -W(I)
C
      JDEF = -JDEF
      GO TO (40,1001), J
C     ********** SET ERROR -- IDEF SPECIFIED INCORRECTLY **********
 1000 IERR = 6 * N + 1
 1001 RETURN
```

```
C     ********** LAST CARD OF RATQR **********
      END
      SUBROUTINE TINVIT(NM,N,D,E,E2,M,W,IND,Z,
     X                  IERR,RV1,RV2,RV3,RV4,RV6)
C
      INTEGER I,J,M,N,P,Q,R,S,II,IP,JJ,NM,ITS,TAG,IERR,GROUP
      REAL D(N),E(N),E2(N),W(M),Z(NM,M),
     X       RV1(N),RV2(N),RV3(N),RV4(N),RV6(N)
      REAL U,V,UK,XU,X0,X1,EPS2,EPS3,EPS4,NORM,ORDER,MACHEP
C     REAL SQRT,ABS,FLOAT
      INTEGER IND(M)
C
C     THIS SUBROUTINE IS A TRANSLATION OF THE INVERSE ITERATION TECH-
C     NIQUE IN THE ALGOL PROCEDURE TRISTURM BY PETERS AND WILKINSON.
C     HANDBOOK FOR AUTO. COMP., VOL.II-LINEAR ALGEBRA, 418-439(1971).
C
C     THIS SUBROUTINE FINDS THOSE EIGENVECTORS OF A TRIDIAGONAL
C     SYMMETRIC MATRIX CORRESPONDING TO SPECIFIED EIGENVALUES,
C     USING INVERSE ITERATION.
C
C     ON INPUT-
C
C        NM MUST BE SET TO THE ROW DIMENSION OF TWO-DIMENSIONAL
C          ARRAY PARAMETERS AS DECLARED IN THE CALLING PROGRAM
C          DIMENSION STATEMENT,
C
C        N IS THE ORDER OF THE MATRIX,
C
C        D CONTAINS THE DIAGONAL ELEMENTS OF THE INPUT MATRIX,
C
C        E CONTAINS THE SUBDIAGONAL ELEMENTS OF THE INPUT MATRIX
C          IN ITS LAST N-1 POSITIONS.  E(1) IS ARBITRARY,
C
C        E2 CONTAINS THE SQUARES OF THE CORRESPONDING ELEMENTS OF E,
C          WITH ZEROS CORRESPONDING TO NEGLIGIBLE ELEMENTS OF E.
C          E(I) IS CONSIDERED NEGLIGIBLE IF IT IS NOT LARGER THAN
C          THE PRODUCT OF THE RELATIVE MACHINE PRECISION AND THE SUM
C          OF THE MAGNITUDES OF D(I) AND D(I-1).  E2(1) MUST CONTAIN
C            0.0 IF THE EIGENVALUES ARE IN ASCENDING ORDER, OR
C            2.0 IF THE EIGENVALUES ARE IN DESCENDING ORDER.
C          IF  BISECT  OR  RATQR  HAS BEEN USED TO FIND THE EIGENVALUES,
C          THEIR OUTPUT E2 ARRAY IS EXACTLY WHAT IS EXPECTED HERE,
C
C        M IS THE NUMBER OF SPECIFIED EIGENVALUES,
C
C        W CONTAINS THE M EIGENVALUES IN ASCENDING OR DESCENDING ORDER,
C
C        IND CONTAINS IN ITS FIRST M POSITIONS THE SUBMATRIX INDICES
C          ASSOCIATED WITH THE CORRESPONDING EIGENVALUES IN W --
C          1 FOR EIGENVALUES BELONGING TO THE FIRST SUBMATRIX FROM
C          THE TOP, 2 FOR THOSE BELONGING TO THE SECOND SUBMATRIX, ETC.
C
C      ON OUTPUT-
C
C        ALL INPUT ARRAYS ARE UNALTERED,
C
C        Z CONTAINS THE ASSOCIATED SET OF ORTHONORMAL EIGENVECTORS.
C          ANY VECTOR WHICH FAILS TO CONVERGE IS SET TO ZERO,
C
C        IERR IS SET TO
C          ZERO       FOR NORMAL RETURN,
C          -R         IF THE EIGENVECTOR CORRESPONDING TO THE R-TH
```

```
C                     EIGENVALUE FAILS TO CONVERGE IN 5 ITERATIONS,
C
C        RV1, RV2, RV3, RV4, AND RV6 ARE TEMPORARY STORAGE ARRAYS.
C
C     QUESTIONS AND COMMENTS SHOULD BE DIRECTED TO B. S. GARBOW,
C     APPLIED MATHEMATICS DIVISION, ARGONNE NATIONAL LABORATORY
C
C     ------------------------------------------------------------------
C
C     ********** MACHEP IS A MACHINE DEPENDENT PARAMETER SPECIFYING
C                THE RELATIVE PRECISION OF FLOATING POINT ARITHMETIC.
C
C                **********
      MACHEP = 2.**(-47)
C
      IERR = 0
      TAG = 0
      ORDER = 1.0 - E2(1)
      Q = 0
C     ********** ESTABLISH AND PROCESS NEXT SUBMATRIX **********
  100 P = Q + 1
C
      DO 120 Q = P, N
         IF (Q .EQ. N) GO TO 140
         IF (E2(Q+1) .EQ. 0.0) GO TO 140
  120 CONTINUE
C     ********** FIND VECTORS BY INVERSE ITERATION **********
  140 TAG = TAG + 1
      S = 0
C
      DO 920 R = 1, M
         IF (IND(R) .NE. TAG) GO TO 920
         ITS = 1
         X1 = W(R)
         IF (S .NE. 0) GO TO 510
C     ********** CHECK FOR ISOLATED ROOT **********
         XU = 1.0
         IF (P .NE. Q) GO TO 490
         RV6(P) = 1.0
         GO TO 870
  490    NORM = ABS(D(P))
         IP = P + 1
C
         DO 500 I = IP, Q
  500    NORM = NORM + ABS(D(I)) + ABS(E(I))
C     ********** EPS2 IS THE CRITERION FOR GROUPING,
C                EPS3 REPLACES ZERO PIVOTS AND EQUAL
C                ROOTS ARE MODIFIED BY EPS3,
C                EPS4 IS TAKEN VERY SMALL TO AVOID OVERFLOW **********
         EPS2 = 1.0E-3 * NORM
         EPS3 = MACHEP * NORM
         UK = FLOAT(Q-P+1)
         EPS4 = UK * EPS3
         UK = EPS4 / SQRT(UK)
         S = P
  505    GROUP = 0
         GO TO 520
C     ********** LOOK FOR CLOSE OR COINCIDENT ROOTS **********
  510    IF (ABS(X1-X0) .GE. EPS2) GO TO 505
         GROUP = GROUP + 1
         IF (ORDER * (X1 - X0) .LE. 0.0) X1 = X0 + ORDER * EPS3
C     ********** ELIMINATION WITH INTERCHANGES AND
```

```
C                 INITIALIZATION OF VECTOR **********
  520    V = 0.0
C
         DO 580 I = P, Q
            RV6(I) = UK
            IF (I .EQ. P) GO TO 560
            IF (ABS(E(I)) .LT. ABS(U)) GO TO 540
C     ********** WARNING -- A DIVIDE CHECK MAY OCCUR HERE IF
C                 E2 ARRAY HAS NOT BEEN SPECIFIED CORRECTLY **********
            XU = U / E(I)
            RV4(I) = XU
            RV1(I-1) = E(I)
            RV2(I-1) = D(I) - X1
            RV3(I-1) = 0.0
            IF (I .NE. Q) RV3(I-1) = E(I+1)
            U = V - XU * RV2(I-1)
            V = -XU * RV3(I-1)
            GO TO 580
  540       XU = E(I) / U
            RV4(I) = XU
            RV1(I-1) = U
            RV2(I-1) = V
            RV3(I-1) = 0.0
  560       U = D(I) - X1 - XU * V
            IF (I .NE. Q) V = E(I+1)
  580    CONTINUE
C
         IF (U .EQ. 0.0) U = EPS3
         RV1(Q) = U
         RV2(Q) = 0.0
         RV3(Q) = 0.0
C     ********** BACK SUBSTITUTION
C                 FOR I=Q STEP -1 UNTIL P DO -- **********
  600    DO 620 II = P, Q
            I = P + Q - II
            RV6(I) = (RV6(I) - U * RV2(I) - V * RV3(I)) / RV1(I)
            V = U
            U = RV6(I)
  620    CONTINUE
C     ********** ORTHOGONALIZE WITH RESPECT TO PREVIOUS
C                 MEMBERS OF GROUP **********
         IF (GROUP .EQ. 0) GO TO 700
         J = R
C
         DO 680 JJ = 1, GROUP
  630       J = J - 1
            IF (IND(J) .NE. TAG) GO TO 630
            XU = 0.0
C
            DO 640 I = P, Q
  640       XU = XU + RV6(I) * Z(I,J)
C
            DO 660 I = P, Q
  660       RV6(I) = RV6(I) - XU * Z(I,J)
C
  680    CONTINUE
C
  700    NORM = 0.0
C
         DO 720 I = P, Q
  720    NORM = NORM + ABS(RV6(I))
C
```

```
         IF (NORM .GE. 1.0) GO TO 840
C     ********** FORWARD SUBSTITUTION **********
         IF (ITS .EQ. 5) GO TO 830
         IF (NORM .NE. 0.0) GO TO 740
         RV6(S) = EPS4
         S = S + 1
         IF (S .GT. Q) S = P
         GO TO 780
  740    XU = EPS4 / NORM
C
         DO 760 I = P, Q
  760    RV6(I) = RV6(I) * XU
C     ********** ELIMINATION OPERATIONS ON NEXT VECTOR
C                ITERATE **********
  780    DO 820 I = IP, Q
            U = RV6(I)
C     ********** IF RV1(I-1) .EQ. E(I), A ROW INTERCHANGE
C                WAS PERFORMED EARLIER IN THE
C                TRIANGULARIZATION PROCESS **********
            IF (RV1(I-1) .NE. E(I)) GO TO 800
            U = RV6(I-1)
            RV6(I-1) = RV6(I)
  800       RV6(I) = U - RV4(I) * RV6(I-1)
  820    CONTINUE
C
         ITS = ITS + 1
         GO TO 600
C     ********** SET ERROR -- NON-CONVERGED EIGENVECTOR **********
  830    IERR = -R
         XU = 0.0
         GO TO 870
C     ********** NORMALIZE SO THAT SUM OF SQUARES IS
C                1 AND EXPAND TO FULL ORDER **********
  840    U = 0.0
C
         DO 860 I = P, Q
  860    U = U + RV6(I)**2
C
         XU = 1.0 / SQRT(U)
C
  870    DO 880 I = 1, N
  880    Z(I,R) = 0.0
C
         DO 900 I = P, Q
  900    Z(I,R) = RV6(I) * XU
C
         X0 = X1
  920 CONTINUE
C
      IF (Q .LT. N) GO TO 100
      RETURN
C     ********** LAST CARD OF TINVIT **********
      END
      SUBROUTINE BAKVEC(NM,N,T,E,M,Z,IERR)
C
      INTEGER I,J,M,N,NM,IERR
      REAL T(NM,3),E(N),Z(NM,M)
C
C     THIS SUBROUTINE FORMS THE EIGENVECTORS OF A NONSYMMETRIC
C     TRIDIAGONAL MATRIX BY BACK TRANSFORMING THOSE OF THE
C     CORRESPONDING SYMMETRIC MATRIX DETERMINED BY  FIGI.
C
```

```
C     ON INPUT-
C
C        NM MUST BE SET TO THE ROW DIMENSION OF TWO-DIMENSIONAL
C          ARRAY PARAMETERS AS DECLARED IN THE CALLING PROGRAM
C          DIMENSION STATEMENT,
C
C        N IS THE ORDER OF THE MATRIX,
C
C        T CONTAINS THE NONSYMMETRIC MATRIX.  ITS SUBDIAGONAL IS
C          STORED IN THE LAST N-1 POSITIONS OF THE FIRST COLUMN,
C          ITS DIAGONAL IN THE N POSITIONS OF THE SECOND COLUMN,
C          AND ITS SUPERDIAGONAL IN THE FIRST N-1 POSITIONS OF
C          THE THIRD COLUMN.  T(1,1) AND T(N,3) ARE ARBITRARY,
C
C        E CONTAINS THE SUBDIAGONAL ELEMENTS OF THE SYMMETRIC
C          MATRIX IN ITS LAST N-1 POSITIONS.  E(1) IS ARBITRARY,
C
C        M IS THE NUMBER OF EIGENVECTORS TO BE BACK TRANSFORMED,
C
C        Z CONTAINS THE EIGENVECTORS TO BE BACK TRANSFORMED
C          IN ITS FIRST M COLUMNS.
C
C     ON OUTPUT-
C
C        T IS UNALTERED,
C
C        E IS DESTROYED,
C
C        Z CONTAINS THE TRANSFORMED EIGENVECTORS
C          IN ITS FIRST M COLUMNS,
C
C        IERR IS SET TO
C          ZERO       FOR NORMAL RETURN,
C          2*N+1      IF E(I) IS ZERO WITH T(I,1) OR T(I-1,3) NON-ZERO.
C                     IN THIS CASE, THE SYMMETRIC MATRIX IS NOT SIMILAR
C                     TO THE ORIGINAL MATRIX, AND THE EIGENVECTORS
C                     CANNOT BE FOUND BY THIS PROGRAM.
C
C     QUESTIONS AND COMMENTS SHOULD BE DIRECTED TO B. S. GARBOW,
C     APPLIED MATHEMATICS DIVISION, ARGONNE NATIONAL LABORATORY
C
C     ------------------------------------------------------------------
C
      IERR = 0
      E(1) = 1.0
      IF (N .EQ. 1) GO TO 1001
C
      DO 100 I = 2, N
         IF (E(I) .NE. 0.0) GO TO 80
         IF (T(I,1) .NE. 0.0 .OR. T(I-1,3) .NE. 0.0) GO TO 1000
         E(I) = 1.0
         GO TO 100
   80    E(I) = E(I-1) * E(I) / T(I-1,3)
  100 CONTINUE
C
      DO 120 J = 1, M
C
         DO 120 I = 2, N
         Z(I,J) = Z(I,J) * E(I)
  120 CONTINUE
C
      GO TO 1001
```

```
C     ********** SET ERROR -- EIGENVECTORS CANNOT BE
C                FOUND BY THIS PROGRAM **********
 1000 IERR = 2 * N + I
 1001 RETURN
C     ********** LAST CARD OF BAKVEC **********
      END
      SUBROUTINE FOURT(DATA,NN,NDIM,ISIGN,IFORM,WORK)
C
C     THE COOLEY-TUKEY FAST FOURIER TRANSFORM IN USASI BASIC FORTRAN
C
C     TRANSFORM(J1,J2,,,,) = SUM(DATA(I1,I2,,,,)*W1**((I2-1)*(J2-1))
C                                *W2**((I2-1)*(J2-1))*,,,),
C     WHERE I1 AND J1 RUN FROM 1 TO NN(1) AND W1=EXP(ISIGN*2*PI=
C     SQRT(-1)/NN(1)), ETC.  THERE IS NO LIMIT ON THE DIMENSIONALITY
C     (NUMBER OF SUBSCRIPTS) OF THE DATA ARRAY.  IF AN INVERSE
C     TRANSFORM (ISIGN=+1) IS PERFORMED UPON AN ARRAY OF TRANSFORMED
C     (ISIGN=-1) DATA, THE ORIGINAL DATA WILL REAPPEAR.
C     MULTIPLIED BY NN(1)*NN(2)*,,,  THE ARRAY OF INPUT DATA MUST BE
C     IN COMPLEX FORMAT.  HOWEVER, IF ALL IMAGINARY PARTS ARE ZERO (I.E.
C     THE DATA ARE DISGUISED REAL) RUNNING TIME IS CUT UP TO FORTY PER-
C     CENT.  (FOR FASTEST TRANSFORM OF REAL DATA, NN(1) SHOULD BE EVEN.)
C     THE TRANSFORM VALUES ARE ALWAYS COMPLEX AND ARE RETURNED IN THE
C     ORIGINAL ARRAY OF DATA, REPLACING THE INPUT DATA.  THE LENGTH
C     OF EACH DIMENSION OF THE DATA ARRAY MAY BE ANY INTEGER.  THE
C     PROGRAM RUNS FASTER ON COMPOSITE INTEGERS THAN ON PRIMES, AND IS
C     PARTICULARLY FAST ON NUMBERS RICH IN FACTORS OF TWO.
C
C     TIMING IS IN FACT GIVEN BY THE FOLLOWING FORMULA.  LET NTOT BE THE
C     TOTAL NUMBER OF POINTS (REAL OR COMPLEX) IN THE DATA ARRAY, THAT
C     IS, NTOT=NN(1)*NN(2)*...  DECOMPOSE NTOT INTO ITS PRIME FACTORS,
C     SUCH AS 2**K2 * 3**K3 * 5**K5 * ...  LET SUM2 BE THE SUM OF ALL
C     THE FACTORS OF TWO IN NTOT, THAT IS, SUM2 = 2*K2.  LET SUMF BE
C     THE SUM OF ALL OTHER FACTORS OF NTOT, THAT IS, SUMF = 3*K3+5*K5+..
C     THE TIME TAKEN BY A MULTIDIMENSIONAL TRANSFORM ON THESE NTOT DATA
C     IS T = T0 + NTOT*(T1+T2*SUM2+T3*SUMF).  ON THE CDC 3300 (FLOATING
C     POINT ADD TIME = SIX MICROSECONDS), T = 3000 + NTOT*(600+40*SUM2+
C     175*SUMF) MICROSECONDS ON COMPLEX DATA.
C
C     IMPLEMENTATION OF THE DEFINITION BY SUMMATION WILL RUN IN A TIME
C     PROPORTIONAL TO NTOT*(NN(1)+NN(2)+...).  FOR HIGHLY COMPOSITE NTOT
C     THE SAVINGS OFFERED BY THIS PROGRAM CAN BE DRAMATIC.  A ONE-DIMEN-
C     SIONAL ARRAY 4000 IN LENGTH WILL BE TRANSFORMED IN 4000*(600+
C     40*(2+2+2+2+2)+175*(5+5+5)) = 14.5 SECONDS VERSUS ABOUT 4000*
C     4000*175 = 2800 SECONDS FOR THE STRAIGHTFORWARD TECHNIQUE.
C
C     THE FAST FOURIER TRANSFORM PLACES THREE RESTRICTIONS UPON THE
C     DATA.
C     1.  THE NUMBER OF INPUT DATA AND THE NUMBER OF TRANSFORM VALUES
C     MUST BE THE SAME.
C     2.  BOTH THE INPUT DATA AND THE TRANSFORM VALUES MUST REPRESENT
C     EQUISPACED POINTS IN THEIR RESPECTIVE DOMAINS OF TIME AND
C     FREQUENCY.  CALLING THESE SPACINGS DELTAT AND DELTAF, IT MUST BE
C     TRUE THAT DELTAF=2*PI/(NN(I)*DELTAT).  OF COURSE, DELTAT NEED NOT
C     BE THE SAME FOR EVERY DIMENSION.
C     3.  CONCEPTUALLY AT LEAST, THE INPUT DATA AND THE TRANSFORM OUTPUT
C     REPRESENT SINGLE CYCLES OF PERIODIC FUNCTIONS.
C
C     THE CALLING SEQUENCE IS--
C     CALL FOURT(DATA,NN,NDIM,ISIGN,IFORM,WORK)
C
C     DATA IS THE ARRAY USED TO HOLD THE REAL AND IMAGINARY PARTS
C     OF THE DATA ON INPUT AND THE TRANSFORM VALUES ON OUTPUT.  IT
```

```
C     IS A MULTIDIMENSIONAL FLOATING POINT ARRAY, WITH THE REAL AND
C     IMAGINARY PARTS OF A DATUM STORED IMMEDIATELY ADJACENT IN STORAGE
C     (SUCH AS FORTRAN IV PLACES THEM).  NORMAL FORTRAN ORDERING IS
C     EXPECTED, THE FIRST SUBSCRIPT CHANGING FASTEST.  THE DIMENSIONS
C     ARE GIVEN IN THE INTEGER ARRAY NN, OF LENGTH NDIM.  ISIGN IS -1
C     TO INDICATE A FORWARD TRANSFORM (EXPONENTIAL SIGN IS -) AND +1
C     FOR AN INVERSE TRANSFORM (SIGN IS +).  IFORM IS +1 IF THE DATA ARE
C     COMPLEX, 0 IF THE DATA ARE REAL.  IF IT IS 0, THE IMAGINARY
C     PARTS OF THE DATA MUST BE SET TO ZERO.  AS EXPLAINED ABOVE, THE
C     TRANSFORM VALUES ARE ALWAYS COMPLEX AND ARE STORED IN ARRAY DATA.
C     WORK IS AN ARRAY USED FOR WORKING STORAGE.  IT IS FLOATING POINT
C     REAL, ONE DIMENSIONAL OF LENGTH EQUAL TO TWICE THE LARGEST ARRAY
C     DIMENSION NN(I) THAT IS NOT A POWER OF TWO.  IF ALL NN(I) ARE
C     POWERS OF TWO, IT IS NOT NEEDED AND MAY BE REPLACED BY ZERO IN THE
C     CALLING SEQUENCE.  THUS, FOR A ONE-DIMENSIONAL ARRAY, NN(1) ODD,
C     WORK OCCUPIES AS MANY STORAGE LOCATIONS AS DATA.  IF SUPPLIED,
C     WORK MUST NOT BE THE SAME ARRAY AS DATA.  ALL SUBSCRIPTS OF ALL
C     ARRAYS BEGIN AT ONE.
C
C     EXAMPLE 1.  THREE-DIMENSIONAL FORWARD FOURIER TRANSFORM OF A
C     COMPLEX ARRAY DIMENSIONED 32 BY 25 BY 13 IN FORTRAN IV.
C     DIMENSION DATA(32,25,13),WORK(50),NN(3)
C     COMPLEX DATA
C     DATA NN/32,25,13/
C     DO 1 I=1,32
C     DO 1 J=1,25
C     DO 1 K=1,13
C  1  DATA(I,J,K)=COMPLEX VALUE
C     CALL FOURT(DATA,NN,3,-1,1,WORK)
C
C     EXAMPLE 2.  ONE-DIMENSIONAL FORWARD TRANSFORM OF A REAL ARRAY OF
C     LENGTH 64 IN FORTRAN II,
C     DIMENSION DATA(2,64)
C     DO 2 I=1,64
C     DATA(1,I)=REAL PART
C  2  DATA(2,I)=0.
C     CALL FOURT(DATA,64,1,-1,0,0)
C
C     THERE ARE NO ERROR MESSAGES OR ERROR HALTS IN THIS PROGRAM.  THE
C     PROGRAM RETURNS IMMEDIATELY IF NDIM OR ANY NN(I) IS LESS THAN ONE.
C
C     PROGRAM BY NORMAN BRENNER FROM THE BASIC PROGRAM BY CHARLES
C     RADER,  JUNE 1967.  THE IDEA FOR THE DIGIT REVERSAL WAS
C     SUGGESTED BY RALPH ALTER.
C
C     THIS IS THE FASTEST AND MOST VERSATILE VERSION OF THE FFT KNOWN
C     TO THE AUTHOR.  A PROGRAM CALLED FOUR2 IS AVAILABLE THAT ALSO
C     PERFORMS THE FAST FOURIER TRANSFORM AND IS WRITTEN IN USASI BASIC
C     FORTRAN.  IT IS ABOUT ONE THIRD AS LONG AND RESTRICTS THE
C     DIMENSIONS OF THE INPUT ARRAY (WHICH MUST BE COMPLEX) TO BE POWERS
C     OF TWO.  ANOTHER PROGRAM, CALLED FOUR1, IS ONE TENTH AS LONG AND
C     RUNS TWO THIRDS AS FAST ON A ONE-DIMENSIONAL COMPLEX ARRAY WHOSE
C     LENGTH IS A POWER OF TWO.
C
C     REFERENCE--
C     IEEE AUDIO TRANSACTIONS (JUNE 1967), SPECIAL ISSUE ON THE FFT.
C
      DIMENSION DATA(1),NN(1),IFACT(32),WORK(1)
      DATA TWOPI/6.2831853071796/,RTHLF/0.70710678118655/
      IF(NDIM-1)920,1,1
1     NTOT=2
      DO 2 IDIM=1,NDIM
```

```
      IF(NN(IDIM))920,920,2
2     NTOT=NTOT*NN(IDIM)
C
C     MAIN LOOP FOR EACH DIMENSION
C
      NP1=2
      DO 910  IDIM=1,NDIM
      N=NN(IDIM)
      NP2=NP1*N
      IF(N-1)920,900,5
C
C     IS  N  A  POWER  OF  TWO  AND  IF  NOT,  WHAT  ARE  ITS  FACTORS
C
5     M=N
      NTWO=NP1
      IF=1
      IDIV=2
10    IQUOT=M/IDIV
      IREM=M-IDIV*IQUOT
      IF(IQUOT-IDIV)50,11,11
11    IF(IREM)20,12,20
12    NTWO=NTWO+NTWO
      IFACT(IF)=IDIV
      IF=IF+1
      M=IQUOT
      GO TO 10
20    IDIV=3
      INON2=IF
30    IQUOT=M/IDIV
      IREM=M-IDIV*IQUOT
      IF(IQUOT-IDIV)60,31,31
31    IF(IREM)40,32,40
32    IFACT(IF)=IDIV
      IF=IF+1
      M=IQUOT
      GO TO 30
40    IDIV=IDIV+2
      GO TO 30
50    INON2=IF
      IF(IREM)60,51,60
51    NTWO=NTWO+NTWO
      GO TO 70
60    IFACT(IF)=M
C
C     SEPARATE  FOUR  CASES--
C        1.  COMPLEX  TRANSFORM  OR  REAL  TRANSFORM  FOR  THE  4TH,  9TH,ETC.
C              DIMENSIONS.
C        2.  REAL  TRANSFORM  FOR  THE  2ND  OR  3RD  DIMENSION.   METHOD--
C            TRANSFORM  HALF  THE  DATA,  SUPPLYING  THE  OTHER  HALF  BY  CON-
C            JUGATE  SYMMETRY.
C        3.  REAL  TRANSFORM  FOR  THE  1ST  DIMENSION,  N  ODD.   METHOD--
C            SET  THE  IMAGINARY  PARTS  TO  ZERO.
C        4.  REAL  TRANSFORM  FOR  THE  1ST  DIMENSION,  N  EVEN.   METHOD--
C            TRANSFORM  A  COMPLEX  ARRAY  OF  LENGTH  N/2  WHOSE  REAL  PARTS
C            ARE  THE  EVEN  NUMBERED  REAL  VALUES  AND  WHOSE  IMAGINARY  PARTS
C            ARE  THE  ODD  NUMBERED  REAL  VALUES.   SEPARATE  AND  SUPPLY
C            THE  SECOND  HALF  BY  CONJUGATE  SYMMETRY.
C
70    ICASE=1
      IFMIN=1
      I1RNG=NP1
      IF(IDIM-4)71,100,100
```

```
71    IF(IFORM)72,72,100
72    ICASE=2
      I1RNG=NP0*(1+NPREV/2)
      IF(IDIM-1)73,73,100
73    ICASE=3
      I1RNG=NP1
      IF(NTWO-NP1)100,100,74
74    ICASE=4
      IFMIN=2
      NTWO=NTWO/2
      N=N/2
      NP2=NP2/2
      NTOT=NTOT/2
      I=1
      DO 80 J=1,NTOT
      DATA(J)=DATA(I)
80    I=I+2
C
C     SHUFFLE DATA BY BIT REVERSAL, SINCE N=2**K.  AS THE SHUFFLING
C     CAN BE DONE BY SIMPLE INTERCHANGE, NO WORKING ARRAY IS NEEDED
C
100   IF(NTWO-NP2)200,110,110
110   NP2HF=NP2/2
      J=1
      DO 150 I2=1,NP2,NP1
      IF(J-I2)120,130,130
120   I1MAX=I2+NP1-2
      DO 125 I1=I2,I1MAX,2
      DO 125 I3=I1,NTOT,NP2
      J3=J+I3-I2
      TEMPR=DATA(I3)
      TEMPI=DATA(I3+1)
      DATA(I3)=DATA(J3)
      DATA(I3+1)=DATA(J3+1)
      DATA(J3)=TEMPR
125   DATA(J3+1)=TEMPI
130   M=NP2HF
140   IF(J-M)150,150,145
145   J=J-M
      M=M/2
      IF(M-NP1)150,140,140
150   J=J+M
      GO TO 300
C
C     SHUFFLE DATA BY DIGIT REVERSAL FOR GENERAL N
C
200   NWORK=2*N
      DO 270 I1=1,NP1,2
      DO 270 I3=I1,NTOT,NP2
      J=I3
      DO 260 I=1,NWORK,2
      IF(ICASE-3)210,220,210
210   WORK(I)=DATA(J)
      WORK(I+1)=DATA(J+1)
      GO TO 230
220   WORK(I)=DATA(J)
      WORK(I+1)=0.
230   IFP2=NP2
      IF=IFMIN
240   IFP1=IFP2/IFACT(IF)
      J=J+IFP1
      IF(J-I3-IFP2)260,250,250
```

```
250   J=J-IFP2
      IFP2=IFP1
      IF=IF+1
      IF(IFP2-NP1)260,260,240
260   CONTINUE
      I2MAX=I3+NP2-NP1
      I=1
      DO 270 I2=I3,I2MAX,NP1
      DATA(I2)=WORK(I)
      DATA(I2+1)=WORK(I+1)
270   I=I+2
C
C     MAIN LOOP FOR FACTORS OF TWO.  PERFORM FOURIER TRANSFORMS OF
C     LENGTH FOUR, WITH ONE OF LENGTH TWO IF NEEDED.  THE TWIDDLE FACTOR
C     W=EXP(ISIGN*2*PI*SQRT(-1)*M/(4*MMAX)).  CHECK FOR W=ISIGN*SQRT(-1)
C     AND REPEAT FOR W=W*(1+ISIGN*SQRT(-1))/SQRT(2).
C
300   IF(NTWO-NP1)600,600,305
305   NP1TW=NP1+NP1
      IPAR=NTWO/NP1
310   IF(IPAR-2)350,330,320
320   IPAR=IPAR/4
      GO TO 310
330   DO 340 I1=1,I1RNG,2
      DO 340 K1=I1,NTOT,NP1TW
      K2=K1+NP1
      TEMPR=DATA(K2)
      TEMPI=DATA(K2+1)
      DATA(K2)=DATA(K1)-TEMPR
      DATA(K2+1)=DATA(K1+1)-TEMPI
      DATA(K1)=DATA(K1)+TEMPR
340   DATA(K1+1)=DATA(K1+1)+TEMPI
350   MMAX=NP1
360   IF(MMAX-NTWO/2)370,600,600
370   LMAX=MAX0(NP1TW,MMAX/2)
      DO 570 L=NP1,LMAX,NP1TW
      M=L
      IF(MMAX-NP1)420,420,380
380   THETA=-TWOPI*FLOAT(L)/FLOAT(4*MMAX)
      IF(ISIGN)400,390,390
390   THETA=-THETA
400   WR=COS(THETA)
      WI=SIN(THETA)
410   W2R=WR*WR-WI*WI
      W2I=2.*WR*WI
      W3R=W2R*WR-W2I*WI
      W3I=W2R*WI+W2I*WR
420   DO 530 I1=1,I1RNG,2
      KMIN=I1+IPAR*M
      IF(MMAX-NP1)430,430,440
430   KMIN=I1
440   KDIF=IPAR*MMAX
450   KSTEP=4*KDIF
      IF(KSTEP-NTWO)460,460,530
460   DO 520 K1=KMIN,NTOT,KSTEP
      K2=K1+KDIF
      K3=K2+KDIF
      K4=K3+KDIF
      IF(MMAX-NP1)470,470,480
470   U1R=DATA(K1)+DATA(K2)
      U1I=DATA(K1+1)+DATA(K2+1)
      U2R=DATA(K3)+DATA(K4)
```

```
      U2I=DATA(K3+1)+DATA(K4+1)
      U3R=DATA(K1)-DATA(K2)
      U3I=DATA(K1+1)-DATA(K2+1)
      IF(ISIGN)471,472,472
471   U4R=DATA(K3+1)-DATA(K4+1)
      U4I=DATA(K4)-DATA(K3)
      GO TO 510
472   U4R=DATA(K4+1)-DATA(K3+1)
      U4I=DATA(K3)-DATA(K4)
      GO TO 510
480   T2R=W2R*DATA(K2)-W2I*DATA(K2+1)
      T2I=W2R*DATA(K2+1)+W2I*DATA(K2)
      T3R=WR*DATA(K3)-WI*DATA(K3+1)
      T3I=WR*DATA(K3+1)+WI*DATA(K3)
      T4R=W3R*DATA(K4)-W3I*DATA(K4+1)
      T4I=W3R*DATA(K4+1)+W3I*DATA(K4)
      U1R=DATA(K1)+T2R
      U1I=DATA(K1+1)+T2I
      U2R=T3R+T4R
      U2I=T3I+T4I
      U3R=DATA(K1)-T2R
      U3I=DATA(K1+1)-T2I
      IF(ISIGN)490,500,500
490   U4R=T3I-T4I
      U4I=T4R-T3R
      GO TO 510
500   U4R=T4I-T3I
      U4I=T3R-T4R
510   DATA(K1)=U1R+U2R
      DATA(K1+1)=U1I+U2I
      DATA(K2)=U3R+U4R
      DATA(K2+1)=U3I+U4I
      DATA(K3)=U1R-U2R
      DATA(K3+1)=U1I-U2I
      DATA(K4)=U3R-U4R
520   DATA(K4+1)=U3I-U4I
      KDIF=KSTEP
      KMIN=4*(KMIN-I1)+I1
      GO TO 450
530   CONTINUE
      M=M+LMAX
      IF(M-MMAX)540,540,570
540   IF(ISIGN)550,560,560
550   TEMPR=WR
      WR=(WR+WI)*RTHLF
      WI=(WI-TEMPR)*RTHLF
      GO TO 410
560   TEMPR=WR
      WR=(WR-WI)*RTHLF
      WI=(TEMPR+WI)*RTHLF
      GO TO 410
570   CONTINUE
      IPAR=3-IPAR
      MMAX=MMAX+MMAX
      GO TO 360
C
C     MAIN LOOP FOR FACTORS NOT EQUAL TO TWO.  APPLY THE TWIDDLE FACTOR
C     W=EXP(ISIGN*2*PI*SQRT(-1)*(J1-1)*(J2-J1)/(IFP1+IFP2)), THEN
C     PERFORM A FOURIER TRANSFORM OF LENGTH IFACT(IF), MAKING USE OF
C     CONJUGATE SYMMETRIES.
C
600   IF(NTWO-NP2)605,700,700
```

```
605   IFP1=NTWO
      IF=INON2
      NP1HF=NP1/2
610   IFP2=IFACT(IF)*IFP1
      J1MIN=NP1+1
      IF(J1MIN-IFP1)615,615,640
615   DO 635 J1=J1MIN,IFP1,NP1
      THETA=-TWOPI*FLOAT(J1-1)/FLOAT(IFP2)
      IF(ISIGN)625,620,620
620   THETA=-THETA
625   WSTPR=COS(THETA)
      WSTPI=SIN(THETA)
      WR=WSTPR
      WI=WSTPI
      J2MIN=J1+IFP1
      J2MAX=J1+IFP2-IFP1
      DO 635 J2=J2MIN,J2MAX,IFP1
      I1MAX=J2+I1RNG-2
      DO 630 I1=J2,I1MAX,2
      DO 630 J3=I1,NTOT,IFP2
      TEMPR=DATA(J3)
      DATA(J3)=DATA(J3)*WR-DATA(J3+1)*WI
630   DATA(J3+1)=TEMPR*WI+DATA(J3+1)*WR
      TEMPR=WR
      WR=WR*WSTPR-WI*WSTPI
635   WI=TEMPR*WSTPI+WI*WSTPR
640   THETA=-TWOPI/FLOAT(IFACT(IF))
      IF(ISIGN)650,645,645
645   THETA=-THETA
650   WSTPR=COS(THETA)
      WSTPI=SIN(THETA)
      J2RNG=IFP1*(1+IFACT(IF)/2)
      DO 695 I1=1,I1RNG,2
      DO 695 I3=I1,NTOT,NP2
      J2MAX=I3+J2RNG-IFP1
      DO 690 J2=I3,J2MAX,IFP1
      J1MAX=J2+IFP1-NP1
      DO 680 J1=J2,J1MAX,NP1
      J3MAX=J1+NP2-IFP2
      DO 680 J3=J1,J3MAX,IFP2
      JMIN=J3-J2+I3
      JMAX=JMIN+IFP2-IFP1
      I=1+(J3-I3)/NP1HF
      IF(J2-I3)655,655,665
655   SUMR=0.
      SUMI=0.
      DO 660 J=JMIN,JMAX,IFP1
 659  SUMR=SUMR+DATA(J)
660   SUMI=SUMI+DATA(J+1)
      WORK(I)=SUMR
      WORK(I+1)=SUMI
      GO TO 680
665   ICONJ=1+(IFP2-2*J2+I3+J3)/NP1HF
      J=JMAX
      SUMR=DATA(J)
      SUMI=DATA(J+1)
      OLDSR=0.
      OLDSI=0.
      J=J-IFP1
670   TEMPR=SUMR
      TEMPI=SUMI
      SUMR=TWOWR*SUMR-OLDSR+DATA(J)
```

```
      SUMI=TWOWR*SUMI-OLDSI+DATA(J+1)
      OLDSR=TEMPR
      OLDSI=TEMPI
      J=J-IFP1
      IF(J-JMIN)675,675,670
 675  TEMPR=WR*SUMR-OLDSR+DATA(J)
      TEMPI=WI*SUMI
      WORK(I)=TEMPR-TEMPI
      WORK(ICONJ)=TEMPR+TEMPI
      TEMPR=WR*SUMI-OLDSI+DATA(J+1)
      TEMPI=WI*SUMR
      WORK(I+1)=TEMPR+TEMPI
      WORK(ICONJ+1)=TEMPR-TEMPI
 680  CONTINUE
      IF(J2-I3)685,685,686
 685  WR=WSTPR
      WI=WSTPI
      GO TO 690
 686  TEMPR=WR
      WR=WR*WSTPR-WI*WSTPI
      WI=TEMPR*WSTPI+WI*WSTPR
 690  TWOWR=WR+WR
      I=1
      I2MAX=I3+NP2-NP1
      DO 695 I2=I3,I2MAX,NP1
      DATA(I2)=WORK(I)
      DATA(I2+1)=WORK(I+1)
 695  I=I+2
      IF=IF+1
      IFP1=IFP2
      IF(IFP1-NP2)610,700,700
C
C     COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N EVEN, BY CON-
C     JUGATE SYMMETRIES.
C
 700  GO TO (900,800,900,701),ICASE
 701  NHALF=N
      N=N+N
      THETA=-TWOPI/FLOAT(N)
      IF(ISIGN)703,702,702
 702  THETA=-THETA
 703  WSTPR=COS(THETA)
      WSTPI=SIN(THETA)
      WR=WSTPR
      WI=WSTPI
      IMIN=3
      JMIN=2*NHALF-1
      GO TO 725
 710  J=JMIN
      DO 720 I=IMIN,NTOT,NP2
      SUMR=(DATA(I)+DATA(J))/2.
      SUMI=(DATA(I+1)+DATA(J+1))/2.
      DIFR=(DATA(I)-DATA(J))/2.
      DIFI=(DATA(I+1)-DATA(J+1))/2.
      TEMPR=WR*SUMI+WI*DIFR
      TEMPI=WI*SUMI-WR*DIFR
      DATA(I)=SUMR+TEMPR
      DATA(I+1)=DIFI+TEMPI
      DATA(J)=SUMR-TEMPR
      DATA(J+1)=-DIFI+TEMPI
 720  J=J+NP2
      IMIN=IMIN+2
```

```
      JMIN=JMIN-2
      TEMPR=WR
      WR=WR*WSTPR-WI*WSTPI
      WI=TEMPR*WSTPI+WI*WSTPR
725   IF(IMIN-JMIN)710,730,740
730   IF(ISIGN)731,740,740
731   DO 735 I=IMIN,NTOT,NP2
735   DATA(I+1)=-DATA(I+1)
740   NP2=NP2+NP2
      NTOT=NTOT+NTOT
      J=NTOT+1
      IMAX=NTOT/2+1
745   IMIN=IMAX-2*NHALF
      I=IMIN
      GO TO 755
750   DATA(J)=DATA(I)
      DATA(J+1)=-DATA(I+1)
755   I=I+2
      J=J-2
      IF(I-IMAX)750,760,760
760   DATA(J)=DATA(IMIN)-DATA(IMIN+1)
      DATA(J+1)=0.
      IF(I-J)770,780,780
765   DATA(J)=DATA(I)
      DATA(J+1)=DATA(I+1)
770   I=I-2
      J=J-2
      IF(I-IMIN)775,775,765
775   DATA(J)=DATA(IMIN)+DATA(IMIN+1)
      DATA(J+1)=0.
      IMAX=IMIN
      GO TO 745
780   DATA(1)=DATA(1)+DATA(2)
      DATA(2)=0.
      GO TO 900
C
C     COMPLETE A REAL TRANSFORM FOR THE 2ND OR 3RD DIMENSION BY
C     CONJUGATE SYMMETRIES.
C
800   IF(I1RNG-NP1)805,900,900
805   DO 860 I3=1,NTOT,NP2
      I2MAX=I3+NP2-NP1
      DO 860 I2=I3,I2MAX,NP1
      IMIN=I2+I1RNG
      IMAX=I2+NP1-2
      JMAX=2*I3+NP1-IMIN
      IF(I2-I3)820,820,810
810   JMAX=JMAX+NP2
820   IF(IDIM-2)850,850,830
830   J=JMAX+NP0
      DO 840 I=IMIN,IMAX,2
      DATA(I)=DATA(J)
      DATA(I+1)=-DATA(J+1)
840   J=J-2
850   J=JMAX
      DO 860 I=IMIN,IMAX,NP0
      DATA(I)=DATA(J)
      DATA(I+1)=-DATA(J+1)
860   J=J-NP0
C
C     END OF LOOP ON EACH DIMENSION
C
```

```
900    NPO=NP1
       NP1=NP2
910    NPREV=N
920    RETURN
       END
       SUBROUTINE AIRY(A,B,X)
C
C...   ROUTINE TO CALCULATE AIRY FUNCTION AT X
C
       DIMENSION ATAB(67),BTAB(67),ADTAB(67),BDTAB(67),AA(11)
       DATA PI/3.1415926535/,  IW/0/
C
C...   SET UP TAYLOR TABLES
C
       IF(IW.NE.0) GOTO 30
       IW=1
       ATAB(67)=2.1565999525E-6
       ADTAB(67)=-5.619319442E-6
       BTAB(34)=0.6149266274
       BDTAB(34)=0.4482883574
       XTAB=0.
       DO 10 I=34,66
       CALL TAYLOR(B,BD,XTAB,0.1,BTAB(I),BDTAB(I))
       CALL TAYLOR(BTAB(I+1),BDTAB(I+1),XTAB+0.1,0.1,B,BD)
       CALL TAYLOR(B,BD,-XTAB,-0.1,BTAB(68-I),BDTAB(68-I))
       CALL TAYLOR(BTAB(67-I),BDTAB(67-I),-XTAB-0.1,-0.1,B,BD)
       XTAB=XTAB+0.2
   10  CONTINUE
       DO 20 I=2,67
       CALL TAYLOR(A,AD,XTAB,-0.1,ATAB(69-I),ADTAB(69-I))
       CALL TAYLOR(ATAB(68-I),ADTAB(68-I),XTAB-0.1,-0.1,A,AD)
       XTAB=XTAB-0.2
   20  CONTINUE
C
C...   TAYLOR SERIES EXPANSION
C
   30  CONTINUE
       IF(ABS(X).GT.6.6) GOTO 40
       J=5.*X
       XTAB=FLOAT(J)/5.
       H=X-XTAB
       CALL TAYLOR(A,AD,XTAB,H,ATAB(34+J),ADTAB(34+J))
       CALL TAYLOR(B,BD,XTAB,H,BTAB(34+J),BDTAB(34+J))
       GOTO 70
C
C...   ASYMPTOTIC SOLUTION
C
   40  CONTINUE
       RTMDX=SQRT(ABS(X))
       XI=RTMDX**3/1.5
       FACTOR=1./(12.*XI)
       AA(1)=1./SQRT(PI*RTMDX)
       R=6.
       DO 50 I=1,10
       AA(I+1)=(R-1.)*(R-5.)*FACTOR*AA(I)/R
       R=R+6.
   50  CONTINUE
       IF(X.LT.0.) GOTO  60
       P=AA(1)+AA(3)+AA(5)+AA(7)+AA(9)+AA(11)
       Q=AA(2)+AA(4)+AA(6)+AA(8)+AA(10)
       SCALE=EXP(XI)
       A=(P-Q)/(2.*SCALE)
```

```
      B=(P+Q)*SCALE
      GOTO 70
C
   60 CONTINUE
      P=AA(1)-AA(3)+AA(5)-AA(7)+AA(9)-AA(11)
      Q=AA(2)-AA(4)+AA(6)-AA(8)+AA(10)
      S=SIN(XI+PI/4.)
      C=COS(XI+PI/4.)
      A=P*S-Q*C
      B=P*C+Q*S
C
   70 CONTINUE
      RETURN
      END
      SUBROUTINE TAYLOR(Y1,D1,X,H,Y,D)
C
C...  SUBROUTINE TO CALCULATE Y(X+H) FROM Y(X) BY SERIES EXPANSION
C
      DIMENSION TOR(11)
C
      IF(H.NE.0.) GOTO 10
      Y1=Y
      D1=D
      GOTO 30
C
   10 CONTINUE
      TOR(1)=Y
      TOR(2)=H*D
      SQUARE=H*H
      TOR(3)=.5*SQUARE*X*TOR
      Y1=TOR+TOR(2)+TOR(3)
      D1=TOR(2)+2.*TOR(3)
      DO 20 N=4,11
      TOR(N)=SQUARE*(X*TOR(N-2)+H*TOR(N-3))/((N-1)*(N-2))
      Y1=Y1+TOR(N)
      D1=D1+(N-1)*TOR(N)
   20 CONTINUE
      D1=D1/H
C
   30 CONTINUE
      RETURN
      END
```

1

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

14

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
TRW 20086-603-RU-00

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
6
SUBMARINE EFFECTS ENGINEERING CODE.
4. Operating Instructions.

5. TYPE OF REPORT & PERIOD COVERED
9
Technical Report.

6. PERFORMING ORG. REPORT NUMBER
20086-6013-RU-00

7. AUTHOR(s)
10
David J. Henryson

8. CONTRACT OR GRANT NUMBER(s)
15
N00014-72-C-0074

9. PERFORMING ORGANIZATION NAME AND ADDRESS
TRW Systems Group
One Space Park
Redondo Beach, CAlifornia 90278

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
ARPA Order 1910, Amend #13
4/9/74 Program Code #5E30

11. CONTROLLING OFFICE NAME AND ADDRESS
Defense Advanced Research Projects Agency
1400 Wilson Boulevard
Arlington, Virginia 22209

12. REPORT DATE
11
31 July 1975
12

13. NUMBER OF PAGES
179
183 p.
1

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
Department of the Navy
Office of Naval Research
Arlington, Virginia 22217

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Internal waves -- Stratified Flows

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Volumes 1-3 of this series present an analytical description of the invisci disturbance generated by a submerged moving body in a generally stratified medium. This volume describes the operation of a computer code implementing this analysis, including test cases and code listings.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

354 595

DISTRIBUTION LIST

Dr. J. D'Albora
Naval Underwater Systems Center
Newport, Rhode Island 02840

Johns Hopkins University
Applied Physics Laboratory
8621 Georgia Avenue
Silver Spring, Maryland 20910
Attn: L. Cronvich
R. Gasparovic
H. Gilreath
J. Lowndes
W. May
A. Stone

University of California, SD
Marine Physical Laboratory
Scripps Institution of Oceanography
San Diego, California 92152
Attn: Dr. F. Spiess

Dr. Martin H. Bloom
Polytechnic Inst. of Brooklyn
Route 110
Farmingdale, New York 11735

General Research Corporation
1501 Wilson Blvd., Suite 700
Arlington, Va. 22209
Attn: Mr. P. Donahoe

Dr. Marvin King
Riverside Research Institute
80 West End Avenue
New York, New York 10023

Mr. V. Lujetic
Tetra Tech, Inc.
1911 N. Ft. Myer Drive
Arlington, Virginia 22209

University of California
Lawrence Livermore Laboratory
PO Box 808
Livermore, California 94550
Attn: Dr. H. P. Smith
Dr. Michael Wirth

Dr. W. S. Lewellen
Aeronautical Research Associates
of Princeton, Inc.
50 Washington Road
Princeton, New Jersey 08540

R&D Associates
PO Box 3580
Santa Monica, California 90403
Attn: Dr. D Holliday

Dr. K. V. Saunders
Pacific-Sierra Research Corp.
PO Box 578
Malibu, California 90625

Dr. Thomas Taylor
The Aerospace Corporation
PO Box 92957
Los Angeles, California 90009

Mr. L. W. Griswold
Naval Ship Research &
Development Center
Annapolis Laboratory
Annapolis, Maryland 21402

Dr. Frank Lane
KLD Associates, Inc.
7 High Street, Suite 204
Huntington, New York 11743

Flow Research, Inc.
1819 S. Central Ave., Suite 72
Kent, Washington 98031
Attn: Dr. D.R.S. Ko
Dr. M.Y.H. Pao

Hydronautics, Inc.
Pindell School Road
Howard County
Laurel, Maryland 20810
Attn: Dr. J. Wu
Dr. T. R. Sundaram

Xonics, Inc.
6837 Hayvenhurst Ave.
Van Nuys, California 91406
Attn: Dr. M. Balser

Dr. J. Alex Thomson
Physical Dynamics, Inc.
PO Box 1069
Berkeley, California 94701

Dr. Roberto Vaglio-Laurin
Advanced Technology Labs., Inc.
400 Jericho Turnpike
Jericho, New York 11753

Mr. R. Rehm
CALSPAN Corporation
PO Box 235
Buffalo, New York 14221

Stanford Research Institute
333 Ravenswood Avenue
Menlo Park, California 94025
Attn: H. Guthardt
K. Krishnan

Institute for Defense Analyses
400 Army Navy Drive
Arlington, Virginia 22202
Attn: Dr. P. A. Selwyn
Dr. Joel Bengston
Mr. J. C. Nolen

Naval Research Laboratory
4555 Overlook Avenue
Washington, D. C. 20390
Attn: F. MacDonald
J. O. Elliot
K. G. Williams
S. Piacsek
A. H. Schooley

Office of Naval Research
800 N. Quincy Street
Arlington, Virginia 22217
Attn: M. Cooper (438)
R. Cooper (438)
J. Witting (481)
Cdr J. Ballou (466)
Dr. S. Reed (102T)

Naval Undersea Center
San Diego, California 92132
Attn: O. Lee

Defense Advanced Research Projects Agency
1400 Wilson Blvd.
Arlington, Virginia 22209
Attn: K. Kresa, TTO
S. Ruby, MATS

ODDR&E
The Pentagon
Washington, D. C. 20301
Attn: D. R. Heebner, 3E1040
N. F. Wikner, 3E1087
G. Cann, 3D1048

Director
Central Intelligence Agency
Main Station
Washington, D. C. 20305
Attn: H. Farmer

Cdr D. Walsh
OASN/R&D
The Pentagon, 4E 471
Washington, D. C. 20350

Mr. D. A. Rogers
SP 2018/NSP
Department of the Navy
Washington, D. C. 20390

Captain D. Keech
Deputy Director of Navy Labs
Department of the Navy
Washington, D. C. 20360

Naval Scientific & Technical Intelligence Center
4301 Suitland Road
Washington, D. C. 20390
Attn: Capt J. P. Prisley

Mr. I. H. Gatzke
Naval Air Systems Command
Department of the Navy
Washington, D. C. 20360

Defense Documentation Center (2 cys)
Cameron Station
Alexandria, Virginia 22314

Dr. John Dugan
Code 8340
Naval Research Laboratory
Washington, D. C. 20375