

FC  
USGS  
OFR  
80-1159

U.S. Department of the Interior  
Geological Survey

Mail Stop 964  
Box 25046, Federal Center  
Denver, Colorado 80225

Program MQLVTHXYZ:  
Computer inversion of three-component, time-domain,  
magnetic-field sounding data  
generated using an electric wire source

by

James P. Kaushikava

**UNIVERSITY OF UTAH  
RESEARCH INSTITUTE  
EARTH SCIENCE LAB.**

OPEN-FILE REPORT 80-1159

1980

CONTENTS.

DISCLAIMER	3
INTRODUCTION	4
PARAMETERS AND DATA REQUIRED	6
PROGRAM FILES	6
DETAIL PARAMETER AND DATA DEFINITIONS	7
\$params parameters	7
\$init parameters	14
DATA MATRIX NOTES	17
EXAMPLES OF INPUT PARAMETERS AND DATA ORDERING	18
SPECIAL OBJECT FORMAT PHRASES	18
MULTICS OPERATING INSTRUCTIONS	19
ERROR MESSAGES	19
REFERENCES	20
Appendix 1.-- Source listings	22
Source availability	23
Appendix 2.-- Conversion to other systems	104
Appendix 3.-- Test problem input/output listings	106

DISCLAIMER.

This program was written in Fortran IV for a Honeywell Multics 68/80 system\*. Although program tests have been made, no guarantee (expressed or implied) is made by the author regarding accuracy or proper functioning of this program on all computer systems.

---

\* Brand or manufacturers' names used in this report are for descriptive purposes only and do not constitute endorsement by the U.S. Geological Survey.

U.S. Department of the Interior  
Geological Survey

Mail Stop 964  
Box 25046, Federal Center  
Denver, Colorado 80225

Program MQLVTHXYZ:  
Computer inversion of three-component, time-domain,  
magnetic-field sounding data  
generated using an electric wire source

by

James P. Kauahikaua

OPEN-FILE REPORT 80-1159

1980

CONTENTS.

DISCLAIMER	3
INTRODUCTION	4
PARAMETERS AND DATA REQUIRED	6
PROGRAM FILES	6
DETAIL PARAMETER AND DATA DEFINITIONS	7
\$params parameters	7
\$init parameters	14
DATA MATRIX NOTES	17
EXAMPLES OF INPUT PARAMETERS AND DATA ORDERING	18
SPECIAL OBJECT FORMAT PHRASES	18
MULTICS OPERATING INSTRUCTIONS	19
ERROR MESSAGES	19
REFERENCES	20
Appendix 1.-- Source listings	22
Source availability	23
Appendix 2.-- Conversion to other systems	104
Appendix 3.-- Test problem input/output listings	106

DISCLAIMER.

This program was written in Fortran IV for a Honeywell Multics 68/80 system\*. Although program tests have been made, no guarantee (expressed or implied) is made by the author regarding accuracy or proper functioning of this program on all computer systems.

---

\* Brand or manufacturers' names used in this report are for descriptive purposes only and do not constitute endorsement by the U.S. Geological Survey.

(1) The terms "impulse" and "step" response refer to the manner in which the wire source is energized and the resulting magnetic fields are measured. If the wire is driven with a spike current (instantly on and off), then the EM fields observed are impulse responses. On the other hand, if the wire is driven with a step current (on at zero time and kept on), then the EM fields observed are step responses. These two are the most commonly used source current waveforms. If the magnetic field TEM data are measured with a wire loop, then the data are actually the time-derivatives of the actual fields. In this case, an impulse response would be measured for a source wire energized with a current step because the time-derivative of a step response is an impulse response. Modification of the program to accommodate other current source waveforms can be done using the user-defined function FLI which is controlled by shift parameters IFLX, IFLY, and IFLZ.

- (9) Amplitude shift parameters may be used to automatically or filtering in the receiver.
- (8) A nonincreasing frequency function may be convolved with the model to simulate an arbitrary source function (Kauahikaua and Anderson, 1977). (1)
- (7) The impulse or step response may be modeled.
- (6) An infinitesimal- or finite-length wire source may be modeled.
- (5) Object-time format control for reading the observed data matrix.
- (4) Holding certain parameters fixed (constrained).
- (3) Weighted observations.
- (2) Scaling parameter and observation spaces to constrain the solution space and to reduce round-off effects.
- (1) Simultaneous (or joint) inversion of three components of TEM data and/or Schlumberger apparent resistivity soundings data.

The following program options are currently available; computer systems where the IMSL library was not available. soundings data. See appendix 2 notes for conversion to other squares subprogram (ZXSQ) is used for inversion of the TEM Levenburg-Marquardt (Brown-Dennis, 1972) nonlinear least mathematical and Statistical Library, 1977) derivative-free (Anderson, 1977; Kauahikaua, 1978). An IMSL (International (Anderson, 1974; 1975; 1977; 1979); Kauahikaua and inversion of time-domain electromagnetic (TEM) soundings data when a grounded electric wire source is employed. Details of the actual computations are given in other papers.

INTRODUCTION.

By James F. Kauahikaua

- scale the models to match the data magnitudes.
- (10) Orientation parameters may be used to automatically rotate the models for discrepancies in orientation of the measurement axes with respect to the wire.

Much of the form and wording of this report has been adopted from a series of recent USGS Open-File Reports by Anderson (1979b, for example) to provide as much timely program information as possible in a fairly standardized format. These reports have no mathematical formulation section, but the interested reader may consult the cited references for more information on the mathematics.



### PARAMETERS AND DATA REQUIRED.

Parameters that are required by program MQLVTHXYZ are read using Fortran namelist read statements with specific names: \$parms and \$init. Default values are used whenever a corresponding parameter is omitted in a namelist. The input data matrix is read from an optional alternate file (unless overridden) using a Fortran object-time format. Preceding the \$parms statement is a required 80 (or less) character title.

The general input order read by program MQLVTHXYZ is:

1. Title line (always required, max. 80 characters).
2. \$parms --non-default parameters--\$  
(note \$parms may begin in col. 1 on Multics).
3. (Object-time format) statement defining the given format of the input data matrix. The object format begins with "(" placed in col. 1 and ends with ")" before col. 73.
4. Optionally, the data matrix read under the object format may be inserted here if the alternate data file is not used (see parameter ialt below).
5. \$init --non-default parameters--\$
6. Optionally, subsequent runs using the same data matrix but with changed \$parms and \$init parameters may be made by repeating steps 1,2,3, and 5 (provided parameters istop=0 and ialt is not 5).

The above general input order is required whether the job is being run in time-sharing or batch modes (see job operating instructions below).

### PROGRAM FILES.

file05 title, input parameters \$parms, object format (for reading data matrix on unit ialt=10--default), and \$init parameters.

file06 output on-line printer file (see file16 for more detail output).

file10 default input data matrix file read under the object format given in file05. Parameter ialt=10 (default) may be changed to any file number other than 06,13, or 16. Note ialt=05 will mean the data matrix is included immediately after the object-time format on file05.

file13 output scratch disk file used as required during execution of MQLVTHXYZ.

file16 output master print-type disk file--contains complete printable output.

DETAIL PARAMETER AND DATA DEFINITIONS.

\$parms parameters (with defaults and cross-references):

[names below prefixed with a "\*" are not used by MQLVTHXYZ, but are included for conversion compatibility if the CALL IMSLMQ is replaced by CALL MARQRT (see appendix 2 paragraph 6 on this type of conversion)]

n= Number of observed data points  $y(i), i=1, \dots, n$ , where  $n \leq 200$ .

k= Total number of parameters ( $1 \leq k \leq 20, k \leq n$ ). The value of k must be equal to  $2*mm$  if \$init parameter iob=1,2,3, equal to  $2*mm+2$  if iob=4, and equal to  $2*mm+5$  if iob=5, where \$init parameter  $mm > 0$  is the number of layers in the model and iob is the observation type of the data.  
(cref: \$init parameter mm, iob and \$parms n, b).

is= Number of omitted parameters; i.e., number of parameters held fixed or constrained via array ib() to initial input values given in array b(). Default is=0 with the restrictions that  $is < k$  and  $n \geq k - is$ .  
(cref: \$parms k, n, ib(), and b).

m= Number of independent variables ( $m \leq 2$ ) given in the data matrix  $(y(i), x(i, j), j=1, m), i=1, n$ . The value of m must be given as follows:  
= 1 when \$init parameter iob  $\leq 3$  (defines specific observation type in  $y(i)$ );  
= 2 when \$init parameter iob=4,5 (defines mixed observation types in  $y(i)$  via  $x(i, 2)$ ).  
(cref: \$parms iwt, \$init iob, and DATA MATRIX NOTES below for all definitions of  $x(i, m)$  used).

ialt= Input data matrix alternate logical unit number (default 10) for reading the data under the object-time format specified in file05. The value of ialt can be any value the operating system supports, but cannot be equal to 6, 13, or 16. If ialt=5 is used, then the data matrix  $((y(i), x(i, j), j=1, m), i=1, n)$  will immediately follow the object format on file05.  
(cref: \$parms n, m, \$init iob).

istop= 0 to continue processing after completion of the current problem (i.e., a total restart) with the same data matrix as last used, but by using a revised title, \$parms, object-time format, and \$init parameters. Note that istop=0 can only be

used whenever ialt is not 5 (since file ialt is  
rewound and read again). Also, all \$parms and  
\$init parameters previously used will be assumed,  
with the exception of array b(j) which must always  
be given.

= 1 (default) to stop the run after completion of the  
current problem.  
(cref: \$parms b,ialt).

iwt= 0 (default) for unweighted observations; i.e., all n  
observations  $y(i), i=1, \dots, n$  will be weighted units  
(with assumed standard deviations equal to 1.0).

= 1 for weighted observations given by the formula  
 $wt(i)=1.0/x(i,m+1)**2$ , where  $x(i,m+1)$  is the  
standard deviation augmented to the data matrix  
for the given  $m \leq 2$ . Note: to avoid division by  
zero,  $wt(i)=1.0$  is stored automatically if  $iwt=0$   
or when  $iwt=1$  and  $x(i,m+1)=0.0$ .

(cref: \$parms n,m, \$init iob, and DATA MATRIX  
NOTES).

\* ider= 0 (default) to use analytic derivatives, which calls  
both forward problem (fcode) and analytic  
derivative (rcode) subroutines.

= 1 to use estimated derivatives, which calls only  
subroutine fcode. [if converting to subprogram  
MARQRT (as in appendix 2), then ider=1 must always  
be used, since rcode is a dummy routine].  
(cref: \$parms del).

iert= 0 (default) for standard abbreviated printout format  
for each iteration. Note scaled values of  
parameters b(j) and phi (sum of squares) will be  
given via parameter scaler.

= 1 for detail printout format for each iteration,  
which includes the parameter changes from the  
Marquardt algorithm. [note iert=1 behaves like  
iert=-2, unless converting to subprogram MARQRT].

= -1 (recommended if scaler>0 used) for abbreviated  
printout format for each iteration with printed  
unscaled values of b(j) but scaled values of phi.

= -2 same as iert=-1 but also prints on file06  
n-observational lines containing: observed value  
(obs=y(i)), calculated value (cal), residual  
(res), and  $x(i,1)$ . Note file16 will always  
contain the complete obs-cal-res and  $x(i,m)$  data  
printout. Option iert=-2 may be useful for  
time-sharing runs to examine on-line the final  
solution and residuals.

(cref: \$parms iout,sp and DATA MATRIX NOTES).

\* niter= Maximum number of iterations allowed before accepting the results as "forced off" (default niter=10). Four different types of convergence tests are possible one of which is termed "forced off", which will occur whenever niter has been reached and one of the other convergence criteria has not been achieved. Using a small niter may be useful to monitor the progress for a large problem, and as an aid for achieving a convenient restarting procedure with the last b-vector as a new initial estimate.

(cref: #parms b and Marquardt (1963) for convergence tests used).

\* inon= 1 (default) to omit nonlinear confidence region calculations.

= 0 to compute nonlinear confidence regions after the last iteration. This option calls subroutine #code many times, and is not recommended for general use with program MQLVTHXYZ unless one is interested in a detailed nonlinear statistical analysis of the final solution. (see IBM Share Program No. 1428 for more details on this option).

\* ffr=

Variance F-ratio statistic (default 4.0) used to compute linear support-plane confidence limits and nonlinear (if inon=0) confidence limits after convergence or niter iterations. The default value is adequate for most applications.

\* t=

Student's t-statistic (default 2.0) used to compute one-parameter linear confidence limits after convergence or niter iterations. The default value is adequate for most applications.

\* e=

Convergence criterion test parameter (default 0.5e-4). For example, for 2-figure accuracy, use e=.01; for 3-figure accuracy, use e=.001, etc. [For MQLVTHXYZ, e is equivalent to #parms ee (see below)].

\* tau=

Convergence criterion test parameter (default 1e-3).

\* xi=

Initial Marquardt's lambda factor (default .01) to be added to the diagonal of the Jacobian transpose times Jacobian matrix. For some very ill-conditioned problems, or for poor initial parameter estimates, a larger xi (e.g., 1.0) may

prove to be advantageous.  
(cref: Marquardt, 1963 and IBM Share Program No. 1428).

\* modiam= 1 (default) to use a modified Marquardt lambda method at each iteration as described in Tabata and Ito (1973).  
= 0 to use the original Marquardt (1963) lambda method at each iteration.

\* samcr= Marquardt's critical angle between the gradient and adjustment vectors (default 45.0 degrees). The value of samcr should not be set greater than 90 degrees. The default value is usually adequate for most applications.  
(cref: Marquardt, 1963).

\* del= Factor used in finite-difference equations (default 1e-5). Note del is used only when ider=1 for estimated partial derivative calculations.  
(cref: \$parms ider).

\* zeta= Singularity criterion for matrix inversion (default 1e-31), which may be selected greater than or equal to the machine smallest exponent range.

\* iout= Printout file06 and file6 control.  
= 1 (default) for printable output to both file06 and file6.  
= 0 for print output only on file06.  
Note: file6 output was be useful for deferred output when running the job from a time-sharing terminal; also, file6 may be used as an input file for other processing programs (e.g., plot routines). In this version of the program, output to file06 has been purposely reduced to take less time to printout on a time-sharing terminal; however, for iout=1 (default), a complete printable output is always given on file6.  
(cref: \$parms iout).

scaler (equivalent names) is a parameter scaling option.

= 0 (default) to ignore parameter scaling (i.e., to use unscaler parameters).

= 1 (recommended for program MQLVTHXYZ) to scale parameters b(j) using ln(b(j)), provided the initial b(j)>0 for all j=1,2,...,k. Note scaler=1 will automatically constrain the final solution space such that b(j)>0 for all j in (1:k).  
= 2 to scale parameters b(j) using exp(ln(b(j))).

This option allows for log-type parameter scaling whenever  $b(J)$  is positive or negative for any  $J$  in  $(1,k)$ . However, for program MQLVTHXYZ (\$init parameter  $iob \leq 4$ ), the initial parameters  $b(J) > 0$  must be given; hence  $sp=2$  should not be used ( $sp=2$  is defined here for the case where \$init  $iob=5$  or for possible use in other applications). (cref: \$parms  $b,k$ ).

\*  $sy=$  scale $y$  (equivalent names) is an observation scaling option.  
= 0 (default) to ignore observation scaling (i.e., to use unscaled observations  $y(i)$ ),  
= 1 to scale observations  $y(i)$  using  $\ln(y(i))$ , provided  $y(i) > 0$  for all  $i=1,2,\dots,n$ .  
= 2 to scale observations  $y(i)$  using  $\operatorname{arcsinh}(y(i))$ . This option allows for log-type observation scaling whenever  $y(i)$  is positive, negative, or zero for any  $i$  in  $(1,n)$ . (cref: \$init  $iob$ , \$parms  $n$  and DATA MATRIX NOTES)

$b()$ = Array of initial guesses for all  $k$ -parameters. These values must be supplied greater than zero for program MQLVTHXYZ (i.e., positive conductivities and thicknesses), unless \$init parameter  $iob=5$ . The default values are set to  $b(J)=0$  for all  $J=1$  to  $k$ , and would result in an error condition if any  $b(J)$  was not supplied greater than zero.

For  $iob \leq 3$ , the parameter order must be given as:

$b(1)$  is the amplitude shift parameter,

$b(2), b(3), \dots, b(mm+1)$  are the  $mm$  layer conductivities (in mhos/meter), and

$b(mm+2), b(mm+3), \dots, b(2*mm)$  are the  $mm-1$  layer thicknesses (in meters).

For  $iob=4$ , the parameter order must be given as:

$b(1), b(2), b(3)$  are the  $x$ -,  $y$ -, and  $z$ -component amplitude shift parameters, respectively (any unused ones should be constrained via \$parms parameters  $ip$  and  $ib()$ ),

$b(4), b(5), \dots, b(mm+3)$  are the  $mm$  layer conductivities (in mhos/meter), and

$b(mm+4), b(mm+5), \dots, b(2*mm+2)$  are the  $mm-1$  layer thicknesses, in meters.

For job=5, the parameter order must be given as:

b(1),b(2),b(3) are the x-, y- and z-component  
 amplitude shift parameters, respectively (any  
 unused ones should be constrained via parms  
 parameters ie and id( ));

b(4),b(5),b(6) are the orientation parameters,  
 angles theta, phi, and omega, where

$z_{new} = z_{prime} \cos(\theta) + \sin(\theta) * (x_{prime} \cos(\phi) + y_{prime} \sin(\phi))$   
 $x_{new} = x_{prime} \cos(\omega) + y_{prime} \sin(\omega)$   
 $y_{new} = y_{prime} \cos(\omega) - x_{prime} \sin(\omega)$   
 where  $x_{prime}$  and  $y_{prime}$  are defined by

$x_{prime} = x_{prime} \cos(\theta) + \sin(\theta) * (x_{prime} \sin(\phi) - y_{prime} \cos(\phi))$   
 $y_{prime} = y_{prime} \cos(\theta) - \sin(\theta) * (x_{prime} \sin(\phi) + y_{prime} \cos(\phi))$

b(7),b(8),...b(2kmm+5) are the mm-1 layer  
 thicknesses in meters.  
 (cref: parms k,ie,id and #int m,myid).

Array of ie-indices (in any order) corresponding  
 to any b() parameter to hold fixed to its input  
 value, e.g., ie=2,ib(1)=3,ib(2)=5 will hold fixed  
 b(3), b(5) in the least squares. If ie=0  
 (default) leave out array id in the namelist.  
 (cref: parms ie,ib).

The following parms are parameters used only by IML  
 subprogram ZXSSQ, and cannot be used if converting to  
 subprogram MARKIT as described in appendix 2].

lopt= 1 (default) implies strict descent of the sum of  
 squares is desired in the derivative-free  
 Marquardt algorithm (ZXSSQ), with default values  
 used in the input array parm().  
 = 0 implies strict descent is not necessary (i.e., the  
 "best" or optimum Marquardt parameter used may not  
 yield a strict decreasing sum of squares at each  
 iteration).  
 = 2 implies strict descent is desired with user  
 parameter choices as given (or assumed) in input  
 array parm().  
 (cref: parms parm()).

parm() = Array of length 4 required only when lopt=2. The  
 default is parm(1)=.01,parm(2)=.120,parm(3)=.1, where each

element is defined by the corresponding index as follows:

i=1, the initial value of the Marquardt parameter used to scale the diagonal of the approximate Hessian matrix,  $x_{ij}$ , by the factor  $(1.0 + \text{parm}(1))$ . A small value gives a Newton step, while a large value gives a steepest descent step. (default  $\text{parm}(1) = .01$ ).

i=2, the scaling factor used to modify the Marquardt parameter, which is decreased by  $\text{parm}(2)$  after an immediately successful descent direction, and increased by the square of  $\text{parm}(2)$  if not. (default  $\text{parm}(2) = 2$  where  $\text{parm}(2) > 1$  must be used).

i=3, an upper bound for increasing the Marquardt parameter. The search for a descent point is abandoned if  $\text{parm}(3)$  is exceeded.  $\text{parm}(3) > 100$  is recommended. (default  $\text{parm}(3) = 120$ ).

i=4, value for indicating when central rather than forward differencing is to be used for calculating the Jacobian (partial derivatives). The switch is made when the norm of the gradient of the sum of squares function becomes smaller than  $\text{parm}(4)$ . Central differencing is good in the vicinity of the solution, so  $\text{parm}(4)$  should be small. (default  $\text{parm}(4) = .1$ ).

( $\text{cref}; \text{parms } \text{loft}$ ).

**nsis=** The first convergence criterion. Convergence is satisfied if on 2 successive iterations, the parameter estimates agree, component by component, to  $\text{nsis}$  digits. (default  $\text{nsis} = 5$  using  $\text{nsis} > 5$  may not converge since single precision is used).

**eps=** The second convergence criterion. Convergence is satisfied if on 2 successive iterations the residual sum of squares estimates have relative differences  $\leq \text{eps}$ . (default  $\text{eps} = 0.0$ ).

( $\text{cref}; \text{parms } \text{ep}$ , which is equivalent to  $\text{eps}$ ).

**delta=** The third convergence criterion. Convergence is stated if the Euclidean norm of the approximate gradient is  $\leq \text{delta}$ . (default  $\text{delta} = 0.0$ ).

Note: The Marquardt iteration is terminated, and convergence is considered achieved, if any one of the three convergence conditions ( $\text{nsis}, \text{eps}$ , or  $\text{delta}$ ) is satisfied.

**maxfn=** The maximum number of function evaluations (i.e., calls to subroutine FUNC in ZXSSQ) allowed. The actual number of calls to FUNC may exceed  $\text{maxfn}$ .



```

method= selects the numerical technique for computing the
step response (only used if #init itstep=1);
= 1 computed as the integral from 0 to t of the
impulse response (see Kanahikawa and Anderson);

= 0 (default) for the impulse response,
= 1 for the step response.

istep = selects which of the source current waveforms is
to be used.
= 0 (default) for the impulse response,
= 1 for the step response.

mm= number of layers in the model (1<=mm<=10; default
mm=1).
Note: make sure #parms k=2*mm for job<=3, k=2*mm+2
for job=4, and k=2*mm+5 for job=5.
(cref: #parms k*d()* #init job).

```

```

= 5 behaves like job=4 except that the orientation
parameters may be used. It is advisable to use
the job=5 option only after the best solution has
been obtained with the appropriate job<=4. Then
when the job=5 run is made, #parms sp should be
set to zero to allow the angles to be positive or
negative.
(note: for job=4+5, m=2 must also be given in
#parms).
(cref: #parms m*d()* #init mm; and DATA MATRIX
NOTES).

= 4 defines mixed observation-type soundings where the
1-th observation type is given by x(1,2)=1.0 for
x-component, =2.0 for y-component, =3.0 for
z-component, and =4.0 for Schlumberger apparent
resistivities.
= 5 behaves like job=4 except that the orientation
parameters may be used. It is advisable to use
the job=5 option only after the best solution has
been obtained with the appropriate job<=4. Then
when the job=5 run is made, #parms sp should be
set to zero to allow the angles to be positive or
negative.
(note: for job=4+5, m=2 must also be given in
#parms).
(cref: #parms m*d()* #init mm; and DATA MATRIX
NOTES).

```

```

= 1 defines v(1) as x-component magnetic field TDEM
data; in amps/m (parallel to wire source);
= 2 defines v(1) as y-component magnetic field TDEM
data; in amps/m (perpendicular to the wire
source);
= 3 defines v(1) as z-component magnetic field TDEM
data; in amps/m (vertical);
(note: for job<=3, m=1 must also be given in
#parms).
= 4 defines mixed observation-type soundings where the
1-th observation type is given by x(1,2)=1.0 for
x-component, =2.0 for y-component, =3.0 for
z-component, and =4.0 for Schlumberger apparent
resistivities.
= 5 behaves like job=4 except that the orientation
parameters may be used. It is advisable to use
the job=5 option only after the best solution has
been obtained with the appropriate job<=4. Then
when the job=5 run is made, #parms sp should be
set to zero to allow the angles to be positive or
negative.
(note: for job=4+5, m=2 must also be given in
#parms).
(cref: #parms m*d()* #init mm; and DATA MATRIX
NOTES).

```

```

= 1 defines v(1) as x-component magnetic field TDEM
data; in amps/m (parallel to wire source);
= 2 defines v(1) as y-component magnetic field TDEM
data; in amps/m (perpendicular to the wire
source);
= 3 defines v(1) as z-component magnetic field TDEM
data; in amps/m (vertical);
(note: for job<=3, m=1 must also be given in
#parms).
= 4 defines mixed observation-type soundings where the
1-th observation type is given by x(1,2)=1.0 for
x-component, =2.0 for y-component, =3.0 for
z-component, and =4.0 for Schlumberger apparent
resistivities.
= 5 behaves like job=4 except that the orientation
parameters may be used. It is advisable to use
the job=5 option only after the best solution has
been obtained with the appropriate job<=4. Then
when the job=5 run is made, #parms sp should be
set to zero to allow the angles to be positive or
negative.
(note: for job=4+5, m=2 must also be given in
#parms).
(cref: #parms m*d()* #init mm; and DATA MATRIX
NOTES).

```

1977).

= 2 (default) computed as the transform of the field frequency response divided by the frequency.

**fltkx=** filter option parameter for the x-component computations (default fltkx=0). If fltkx is different from zero, the magnetic field IDFM response will be computed as the transform of the x-component frequency response multiplied by a user-defined complex function FLT(IFLTIX,F) where F is the frequency in Hertz. For multicomponent inversion (fltk idb=4;5), fltk may be used to differentiate between a call to FLT(IFLTIX,F) and either FLT(IFLTY,F) or FLT(IFLTZ,F) if desired, simply by including the appropriate FORTRAN statements in the user-written COMPLEX FUNCTION FLT. See example routine FLT listed in appendix 1.

**flty=** similar to fltk except for the y-component (default flty=0).

**fltz=** similar to fltk except for the z-component (default fltz=0).

**x=** x-coordinate of observation position assuming that the wire source is centered at the origin and oriented along the x-axis (x>=0; observation position cannot be along the source-wire).

**y=** y-coordinate of observation position assuming that the wire source is centered at the origin and oriented along the x-axis (y>=0; observation position cannot be along the source-wire).

**ci=** current in the source wire in amps (default ci=1).  
**sl=** total length of the source wire in meters (default sl=1). The type of wire source used in the calculations is defined by fltk parameter icase, (cref; fltk icase).

**blwr=** lower normalized frequency used to compute the field frequency response before the transformation to the time domain (default blwr=.005).

**bufr=** upper normalized frequency used to compute the field frequency response before the transformation to the time domain (default bufr=10000).

**tol=** relative error desired for all Hankel, cosine, and sine transform calculations (default tol=1.e-4).

number of frequencies per decade at which to compute the magnetic field frequency response between `init blwr` and `bufr` (default `nf=12`). It is recommended that preliminary inversions be done with `nf=4` or `5` and the final one be done with `nf=12` to insure sufficient accuracy in the final result without using more computer time than is necessary.  
 (cref: `init blwr`, `bufr`).

`icase=` selects whether an infinitesimal or `finite-length` wire source model is to be used.  
 = 1 (default) infinitesimal wire source (horizontal electric dipole);  
 = 2 `finite-length` wire source, with `wire length` specified in `init parameter sl`.  
 (cref: `init sl`).

`cnctol=` relative error desired for adaptive integration in `finite-length` wire model calculation (default `cnctol=1.e-5`).  
 (cref: `init icase`, `intpe`).

`mev=` maximum number of function evaluations to be allowed for the `finite-length` wire integration (default `mev=300`). Setting `init cnctol` to a value smaller than the default may require `mev` to be set to a number larger than 300.  
 (cref: `init cnctol`, `intpe`).

`intpe=` type of integration algorithm (subprogram `canca4`) and convergence test desired for `finite-length` wire integration.  
 = 1,2,3 uses Newton-Cotes no. 4, quadrature;  
 = 1 for absolute error test (not generally recommended);  
 = 2 for "1-one" type error test;  
 = 3 for "1-infinity" type error test;  
 = 4 (default) for relative error test using adaptive Gaussian quadrature, and  
 = 5 for relative error test using non-adaptive Gaussian quadrature.  
 Note: The selected convergence test must be satisfied by both the real and the imaginary parts of the complex integral simultaneously.  
 (cref: `init cnctol`).

`nrune=` number of spline nodes to be calculated per digital filter interval for the `finite-length` wire integration (default `nrune=1`). Increasing `nrune` from 1 to 2 roughly doubles the execution time of

MQLVTHXYZ, but yields more accurate field value and is recommended for soundings made close to a long wire.

\$end [end of \$init parameters]

DATA MATRIX NOTES.

The data matrix is defined as the sequence of ordered rows:  $(g(i), x(i,j), j=1:m),$  where  $i$  is row number  $1, 2, \dots, m,$  and  $m = m+1$  if  $lwt=1$ , otherwise  $m = m+2$ . The data matrix is read on logical unit  $lwt$  (default 10) using an object-time format statement (see any Fortran manual). The number of items read depends on  $\$parms$   $m$ ,  $lwt$  and  $\$init$   $lwt$  as previously defined. The various data matrix options are summarized as follows:

- (a) Specific observation component, THEM soundings ( $lob < 3$ ),  $m=1$ , and  $max=3$  items per record):
1.  $g(i) = 1$ -th observation, where  $\$init$   $lob < 3$  defines the particular component.
  2.  $x(i,1) = 1$ -th time ( $x(i,1) > 0.0$  seconds).
  3.  $x(i,2) =$  standard deviation of observation  $i$  (include only if  $lwt=1$ ).
- Mixed observation types:  $x=1$  or  $z$ -component THEM soundings and/or Schlumberger soundings data ( $lob=4$  or  $5$ ;  $m=2$ , and  $max=4$  items per record):

1.  $g(i) = 1$ -th observation (where actual type is defined by  $x(i,2)$ ).
2.  $x(i,1) = 1$ -th time ( $x(i,1) > 0.0$  seconds) if  $x(i,2) < 3.0$  or corresponding  $AB/2$  for  $x(i,2) = 4.0$  ( $x(i,1) > 0.0$  meters).
3.  $x(i,2) =$  observation type in  $g(i)$ ; use  $x(i,2) = 1.0$  for  $x$ -component THEM data,  $= 2.0$  for  $z$ -component THEM data,  $= 3.0$  for  $z$ -component Schlumberger THEM data, or  $= 4.0$  for Schlumberger apparent resistivity data.
4.  $x(i,3) =$  standard deviation of observation  $i$  (include only if  $lwt=1$ ). Note: for joint inversion of Schlumberger apparent resistivity and THEM data, a weighted least squares should be used ( $lwt=1$  option) to produce residuals of near-equal magnitude.

EXAMPLES OF INPUT PARAMETERS AND DATA ORDERING.

1. Vertical magnetic field TEM soundings (default job=3, istep=0);

example 1.  
 \$parms n=60,k=6,m=1,iprt=-1,sp=1,alt=5,  
 b=1e0,.01,.2,.01,100,200\$  
 (2e14.5)  
 0.20961e-08 0.55166e-02  
 0.69697e-08 0.18316e-01  
 --- (etc. for 58 more observations) ---  
 \$init mm=3,s=1500,icase=1,job=5\$

2. Mixed observation types (job=4), s-component TEM (step response) and Schlumberger soundings data, and weighted observations (lwt=1);

example 2  
 \$parms n=100,k=8,m=2,iprt=-2,sp=1,lwt=1,alt=5,ip=2,ib=1,3,  
 b=1e0,1e9,1e0,.01,.2,.01,100,200\$  
 (4f10.0)  
 20.1 .05 2. .05  
 29.87 .06 2. .05  
 --- (etc. for rest of TEM soundings data) ---  
 108. 10. 4. 5.  
 120. 13. 4. 5.  
 --- (etc. for rest of Schlumberger soundings data) ---  
 \$init mm=3,job=4,icase=1,istep=1,s=1000\$

3. Vertical magnetic field TEM soundings (default istep=0), orientation option desired;

example 3  
 \$parms n=60,k=9,m=1,iprt=-1,sp=0,alt=5,ip=2,ib=1,2,  
 b=1e0,1e0,1e0,1e-6,1e-6,1e-6,.01,.2,.01,100,200\$  
 (2e14.5)  
 0.20961e-08 0.55166e-02  
 0.69697e-08 0.18316e-01  
 --- (etc. for 58 more observations) ---  
 \$init mm=3,s=1500,icase=1,job=5\$

SPECIAL OBJECT-FORMAT PHRASES.

If an existing data matrix file does not have the properly defined column ordering in the form (s(i),x(i,j),j=1,m)), then the Fortran "tn" format phrase may be used to begin at any column n in the data record. For example, the format (t41,f10.0,t1,3f10.0) will select s(i) using col.41-50 and x(i,1) beginning at col.1.

### MULTICS OPERATING INSTRUCTIONS.

1. Initially, one should add the following libraries (via the command "asr") to his search rules on the Denver Multics system after the working directory:  
>addEmodlinv>kauahikau>execsess;  
>addEmodlinv>wAnderson>libem;  
>addEmodlinv>wAnderson>libl; and >iml>imsl.

2. Either attach "file05" to a predetermined ascii (stream) parameter file, or let file05 default to "user-input" (i.e., the user's terminal). The order of parameters and data on file05 must be given as defined in the section PARAMETERS AND DATA REQUIRED above. To attach file05, type:  
to attach file05 vfile-parameter-filename

3. Attach "file10" to an input data-matrix ascii file if tail=10 (default) is used. If tail=5 is selected, then ignore this step, but include the data matrix following the object-time format on "file05"---see examples 1 and 2 above. In practice, it is usually best to use distinct files file05 and file10 for parameters and data respectively. To attach file10, type:  
to attach file10 vfile-data-filename

4. Set the underflow condition handler off by typing:  
setuffl-off
5. Execute Program MQLVTHXYZ by typing: malvthxyz

If file05 was not attached, then the user must anticipate the required title, \$param, object format, and \$unit to be typed on "user-input". Prompt messages are not printed on the terminal.

Note "file16" is the complete print file (normally disk on Multics); and "file06" is always the on-line terminal print file. File16 should either be deleted or printed to a line-printer after running Program MQLVTHXYZ. Also, file13 (if used) should be deleted after running the program. To submit the job as a batch job (called absentee on Multics), prepare step 1-5 above in a segment with ;absin suffix and use the "enter-abs-request" command.

### ERROR MESSAGES.

Most parameter and/or data errors are noted by self-explanatory messages appearing in the printed file(s); and the job is terminated. For example, the message

"error--some parms out of range" means that a violation (or omission) of a required parameter range has been committed in the parms namelist. Check all parms values; correct, and resubmit the job.

Exponent underflow may occur when the argument is less than 10\*\*38 on Multics; this is not fatal since 0.0 replaces all underflows. To suppress the underflow messages, the command "set-uf1-off" can be used prior to executing MQLVTHXYZ.

Exponent overflow and/or arithmetic overflow messages will terminate the run under Multics control. An overflow condition usually means a very poor initial parameter estimate was given in array b() for the model chosen; first check that all parms; finally data matrix values; and object-time format are correct. If no errors are found, then try to revise the model (mm) and/or use better guessed estimates for the starting parameters in array b().

If any parameter begins to approach zero or become unbounded during the least squares iterations, then one may fix (constrain) the parameter to a reasonable value, and restart the program to obtain a constrained least squares solution. This is usually required when the data are not sufficient to resolve all the parameters for the model mm chosen.

REFERENCES.

Anderson, W.L., 1974, Electromagnetic fields about a finite electric wire source; U.S. Geol. Survey Rept. USGS-GD-74-041, 205 p., avail. from U.S. Dept. Comm. NTIS, Springfield, Va. 22161 as Rept. PB-238-199.

Fourier and Hankel transform integrals; U.S. Geol. Survey Rept. USGS-GD-75-012, 119 p., avail. from U.S. Dept. Comm. NTIS, Springfield, Va. 22161 as Rept. PB-242-800.

1977, Marquardt inversion of vertical magnetic field measurements from a grounded wire source; Program MARQHZ, CYBER 74-28 Documentation; U.S. Geol. Survey Rept. USGS-GD-77-003, 76 p., avail. from U.S. Dept. Comm. NTIS, Springfield, Va. 22161 as Rept. PB-263-924.

1979a, Numerical integration of related Hankel transforms of orders 0 and 1 by adaptive digital filtering; Geophysics, v. 7, p. 1287-1305.

-----, 1979b, Program IMSLPM; Marquardt inversion of  
plane-wave frequency soundings; U.S. Geol. Survey  
Open-File Rept. 79-586, 37 p.

-----, 1979c, Program MARQLOPFS; Marquardt inversion of  
loop-loop frequency soundings; U.S. Geol. Survey  
Open-File Rept. 79-240, 75p.

Brown, K.M., and Dennis, J.E., 1972, Derivative free  
analyses of the Levenburg-Marquardt and Gauss  
algorithms for nonlinear least squares approximations;  
Numerische Mathematik, vol. 18, pp. 289-297.

Herriot, J.G., and Reinsch, C.H., 1976, Algorithm 507,  
Procedures for quintic natural spline interpolation;  
ACM Trans. on Math. Software, v. 2, p. 281-289.

International Mathematical and Statistical Libraries (IMSL),  
1977, 7500 Bellaire Blvd., 6th Floor, GNB Bldg.,  
Houston, Texas 77036.

Johansen, H.K., 1975, An interactive  
computer/graphic-display-terminal system for  
interpretation of resistivity soundings; Geophysical  
Prospecting, v. 23, p. 449-458.

Kauhikaua, J., 1978, Electromagnetic fields about a  
horizontal electric wire source of arbitrary length;  
Geophysics, v. 43, p. 1019-1022.

Kauhikaua, J. and Anderson, W.L., 1977, Calculation of  
standard transient and frequency soundings curves for a  
horizontal wire source of arbitrary length; U.S. Geol.  
Survey Rept. USGS-GD-77-007, 61 p., avail. from U.S.  
Dept. Comm. NIS, Springfield, Va. 22161 as Rept.  
PR-274-119.

Marquardt, D.W., 1963, An algorithm for least-squares  
estimation of nonlinear parameters; J. Soc. Indust.  
Appl. Math., v. 11, no. 2, pp. 431-441.

Patternson, T.N.L., 1973, Algorithm for automatic numerical  
integration over a finite interval [D1]; Assoc. for  
Comp. Mach. Comm., v. 16, p. 694-699.

Tabata, T. and Ito, R., 1973, Effective treatment of the  
interpolation factor in Marquardt's nonlinear  
least-squares fit algorithm; The Computer Journal, v.  
18, no. 3, pp. 250-251.



Appendix 1.-- Source listings

The attached subprograms are listed with beginnings line numbers in the following order:

```

C*****      PROGRAM MQLVTHXYZ      *****      00000010
SUBROUTINE ETHXYZ(Y,X,B,K,NN,TITLE,IOUT)      00000150
SUBROUTINE ITHXYZ(DV,XM,B,PRNT,NPRNT,N,TITLE,IOUT)      00000670
SUBROUTINE FCODE(DV,XM,BPARMS,PRNT,F,IF,IDER)      00003050
SUBROUTINE HXYZ(B2,HX,HY,HZ,IOB)      00005300
COMPLEX FUNCTION FLT(IFLAG,F)      00006060
SUBROUTINE TRNSI(FUNC,TMAX,TMIN,NT,T,U,MTD)      00006250
SUBROUTINE SCHSDG(SIG1,XMAX,XMIN,AB2,RHOA,NRHO,TOL)      00006700
REAL FUNCTION SPLINT(T,A,B,C,N,X,Y)      00007090
COMPLEX FUNCTION DCKERN(AMBDAA)      00007260
COMPLEX FUNCTION CSPLN(RT)      00007750
COMPLEX FUNCTION FINHX(B)      00008130
COMPLEX FUNCTION FINHY(B)      00008390
COMPLEX FUNCTION FINHZ(B)      00008800
COMPLEX FUNCTION F3(G)      00008930
COMPLEX FUNCTION F4(G)      00009010
SUBROUTINE RECURS(G,V1,F1)      00009090
SUBROUTINE FINF3(B1,B2,F31,F32)      00009350
COMPLEX FUNCTION FINITE(FUNC,BFIN)      00009460
COMPLEX FUNCTION ZHY(B,NEW,R)      00010440
COMPLEX FUNCTION ZHZ(B,NEW,R)      00010620
COMPLEX FUNCTION FUNINT(X)      00010770
COMPLEX FUNCTION I1K1(B8)      00010930
SUBROUTINE SPLINI(M,H,X,Y,A,B,C,IT,D,P,S)      00011230
  COMPLEX FUNCTION ZLAGH0(X,FUN,TOL,L,NEW)      00012430
  COMPLEX FUNCTION ZLAGH1(X,FUN,TOL,L,NEW)      00014670
  COMPLEX FUNCTION ZLAGF0(X,FUN,TOL,L,NEW)      00017030
  COMPLEX FUNCTION ZLAGF1(X,FUN,TOL,L,NEW)      00019500
SUBROUTINE IKS(B8,I1K1,IKDIF)      00021940
SUBROUTINE KELVIN(X,M,B)      00022580
SUBROUTINE QUINT(NY,Y,B,C,D,E,F)      00024380
SUBROUTINE QPOINT(NY,Y,B,C,D,E,F,X1,DELX,XX,YY)      00025150
COMPLEX FUNCTION CANC4(A1,B1,EP,M,N,FUN,MF,ESUM)      00025340
SUBROUTINE CQUAD(A,B,RESULT,K,EPSIL,NPTS,ICHECK,F,MEV)      00026910
COMPLEX FUNCTION CQSUB(A,B,EPSIL,NPTS,ICHECK,RELERR,F,MEV)      00030430
COMPLEX FUNCTION CQSUBA(A,B,EPSIL,NPTS,ICHECK,RELERR,F,MEV)      00031960
SUBROUTINE IMSLMQ(SUBZ,SUBEND)      00033370
SUBROUTINE FPXSSQ(C,N,KIP,F)      00036580
SUBROUTINE LNXSSQ(C,N,KIP,F)      00036910
SUBROUTINE SAVER(I,J)      00037040
COMPLEX FUNCTION ZHANKS(N,B,FUN,TOL,NF,NEW)      00037230
SUBROUTINE ERRMSG(MSG,M5,I6,I9)      00040650
SUBROUTINE WARN(MSG,M5,I6,I9,*)      00040880
  
```

- (1). Subprogram QUINT was converted from ALGOL to FORTRAN as published by Herriot and Reinsch (1976); Copyright 1976, Association for Computing Machinery, Inc.; permission to republish, all or in part, granted by ACM. ALGOL to FORTRAN conversion was done by Walt Anderson, USGS.
- (2). Subprograms QUAD, CASUB, and CASUBA are modified versions of subprograms QUAD, QSUB, and QSUBA, respectively, as published by Patterson (1973); Copyright 1973, Association for Computing Machinery, Inc.; permission to republish, all or in part, granted by ACM.

Copyright Notices

Note: The source code for all IMSL Library routines used (ZXSSQ, LEQIF, LUDECF, LUELMP, LINVIP, and UERTSI) are only available from International Mathematical and Statistical Libraries (1977), whose address is given in the references.

The current version of the source code may be obtained by writing directly to the author. A magnetic tape copy of the source code will be sent to requestors to be copied and returned to the author. This method of releasing the program was selected in order to satisfy requests for the latest updated version. The magnetic tape will be recorded in the following mode (unless otherwise requested):

Industry compatible; 9-track, unlabeled, EBCDIC mode; odd-parity, 800 dpi density, 80-character records (unlocked card images), and contained on one file.

Source Availability

```

***** PROGRAM MQLVTHXYZ *****                                00000010
: NONLINEAR LEAST SQUARES INVERSION OF THREE COMPONENT          00000020
: MAGNETIC FIELD (TDEM) DATA USING A MODIFIED                 00000030
: MARQUARDT-LEVENBURG ALGORITHM (FROM IMSL LIBRARY)           00000040
:                                                                00000050
: WRITTEN BY JIM KAUAHIKAUA, USGS, DENVER, COLORADO           00000060
: LAST MODIFIED: NOVEMBER, 1978                                00000070
:                                                                00000080
: EXTERNAL ITHXYZ,ETHXYZ                                       00000090
: CALL INITREF('>UDD>EMODL_INV>JKAUAHIKAUA>ARCHIVES', 'HXYZINV_OLIB', 00000100
1 'FCODE')                                                       00000110
: CALL IMSLMQ(ITHXYZ,ETHXYZ)                                    00000120
: STOP                                                         00000130
: END                                                           00000140

SUBROUTINE ETHXYZ(Y,X,B,K,NN,TITLE,IOUT)                          00000150
                                                                00000160
: TERMINATION ROUTINE FOR COMPUTER INVERSION OF                00000170
: THREE-COMPONENT MAGNETIC FIELD TDEM DATA                    00000180
:                                                                00000190
: CHARACTER*5 TITLE                                           00000200
: DIMENSION Y(1),X(200,5),B(1),TITLE(16)                     00000210
: COMMON/CXYZ/A1,A2,J1,J2,JN(3),A6,A7,A8,A9,A10,A11,SCALE,U1,U2,J3, 00000220
$ IOB,NEXTRA,NPARMS,B1,B2,B3,B4,B5,J4,J5,J6                    00000230
: DATA DCON/57.295779/                                        00000240
: WRITE(6,10) TITLE                                           00000250
: IF(IOUT.EQ.1) WRITE(16,10) TITLE                             00000260
10 FORMAT(12H1T H X Y Z--,5X,16A5//)                            00000270
: IF(NEXTRA.LT.6) GO TO 100                                    00000280
: THETA=B(4)*DCON                                             00000290
: PHI=B(5)*DCON                                               00000300
: OMEGA=B(6)*DCON                                             00000310
: WRITE(6,20) THETA,PHI,OMEGA                                  00000320
: IF(IOUT.EQ.1) WRITE(16,20) THETA,PHI,OMEGA                 00000330
20 FORMAT(// " ORIENTATION OF MEASUREMENT AXES WITH RESPECT TO WIRE", 00000340
$ " AXES"/                                                       00000350
$ " SPHERICAL COORDS OF Z MSMT AXIS: THETA=",F10.6," DEGREES"/ 00000360
$ 36X,"PHI=",F10.6," DEGREES"/                                00000370
$ " ROTATION AROUND Z AXIS: OMEGA=",F10.6," DEGREES"//)      00000380
100 N=1                                                         00000390
: IF(IOB.GT.3) N=3                                           00000400
: WRITE(6,110) (B(I),I=1,N)                                   00000410
: IF(IOUT.EQ.1) WRITE(16,110) (B(I),I=1,N)                   00000420
110 FORMAT(" PRIMARY FIELD FACTORS:", 3E16.8//)                 00000430
: N=(NPARMS+1)/2                                             00000440
: WRITE(6,130)                                                00000450
: IF(IOUT.EQ.1) WRITE(16,130)                                 00000460
130 FORMAT(28H FINAL UNSCALED PARAMETERS--/                      00000470
$ 32X,11HRESISTIVITY,2X,3HLYR,10X,5HDEPTH/)                   00000480
: DO 150 I=1,N                                                00000490
: RHO=1./B(NEXTRA+I)                                         00000500
: WRITE(6,140) I,B(NEXTRA+I),RHO                             00000510

```



```

EQUIVALENCE (M,MM),(L,ICASE),(BL,BLWR),(BU,BUPR) 00001030
NAMELIST/INIT/TOL,METHOD,CNCTOL,INTYPE,NRUN,MEV 00001040
1,NF,ISTEP,X,Y,CI,SL,MM,M,IFLTX,IFLTY,IFLTZ,IOB,L,ICASE,BLWR,BUPR 00001050
C 00001060
C CALL DELETE_EXTERNAL_VARIABLES('SAVEC') 00001070
C 00001080
WRITE(6,100) TITLE 00001090
100 FORMAT(12H1T H X Y Z--,5X,16A5/) 00001100
IF(IOUT.EQ.1) WRITE(16,100) TITLE 00001110
C--SET DEFAULTS 00001120
M=1 00001130
ND=N 00001140
L=1 00001150
BLWR=-1E0 00001160
BUPR=-1E0 00001170
CI=1.0 00001180
SL=1.0 00001190
TOL=1.E-4 00001200
METHOD=2 00001210
MEV=300 00001220
NRUN=1 00001230
INTYPE=4 00001240
CNCTOL=1.E-5 00001250
ISTEP=0 00001260
FNYQ=1200 00001270
BNYQ=1E29 00001280
C 00001290
C FNYQ IS SET TO HAVE NO EFFECT AND IS NOT A PARAMETER. FOR INFO 00001300
C ON ITS ORIGINAL PURPOSE, SEE KAUAHIKAUA AND ANDERSON, 1977 00001310
C 00001320
NF=0 00001330
IFLTX=0 00001340
IFLTY=0 00001350
IFLTZ=0 00001360
IOB=3 00001370
C 00001380
READ(5,INIT) 00001390
C 00001400
IF(IOB.GT.0.AND.IOB.LE.6.AND.(L.EQ.1.OR.L.EQ.2).AND.SL.GE.0.0.AND. 00001410
$ CI.GT.0.0.AND.(ISTEP.EQ.1.OR.ISTEP.EQ.0).AND.X.GE.0.0.AND.Y.GE. 00001420
$ 0.0.AND.FNYQ.GT.0.0.AND.M.GT.0.AND.M.LT.10.AND.TOL.GE.0.0.AND. 00001430
$ (METHOD.EQ.1.OR.METHOD.EQ.2).AND.CNCTOL.GE.0.0.AND.INTYPE.GT.0. 00001440
$ AND.INTYPE.LE.4.AND.NRUN.GT.0.AND.MEV.GT.0) GO TO 4 00001450
WRITE(6,2) 00001460
2 FORMAT(" SOME $INIT PARMS OUT OF RANGE...") 00001470
STOP 00001480
4 R=SQRT(X*X+Y*Y) 00001490
SL=SL/2. 00001500
C 00001510
IF(BLWR.LT.0E0) BLWR=.005 00001520
IF(BUPR.LT.0E0) BUPR=1E4 00001530
C 00001540

```

```
C--CALCULATE ESTIMATE OF PRIMARY FIELD STRENGTH
C FROM DATA
C
R12=(X+SL)**2+Y*Y
R22=(X-SL)**2+Y*Y
R1=SQR(R12)
R2=SQR(R22)
GX=CI*(1./R22-1./R12)*Y/12.56637062
GXO=X*Y/(6.28318531**R**4)
GY=-CI*((X-SL)/R22-(X+SL)/R12)/12.56637062
GYO=-((X*X-Y*Y)/(12.56637062**R**4)
GZ=CI*((X+SL)/R1-(X-SL)/R2)/(12.56637062**Y)
GZO=Y/(12.56637062**R**3)
NRHO=0
DO 1 I=1,3
  NCOMP(I)=0
  TOX=EO
  TNX=EO
  TOY=EO
  TNY=EO
  TOZ=EO
  TNZ=EO
  NFRNT=2
  IF(IOR,GT,3) GO TO 5
  IS=IOB
  IM=IOB
  GO TO 6
  IS=1
  IM=4
  NFRNT=3
  6 DO 30 I=IS,IM
    KOUNT=0
    DO 16 J=1,N
      IF(IOR,LE,3) GO TO 15
      IF(XM(J,2)+GE,1.,OR,XM(J,2)+LE,4.,OR,I.GT,IS) GO TO 14
      WRITE(6,13) J
      FORMAT(' DATA MATRIX ENTRY (' ,I3,' ,2) OUT OF RANGE')
      STOP
      IF(FIX(XM(J,2))+NE,I) GO TO 16
      KOUNT=KOUNT+1
      DT(KOUNT)=XM(J,1)
      AV(KOUNT)=DV(J)
      CONTINUE
      IF(KOUNT,EG,0) GO TO 30
      GO TO (17,18,19,20)*I
      TOX=DT(1)
      TNX=DT(KOUNT)
      NCOMP(1)=KOUNT
      GO TO 30
00001550
00001560
00001570
00001580
00001590
00001600
00001610
00001620
00001630
00001640
00001650
00001660
00001670
00001680
00001690
00001700
00001710
00001720
00001730
00001740
00001750
00001760
00001770
00001780
00001790
00001800
00001810
00001820
00001830
00001840
00001850
00001860
00001870
00001880
00001890
00001900
00001910
00001920
00001930
00001940
00001950
00001960
00001970
00001980
00001990
00002000
00002010
00002020
00002030
00002040
00002050
00002060
```

```

18      TOY=DT(1)                                00002070
        TNY=DT(KOUNT)                            00002080
        NCOMP(2)=KOUNT                           00002090
        GO TO 30                                  00002100
19      TOZ=DT(1)                                00002110
        TNZ=DT(KOUNT)                            00002120
        NCOMP(3)=KOUNT                           00002130
        GO TO 30                                  00002140
20      NRHO=KOUNT                               00002150
        ABMIN=DT(1)                              00002160
        ABMAX=DT(KOUNT)                          00002170
30      CONTINUE                                 00002180
        IF(IOB.LT.5) GO TO 40                    00002190
        DO 35 I=1,3                              00002200
35      NCOMP(I)=1                              00002210
        TMAX=AMAX1(TNX,TNY,TNZ)                 00002220
        TNX=TMAX                                 00002230
        TNY=TMAX                                 00002240
        TNZ=TMAX                                 00002250
        TMIN=TMAX                               00002260
        IF(TOX.LT.TMIN.AND.TOX.GT.OEO) TMIN=TOX 00002270
        IF(TOY.LT.TMIN.AND.TOY.GT.OEO) TMIN=TOY 00002280
        IF(TOZ.LT.TMIN.AND.TOZ.GT.OEO) TMIN=TOZ 00002290
        TOX=TMIN                                00002300
        TOY=TMIN                                00002310
        TOZ=TMIN                                00002320
40      SLL=2.*SL                                00002330
C
C SET UNUSED IFLTX, IFLTY, IFLTZ TO ZERO        00002340
C
45      GO TO (46,47,48,47,49,49),IOB           00002370
46      IFLTY=0                                  00002380
47      IFLTZ=0                                  00002390
        IF(IOB.EQ.4.OR.IOB.EQ.1) GO TO 49        00002400
48      IFLTX=0                                  00002410
        IF(IOB.EQ.2) GO TO 49                   00002420
        IFLTY=0                                  00002430
49      CONTINUE                                 00002440
        WRITE(6,55) ISTEP,X,Y,R,CI,SLL,L,IFLTX,IFLTY,IFLTZ 00002450
        IF(IOB.EQ.1) WRITE(16,55) ISTEP,X,Y,R,CI,SLL,L,IFLTX,IFLTY,IFLTZ 00002460
55      FORMAT(/19H DATA SET PARMS ARE/7H ISTEP=,I2,2X,2HX=,E12.4, 00002470
        12X,2HY=,E12.4,2X,2HR=,E12.4,2X,8HCURRENT=,E12.4/ 00002480
        215H SOURCE LENGTH=,E12.4,2X,3H L=,I2/ 00002490
        4 8H IFLTX=,I2,8H IFLTY=,I2,8H IFLTZ=,I2) 00002500
        SCALE=CI                                00002510
        IF(L.EQ.2) GO TO 56                     00002520
        SL=0.0                                  00002530
        SCALE=SCALE*SLL                        00002540
        GX=GX0*CI*SLL                          00002550
        GY=GY0*CI*SLL                          00002560
        GZ=GZ0*CI*SLL                          00002570
C--SET EQUIVALENT VALUES FOR INITIALIZING COMMON AREAS 00002580

```





```

C FINITE LENGTH WIRE SOURCE. 00003100
C 00003110
C T,VX,VY,VZ WORK ARRAYS INTERNAL TO FCODE 00003120
C DV VOLTAGE DATA ARRAY 00003130
C XM(1,I) TIMES ARRAY 00003140
C XM(2,I) WEIGHTS ARRAY IF IOB.LE.3 00003150
C INDICES ARRAY IF IOB.GT.3 00003160
C (1=HX, 2=HY, 3=HZ, 4=RHOA) 00003170
C XM(3,I) WEIGHTS ARRAY IF IOB.GT.3 00003180
C ND NUMBER OF DATA POINTS 00003190
C P,S,PS WORK ARRAYS REQUIRED BY SPLIN1 00003200
C 00003210
C THE PARAMETER IOB CONTROLS THE OPERATION OF FCODE: 00003220
C 00003230
C IOB=1 CALCULATE X-COMPONENT ONLY 00003240
C IOB=2 CALCULATE Y-COMPONENT ONLY 00003250
C IOB=3 CALCULATE Z-COMPONENT ONLY 00003260
C IOB=4 CALCULATE MIXED COMPONENTS, INCLUDING SCHLUMBERGER RHOA 00003270
C IOB=5 SAME AS IOB=4 WITH ORIENTATION PARAMETERS. 00003280
C 00003290
C WRITTEN BY JIM KAUAHIKAWA, USGS, DENVER, COLORADO 00003300
C LAST MODIFIED NOVEMBER, 1978 00003310
C 00003320
C DIMENSION BPARMS(1),P(50),S(50),DV(1),XM(200,5),PS(2) 00003330
C COMMON/MODEL/K,DD,NLAYER 00003340
C COMMON/THICK/D 00003350
C COMMON/PASS/CHX 00003360
C COMMON/PARM/ISTEP,XX,YY,R,SIG1,BNYQ,M,TOL 00003370
C COMMON/FIN/A1,A2,A3,A4,SIGG1,A6,A7 00003380
C COMMON/SPLXYZ/AX(100),BX(100),DX(100),AY(100),BY(100),DY(100), 00003390
C $ AZ(100),BZ(100),DZ(100),NB,B(100),SPX(100),SPY(100),SPZ(100),IW 00003400
C COMMON/CXYZ/FN,SL,MTD,NF,NCOMP(3),TOX,TNX,TOY,TNY,TOZ,TNZ,SCALE, 00003410
C $ BLWR,BUPR,ND,IOB,NEXTRA,NPARMS,GX,GY,GZ,GT,X0,IFLTX,IFLTY,IFLTZ 00003420
C COMMON/DCSAV/ARHO(100),BRHO(100),DRHO(100),AB2(100),RHOA(100),NRHO 00003430
C $ ,ABMAX,ABMIN 00003440
C COMMON/SAVEC/TX(50),VX(50),NTX,TY(50),VY(50),NTY,TZ(50),VZ(50), 00003450
C $ NTZ,CPX,CPY,CPZ,PARMS(20) 00003460
C 00003470
C GX,GY, AND GZ ARE THE PREDICTED PRIMARY FIELDS (FROM ITHXYZ) 00003480
C SL IS HALF SOURCE LENGTH IF FINITE WIRE MODELS ARE TO BE USED, 00003490
C IS ZERO OTHERWISE. 00003500
C 00003510
C REAL K(10),D(9),DD(9),PRNT(5) 00003520
C EXTERNAL CSPLN 00003530
C COMPLEX COMPL,FINHX,FINHY,FINHZ,CHX,CHY,CHZ,FLT 00003540
C DATA PS/0.0,0.0/ 00003550
C 00003560
C IF(IF.GT.1) GO TO 150 00003570
C 00003580
C MODELS ARE CALCULATED ONLY IF IF=1, OTHERWISE THEY ARE RETRIEVED 00003590
C FROM STORAGE IN COMMON BLOCK /SAVEC/ 00003600
C 00003610
  
```

```

00003620      ISKIP=1
00003630      CHECK BPARAMS TO SEE IF SAME AS LAST FARMS
00003640      DO 5 I=1,NPARAMS
00003650
00003660      IF (PARMS(I).NE.BPARAMS(NEXTRA+1)) ISKIP=0
00003670      PARMS(I)=BPARAMS(NEXTRA+1)
00003680      IF (IOB.EQ.1.OR.IOB.GT.3) CPX=BPARAMS(1)
00003690      IF (IOB.EQ.2) CPY=BPARAMS(1)
00003700      IF (IOB.GT.3) CPY=BPARAMS(2)
00003710      IF (IOB.EQ.3) CPZ=BPARAMS(1)
00003720      IF (IOB.GT.3) CPZ=BPARAMS(3)
00003730      IF (IOB.GT.3) CPZ=BPARAMS(3)
00003740      IF BPARAMS SAME AS LAST FARMS, SKIP MODEL COMPUTATION
00003750
00003760      IF (ISKIP.EQ.1) GO TO 150
00003770      SIGI=PARMS(1)
00003780      SIGI=SIGI
00003790      M1=M-1
00003800      BMIN=1.0
00003810      BMAX=1.0
00003820      DO 10 I=1,M1
00003830      K(I)=PARMS(I)/SIGI
00003840      C---MAXIMUM NORMALIZED FREQUENCY, BMAX, IS DETERMINED BY SMALLEST
00003850      C CONDUCTIVITY CONTRAST IN MODEL OR BY THE NYQUIST FREQUENCY, FN.
00003860      BMAX=AMINI(BMAX,K(I))
00003870      C---MINIMUM NORMALIZED FREQUENCY, BMIN, IS DETERMINED BY LARGEST
00003880      C CONDUCTIVITY CONTRAST IN MODEL
00003890      BMIN=AMAXI(BMIN,K(I))
00003900      D(I)=PARMS(M+1)
00003910      K(M)=PARMS(M)/SIGI
00003920      BMIN=AMAXI(BMIN,K(M))
00003930      BMAX=AMINI(BMAX,K(M))
00003940      CONT=GT/SIGI
00003950      XTMAX=CONT*TNX*1.15
00003960      XTMIN=CONT*TOX
00003970      YTMAX=CONT*TNX*1.15
00003980      YTIMIN=CONT*TOY
00003990      ZTMAX=CONT*TNZ*1.15
00004000      ZTIMIN=CONT*TOZ
00004010      VX(1)=1.0
00004020      VY(1)=1.0
00004030      VZ(1)=3.*CONT*GZ
00004040
00004050      IF (NF.GT.12.OR.NF.LE.0) NF=12
00004060
00004070      C---MODEL CALCULATED ON A SPLINE INTERPOLATED FREQUENCY FUNCTION
00004080      C COMPUTED AT NF FREQUENCIES PER DECADE
00004090      C RANGE AUTOMATICALLY DETERMINED FROM THE MAXIMUM AND MINIMUM
00004100
00004110
00004120
00004130

```

```

C CONDUCTIVITIES IN MODEL 00004140
C 00004150
C 00004160
  BMAX=BUFR/BMAX 00004170
  BMIN=BLWR/BMIN 00004180
C 00004190
15 NB=AIN(T(NF*ALOG10(BMAX/BMIN)))+2 00004200
  NB1=NB+1 00004210
  X0=ALOG10(BMAX)+0.25 00004220
  ISAV=ISTEP 00004230
  IF(MTD,EQ,1) ISTEP=0 00004240
  IF(NB,LE,100) GO TO 18 00004250
C 00004260
C NB MAY NOT EXCEED THE SIZE OF THE ARRAYS IN COMMON BLOCK /SPLXYZ/ 00004270
C 00004280
  WRITE(6,95) NB 00004290
95 FORMAT(" FCODE:  NB,GT,100, NB=",I5) 00004300
  STOP 00004310
18 DO 1 J=1,NB 00004320
  I=NB1-J 00004330
  X=X0-J/FLOAT(NF) 00004340
  B(I)=X 00004350
  X=10E0**X 00004360
  F=X*CONT/6.2831853 00004370
  IF(SL,EQ,0E0) CALL HXYZ(X,CHX,CHY,CHZ,IOB) 00004380
  GO TO (17,20,40,20,20),IOB 00004390
17 IF(SL,GT,0E0) CHX=FINHX(X) 00004400
20 IF(SL,GT,0E0,AND,IOB,GT,1) CHY=FINHY(X) 00004410
  IF(IFLTX,NE,0) CHX=CHX*FLT(IFLTX,F) 00004420
  SPX(I)=REAL(CHX) 00004430
  IF(IOB,EQ,1) GO TO 1 00004440
  IF(IFLTY,NE,0) CHY=CHY*FLT(IFLTY,F) 00004450
  SPY(I)=REAL(CHY) 00004460
  IF(IOB,LT,3) GO TO 1 00004470
40 IF(SL,GT,0E0) CHZ=FINHZ(X) 00004480
  IF(IFLTZ,NE,0) CHZ=CHZ*FLT(IFLTZ,F) 00004490
  SPZ(I)=REAL(CHZ) 00004500
  1 CONTINUE 00004510
  ISTEP=ISAV 00004520
C 00004530
C NOTE:  B() ARRAY IS ACTUALLY ALOG10(B) 00004540
C 00004550
  IF(NCOMP(1),NE,0) CALL SPLINI(NB,0,B,SPX,AX,BX,DX,0,PS,P,S) 00004560
  IF(NCOMP(2),NE,0) CALL SPLINI(NB,0,B,SPY,AY,BY,DY,0,PS,P,S) 00004570
  IF(NCOMP(3),NE,0) CALL SPLINI(NB,0,B,SPZ,AZ,BZ,DZ,0,PS,P,S) 00004580
C 00004590
C THE SWITCH IW WORKS WITH COMMON BLOCK /SPLXYZ/ IN ROUTINE CSPLN 00004600
C TO INTERPOLATE EITHER THE X, Y, OR Z COMPONENT 00004610
C 00004620
  IW=1 00004630
  IF(NCOMP(1),NE,0) CALL TRNSI(CSPLN,XTMAX,XTMIN,NTX,TX,VX,MTD) 00004640
  IW=2 00004650

```

```

IF(NCOMP(2).NE.0) CALL TRNSI(CSPLN,YTMAX,YTMIN,NTY,TY,VY,MTD) 00004660
IW=3 00004670
IF(NCOMP(3).NE.0) CALL TRNSI(CSPLN,ZTMAX,ZTMIN,NTZ,TZ,VZ,MTD) 00004680
C 00004690
IF(IOB.LE.3.OR.NRHO.EQ.0) GO TO 3 00004700
CALL SCHSDG(SIG1,ABMAX,ABMIN,AB2,RHOA,NRHO,TOL) 00004710
DO 2 I=1,NRHO 00004720
2 AB2(I)=ALOG(AB2(I)) 00004730
CALL SPLINI(NRHO,0,AB2,RHOA,ARHO,BRHO,DRHO,0,PS,P,S) 00004740
C 00004750
3 NT=MAX0(NTX,NTY,NTZ) 00004760
IF(ISTEP.EQ.0.OR.MTD.EQ.1) GO TO 35 00004770
DO 30 I=1,NT 00004780
IF(NCOMP(1).NE.0) VX(I)=VX(I)*SCALE+GX 00004790
IF(NCOMP(2).NE.0) VY(I)=VY(I)*SCALE+GY 00004800
30 IF(NCOMP(3).NE.0) VZ(I)=VZ(I)*SCALE+GZ 00004810
35 DO 50 I=1,NT 00004820
IF(I.LE.NTX) TX(I)=ALOG(TX(I)) 00004830
IF(I.LE.NTY) TY(I)=ALOG(TY(I)) 00004840
IF(I.LE.NTZ) TZ(I)=ALOG(TZ(I)) 00004850
50 CONTINUE 00004860
IF(NCOMP(1).NE.0) CALL SPLINI(NTX,0,TX,VX,AX,BX,DX,0,PS,P,S) 00004870
IF(NCOMP(2).NE.0) CALL SPLINI(NTY,0,TY,VY,AY,BY,DY,0,PS,P,S) 00004880
IF(NCOMP(3).NE.0) CALL SPLINI(NTZ,0,TZ,VZ,AZ,BZ,DZ,0,PS,P,S) 00004890
C 00004900
150 TAU=CONT*XM(IF,1) 00004910
TAU=ALOG(TAU) 00004920
INDEX=IOB 00004930
IF(IOB.GT.3) INDEX=IFIX(XM(IF,2)) 00004940
IF(INDEX.EQ.4) GO TO 188 00004950
IF(NEXTRA.EQ.6) GO TO 180 00004960
IF(INDEX.EQ.1) F=CPX*SPLINT(TAU,AX,BX,DX,NTX,TX,VX) 00004970
IF(INDEX.EQ.2) F=CPY*SPLINT(TAU,AY,BY,DY,NTY,TY,VY) 00004980
IF(INDEX.EQ.3) F=CPZ*SPLINT(TAU,AZ,BZ,DZ,NTZ,TZ,VZ) 00004990
GO TO 190 00005000
C 00005010
C ORIENTATION PARAMETERS OPTION HAS BEEN CHOSEN 00005020
C 00005030
180 FX=CPX*SPLINT(TAU,AX,BX,DX,NTX,TX,VX) 00005040
FY=CPY*SPLINT(TAU,AY,BY,DY,NTY,TY,VY) 00005050
FZ=CPZ*SPLINT(TAU,AZ,BZ,DZ,NTZ,TZ,VZ) 00005060
CS1=COS(BPARMS(4)) 00005070
SN1=SIN(BPARMS(4)) 00005080
CS2=COS(BPARMS(5)) 00005090
SN2=SIN(BPARMS(5)) 00005100
CS3=COS(BPARMS(6)) 00005110
SN3=SIN(BPARMS(6)) 00005120
IF(INDEX.EQ.3) GO TO 185 00005130
FXP=FX*CS1+SN1*(-FZ*CS2+FY*SN2) 00005140
FYP=FY*CS1-SN1*(FZ*CS2+FX*SN2) 00005150
IF(INDEX.EQ.1) F=FXP*CS3+FYP*SN3 00005160
IF(INDEX.EQ.2) F=FYP*CS3-FXP*SN3 00005170

```

```

      GO TO 190
185 F=FZ*CS1+SN1*(FX*CS2+FY*SN2)
      GO TO 190
C
188 F=SPLINT(ALOG(XM(IF,1)),ARHO,BRHO,DRHO,RRHO,AB2,RHOA)
C
190 IIMAX=2
      IF(IOB.GT.3) IIMAX=3
      DO 200 II=1,IIMAX
200   PRNT(II)=XM(IF,II)
      RETURN
      END
C
      SUBROUTINE HXYZ(B2,HX,HY,HZ,IOB)
C
C HXYZ CALCULATES THREE COMPONENTS OF THE MAGNETIC FIELD
C OF AN ELECTRIC DIPOLE AT THE ORIGIN AND ORIENTED ALONG THE X-AXIS.
C
C IOB=1 FOR HX ONLY
C IOB=2 FOR HY ONLY
C IOB=3 FOR HZ ONLY
C IOB=4,5 FOR MIXED COMPONENTS.
C
C CALLS ZHANKS,SAVER,F3,F4,IKS
C
      COMPLEX F3,F4,ZHANKS,XX,YY,ZZ,HX,HY,HZ,ONE,ZERO,I1K1,IKDIF,CB
      COMMON/PAARM/ISTEP,X,Y,R,SIG1,BNYQ,M,TOL
      COMMON/MODEL/K,DD,NLAYER
      COMMON/THICK/D
      EXTERNAL F3,F4
      REAL K(10),D(9),DD(9)
      DOUBLE PRECISION B8
      DATA ONE,ZERO/(1.0,0.0),(0.0,0.0)/
      B=SQRT(B2)
      HX=ZERO
      HY=ZERO
      HZ=ZERO
      R2=R*R
      C=Y/(R2*R)
      XR2=X*X/R2
      CB=CMPLX(B,B)
      IF(M.EQ.1) GO TO 10
      M1=M-1
      DEL=R/B
      DEL2=DEL*DEL
      DO 4 I=1,M1
4       DD(I)=2.*D(I)/DEL
C
C HX AND HY REQUIRE THE SAME TWO HANKEL TRANSFORMS
C
      NEW=1
      IF(IOB.EQ.3) GO TO 5

```

00005180  
 00005190  
 00005200  
 00005210  
 00005220  
 00005230  
 00005240  
 00005250  
 00005260  
 00005270  
 00005280  
 00005290  
 00005300  
 00005310  
 00005320  
 00005330  
 00005340  
 00005350  
 00005360  
 00005370  
 00005380  
 00005390  
 00005400  
 00005410  
 00005420  
 00005430  
 00005440  
 00005450  
 00005460  
 00005470  
 00005480  
 00005490  
 00005500  
 00005510  
 00005520  
 00005530  
 00005540  
 00005550  
 00005560  
 00005570  
 00005580  
 00005590  
 00005600  
 00005610  
 00005620  
 00005630  
 00005640  
 00005650  
 00005660  
 00005670  
 00005680

```

NEW=0                                00005690
YY=ZHANKS(1,B,F3,TOL,LW,1)          00005700
CALL SAVER(1,1)                       00005710
XX=ZHANKS(0,B,F4,TOL,LW,0)          00005720
HX=2.*(2.*YY/B-XX)*X*Y/(DEL2*R2)    00005730
HY=(XR2-1.)*2.*XX/DEL2+2.*(1.-2.*XR2)*YY/(R*DEL) 00005740
IF(IOB.LT.3) GO TO 10                 00005750
5 HZ=ZHANKS(1,B,F4,TOL,LW,NEW)*CMPLX(2.0*B2*C,0.0) 00005760
10 B8=.70710678118654D0*B            00005770
C                                     00005780
C CALCULATE THE MODIFIED BESSEL FUNCTIONS OF THE FIRST AND 00005790
C SECOND KINDS FOR THE HX AND HY HALFSpace RESPONSES.    00005800
C                                     00005810
C IF(IOB.NE.3) CALL IKS(B8,I1K1,IKDIF) 00005820
C ZZ=ZERO                                           00005830
C IF(IOB.LT.3) GO TO 12                          00005840
C ZZ=ONE-(ONE+CB+CMPLX(OE0,2E0*B2/3E0))*CEXP(-CB) 00005850
C ZZ=ZZ*CMPLX(OE0,-3E0*C/B2)                    00005860
C                                     00005870
C SUBTRACT THE PRIMARY FIELD VALUES IF ISTEP = 1      00005880
C                                     00005890
C 12 IF(ISTEP.EQ.0) GO TO 20                      00005900
C IF(IOB.EQ.3) GO TO 15                          00005910
C IKDIF=IKDIF-CMPLX(1.0,0.0)                    00005920
C I1K1=I1K1-CMPLX(0.5,0.0)                      00005930
C IF(IOB.LT.3) GO TO 20                          00005940
C 15 ZZ=ZZ-CMPLX(C,0.0)                          00005950
C                                     00005960
C 20 HX=HX+2.*X*Y*IKDIF/(R2*R2)                  00005970
C HY=HY+2.*(Y*Y*IKDIF/R2-I1K1)/R2              00005980
C HZ=HZ+ZZ                                       00005990
C DENOM=12.56637062*B2**ISTEP                   00006000
C HX=HX/DENOM                                    00006010
C HY=HY/DENOM                                    00006020
C HZ=HZ/DENOM                                    00006030
C RETURN                                         00006040
C END                                           00006050

COMPLEX FUNCTION FLT(IFLAG,F)          00006060
C                                     00006070
C USER-DEFINED FUNCTION                     00006080
C IFLAG IS A PARAMETER PASSED FROM MQLVTHXYZ VIA $INIT 00006090
C PARMS IFLTX, IFLTY, OR IFLTZ            00006100
C F IS THE FREQUENCY IN HERTZ             00006110
C                                     00006120
C COMPLEX ONE                               00006130
C DATA ONE/(1E0,0E0)/                     00006140
C IF(IFLAG.EQ.0) RETURN                     00006150
C                                     00006160
C EXAMPLE: SINGLE-POLE LOW PASS FILTER WHERE 00006170
C IFLAG IS USED AS THE CORNER FREQUENCY    00006180
C                                     00006190

```

```

00006200 CF=IFLAG*6.283185
00006210 OMEGA=F*6.283185
00006220 FLT=ONE/CMPLX(-CF,OMEGA)
00006230 RETURN
00006240 END

SUBROUTINE TRNSI(FUNC,TMAX,TMIN,NT,T,V,MTD)
C IF ISTEP = 0, THE COSINE TRANSFORM OF THE REAL
C PART OF FUNC IS CALCULATED USING THE LAGGED
C CONVOLUTION OPERATOR ZLAGFO TO YIELD THE
C IMPULSE RESPONSE.
C IF ISTEP = 1, THE STEP RESPONSE IS CALCULATED AS
C - THE INTEGRAL OVER TIME OF THE IMPULSE
C RESPONSE IF MTD = 1,
C - THE SINE TRANSFORM OF THE REAL PART OF FUNC
C DIVIDED BY THE FREQUENCY USING THE LAGGED
C CONVOLUTION OPERATOR ZLAGF1 IF MTD = 2.
C ***** ZLAGFO AND ZLAGF1 FROM ANDERSON 1975 *****
C
C DIMENSION T(100),V(100)
C
C EXTERNAL FUNC,ZLAGFO,ZLAGF1,CON
C COMMON/PARM/ISTEP,XX,YY,R,SIGT,BNYQ,M,TOL
C NT=AINI(0.5+5.*ALOG(TMAX/TMIN))+1
C NT1=NT+1
C X0=ALOG(TMAX)+0.2
C C=4.0E-7*3.1415927
C X1=C*SIGT*R*R/2.
C C=2./((X1*3.1415927)
C V0=V(1)
C NEM=1
C ISAV=ISTEP
C IF(MTD,EG,1) ISTEP=0
C DO 2 J=1,NT
C I=NT1-J
C X=X0-0.2*XJ
C V(1)=0.0
C T(1)=EXP(X)
C IF(ISTEP,EG,1) GO TO 3
C CON=C*XLAGFO(X,FUNC,TOL,LM,NEW)/T(1)
C GO TO 4
C 3 CON=C*XLAGF1(X,FUNC,TOL,LM,NEW)/T(1)
C 4 V(1)=REAL(CON)
C 2 NEM=0
C ISTEP=ISAV
C IF(MTD,EG,2,OR,ISTEP,EG,0) GO TO 10
C CALL INTEG(NT,T,V,V0)
C DO 9 I=1,NT
C V(I)=V(I)*XI
C 9 RETURN
C 10 END

```

```

00006700 SUBROUTINE SCHSDG(SIG1,XMAX,XMIN,AB2,RHOA,NRHO,TOL)
00006710
00006720 COMPUTE SCHLUMBERGER APPARENT RESISTIVITY SOUNDING CURVE USING
00006730 LAGGED CONVOLUTION
00006740 DIMENSION AB2(1),RHOA(1)
00006750 COMPLEX ZLAGHT,CMP
00006760 COMMON/MODEL/RK(10),DNDRM(9),M
00006770 COMMON/THICK/D(9)
00006780
00006790 MUST TRANSFER MODEL FROM EM COMMON BLOCK, MODEL,
00006800 TO DC COMMON BLOCK DCMPL,
00006810
00006820 COMMON/DCMDL/SIG(10),DDC(9),NLYR,INDEX
00006830 EXTERNAL DCKERN
00006840
00006850 NRHO=AINT(5E0*ALOG(XMAX/XMIN))+2
00006860 NRHO=NRHO+1
00006870 X0=ALOG(XMAX)+0.25
00006880 NLYR=M
00006890 RHO1=TE0/SIG1
00006900 DO 10 I=1,M
00006910 DDC(I)=D(I)
00006920 SIG(I)=SIG1*RK(I)
00006930
00006940 INDEX=0
00006950
00006960 NEW=1
00006970 CMP=CMPLX(0E0,0E0)
00006980 DO 20 J=1,NRHO
00006990 I=NRHO1-J
00006990 XX=X0-0.2*I
00007000 AB2(I)=EXP(XX)
00007010 A2=EXP(2E0*XX)
00007020 IF(M.EQ.1) GO TO 19
00007030 CMP=ZLAGHT(XX,DCKERN,TOL,LW,NEW)/AB2(I)
00007040 RHOA(I)=RHO1+A2*REAL(CMP)
00007050
00007060 NEW=0
00007070 RETURN
00007080 END
00007090 REAL FUNCTION SPLINT(T,A,B,C,N,X,Y)
00007100 C---CUBIC SPLINE INTERPOLATION ROUTINE
00007110 ASSUMES PRIOR CALL TO SPLINT
00007120 DIMENSION A(1),B(1),C(1),X(1),Y(1)
00007130 IF(T.LT.X(1).OR.T.GT.X(N)) GO TO 9
00007140 N1=N-1
00007150 DO 1 I=1,N1
00007160 J=I
00007170 IF(T.LT.X(I+1)) GO TO 2
00007180 1 CONTINUE
00007190 9 SPLINT=0.0
00007200 IF(T.LT.X(1)) SPLINT=Y(1)

```



```

RETURN                                00007210
2 Z=T-X(J)                            00007220
  SPLINT=Y(J)+((C(J)*Z+B(J))*Z+A(J))*Z 00007230
RETURN                                00007240
END                                    00007250

COMPLEX FUNCTION DCKERN(AMBDA)        00007260
C                                     00007270
C RESISTIVITY KERNEL FUNCTION CALLED BY SCHSDG 00007280
C                                     00007290
C FOR INDEX = 0 RETURNS (KERNEL,PDIF W/R RHO(M)) 00007300
C   WHERE PDIF = PARTIAL DERIVATIVE 00007310
C FOR INDEX = I RETURNS (PDIF KERNEL W/R D(I), 00007320
C   PDIF KERNEL W/R RHO(I)) 00007330
C M > I > 0 00007340
C                                     00007350
C ALGORITHM FOR PARTIALS FROM: 00007360
C JOHANSEN, H.K., 1975, AN INTERACTIVE COMPUTER GRAPHICS- 00007370
C DISPLAY-TERMINAL SYSTEM FOR INTERPRETATION OF RESISTIVITY SOUNDING 00007380
C GEOPHYSICAL PROSPECTING, 23: 449-458. 00007390
C                                     00007400
C PROGRAMMED BY JKUAHIKAWA, USGS, DENVER, COLO, 1977. 00007410
C                                     00007420
COMMON/DCMDL/SIG(10),D(9),M,INDEX 00007430
REAL LM,LN,L2 00007440
PSIG=0E0 00007450
PD=0E0 00007460
L=M 00007470
LM=1E0 00007480
10 L1=L-1 00007490
  E=EXP_(-2E0*AMBDA*D(L1)) 00007500
  EN=(1E0-E)/(1E0+E) 00007510
  DENOM=SIG(L)+SIG(L1)*LM*EN 00007520
  IF(L.EQ.2) L2=LM 00007530
  IF(L.EQ.2) E1=EN 00007540
  LN=(SIG(L1)*LM+SIG(L)*EN)/DENOM 00007550
  IF(INDEX+1.LT.M.AND.L-2.EQ.INDEX) PSIG=(1E0-EN*LN)*LM/DENOM 00007560
  IF(INDEX-L1) 15,13,14 00007570
13 PSIG=((EN-LN)+(1E0-EN*LN)*SIG(L1)*PSIG)/DENOM 00007580
  PD=(1E0+EN)*(SIG(L)-LN*LM*SIG(L1))*2E0*AMBDA*E/(DENOM*(1E0+E)) 00007590
  GO TO 15 00007600
14 X=(1E0-LN*EN)*SIG(L1)/DENOM 00007610
  PSIG=PSIG*X 00007620
  PD=PD*X 00007630
15 L=L-1 00007640
  LM=LN 00007650
  IF(L.GT.1) GO TO 10 00007660
  IF(INDEX.GT.0) GO TO 20 00007670
  PL1=L2*(1E0-E1*LM)*AMBDA/DENOM 00007680
  LM=AMBDA*(LM-1E0) 00007690
  DCKERN=CMPLX(LM,PL1) 00007700
RETURN 00007710

```

```

00007720 20 DCKERN=CMPLX(AMRDA*PD,AMRDA*PSIG)
00007730 RETURN
00007740 END
00007750 COMPLEX FUNCTION CSPLN(RT)
00007760 C---CUBIC SPLINE INTERPOLATION ROUTINE
00007770 C RETURNS A COMPLEX RESULT (IMAG PART = 0)
00007780 C CALLED WITH TRNSI BY FTXXYZ
00007790 C ASSUMES PREVIOUS CALL TO SPLINI
00007800 C
00007810 C IM USED AS FLAG (SET BY CALLING ROUTINE):
00007820 C IM=1 - USE AX, BX, AND CX TO INTERPOLATE SPX
00007830 C IM=2 - USE AY, BY, AND CY TO INTERPOLATE SPY
00007840 C IM=3 - USE AZ, BZ, AND CZ TO INTERPOLATE SPZ
00007850 C
00007860 C COMMON/SPLXYZ/AX(100),BX(100),CX(100),AY(100),BY(100),CY(100),
00007870 C AZ(100),BZ(100),CZ(100),NB(100),NB(100),SPX(100),SPY(100),SPZ(100),IM
00007880 C
00007890 C NOTE THAT THE FUNCTION BEING SPLINED IS A FUNCTION
00007900 C OF B( ) = ALOG(B)
00007910 C
00007920 T=ALOG10(RT)
00007930 IF(T.LT,B(1),OR,T.GT,B(NB)) GO TO 20
00007940 N1=NB-1
00007950 DO 10 I=1,N1
00007960 J=I
00007970 IF(T.LT,B(I+1)) GO TO 30
00007980 CONTINUE
00007990 CS=0.0
00008000 IF(T.GE,B(1)) GO TO 25
00008010 IF(IM.EQ,1) CS=SPX(1)
00008020 IF(IM.EQ,2) CS=SPY(1)
00008030 IF(IM.EQ,3) CS=SPZ(1)
00008040 CSPLN=CMPLX(CS,0.0)
00008050 RETURN
00008060 30 Z=T-B(J)
00008070 IF(IM.EQ,1) CS=SPX(J)+((CX(J)*Z+BX(J))*Z+AX(J))*Z
00008080 IF(IM.EQ,2) CS=SPY(J)+((CY(J)*Z+BY(J))*Z+AY(J))*Z
00008090 IF(IM.EQ,3) CS=SPZ(J)+((CZ(J)*Z+BZ(J))*Z+AZ(J))*Z
00008100 CSPLN=CMPLX(CS,0.0)
00008110 RETURN
00008120 END
00008130 COMPLEX FUNCTION FINHX(B)
00008140 C---HX FIELD OF FINITE ELECTRIC WIRE
00008150 COMMON/FARM/ISTEP,A1,A2,A3,A4,BNYG,A5,A6
00008160 COMMON/FIN/R1,R2,R,L,SIG1,X,Y
00008170 COMMON/THICK/D(9)
00008180 COMMON/MODEL/K(10),DD(9),M
00008190 REAL L,K
00008200 DOUBLE PRECISION B8
00008210 COMPLEX F31,F32,I1K1

```

```
00008220 SB=SQRT(B)
00008230 FINHX=CMPLX(0.0,0.0)
00008240 IF(SB.GT.BNYG) RETURN
00008250 DEL=R/SB
00008260 IF(M.EQ.1) GO TO 12
00008270 B1=R1/DEL
00008280 B2=R2/DEL
00008290 M1=M-1
00008300 DO 1 I=1,M1
00008310 1 DD(I)=2.*D(I)/DEL
00008320 CALL FINF3(B1,B2,F31,F32)
00008330 FINHX=CMPLX(1./DEL,0.0)*(F31/R1-F32/R2)
00008340 12 B8=0.7071067811865475D0/DEL
00008350 FINHX=FINHX+(IK1(B8*R1)/R1**2-1IK1(B8*R2)/R2**2)
00008360 RETURN
00008370 END
00008380
00008390 COMPLEX FUNCTION FINHY(B)
00008400 C---HY FIELD OF FINITE ELECTRIC WIRE
00008410 C PASSES HX IN COMMON BLOCK /PASS/
00008420 EXTERNAL ZHY
00008430 COMMON/FIN/R1,R2,R/L,SIG1,X,Y
00008440 COMMON/CONST/DEL,DEL2,Z
00008450 COMMON/PASS/FINHX
00008460 COMMON/PARM/STEP,A1,A2,A3,A4,BNYG,M,A6
00008470 COMPLEX FINITE,F31,F32,FINHX,Z,1IK1,IK1,IK2
00008480 DOUBLE PRECISION B8
00008490 REAL L
00008500 SB=SQRT(B)
00008510 FINHX=CMPLX(0.0,0.0)
00008520 FINHY=CMPLX(0.0,0.0)
00008530 IF(SB.GT.BNYG) RETURN
00008540 DEL=R/SB
00008550 DEL2=DEL*DEL
00008560 FINHY=FINITE(ZHY,B)
00008570 IF(M.EQ.1) GO TO 10
00008580 B1=R1/DEL
00008590 B2=R2/DEL
00008600 CALL FINF3(B1,B2,F31,F32)
00008610 FINHY=FINHY-((X+L)*F31/R1-(X-L)*F32/R2)*CMPLX(1./DEL,0.0)
00008620 10 B8=0.7071067811865475D0/DEL
00008630 R12=R1*R1
00008640 R22=R2*R2
00008650 IK1=1IK1(B8*R1)/R12
00008660 IK2=1IK1(B8*R2)/R22
00008670 REMOVE PRIMARY FIELD IF STEP=1
00008680 IF(ISTEP.EQ.1) IK1=IK1-CMPLX(0.5/R12,0.0)
00008690 IF(ISTEP.EQ.1) IK2=IK2-CMPLX(0.5/R22,0.0)
00008700 FINHY=-((FINHY-((X+L)*IK1-(X-L)*IK2))*CMPLX(1./6.2831853,0.0)
00008710 IF(X*Y.EQ.0.0) GO TO 20
00008720 IF(M.EQ.1) GO TO 15
00008730
```

```

00008730 FINHX=(F31/R1-F32/R2)*CMPLX(1./DEL,0.0)
15 FINHX=-((FINHX+(IK1-IK2))*CMPLX(Y/6,2831853,0.0)
20 IF(ISTEP,ER,0) GO TO 30
FINHY=FINHY/B
FINHX=FINHX/B
30 RETURN
END
C---HZ FIELD OF FINITE ELECTRIC WIRE
COMPLEX FINHZ(B)
EXTERNAL ZHZ
COMMON/PARM/ISTEP,A1,A2,A3,A4,BNYQ,M,A6
COMMON/FIN/R1,R2,R,R1,SIG1,X,Y
SB=SQRT(B)
FINHZ=CMPLX(0,0,0.0)
IF(SB,GT,BNYQ) RETURN
FINHZ=FINITE(ZHZ,B)*CMPLX(Y/12,5663706,0.0)
IF(ISTEP,ER,1) FINHZ=FINHZ/B
RETURN
END
COMPLEX FUNCTION F3(G)
DATA ONE/(1,0,0,0)/
CALL RECURS(G,V1,F1)
C=G
F3=(V1*C*(ONE-F1))/((C+V1*F1)*(C+V1))
RETURN
END
COMPLEX FUNCTION F4(G)
DATA ONE/(1,0,0,0)/
CALL RECURS(G,V1,F1)
C=G
F4=(V1*C*(ONE-F1))/((C+V1*F1)*(C+V1))
RETURN
END
SUBROUTINE RECURS(G,V1,F1)
RECURS ALGORITHM FROM ANDERSON(1974)
COMPLEX C,V,M,V1,F1,EVD,ONE,I
REAL K(10),D(9)
COMMON/MODEL/K,D,M
DATA ONE/(1,0,0,0)/
F1=ONE
G2=G*G
VM=CSQRT(CMPLX(G2,2.0*K(M)))
IF(M,ER,1) GO TO 30

```

```
J=M-1
10 V1=CSQRT(CMPLX(G2,2.0*(J)))
EVD=-V1*(J)
EVD=CEXP(EVD)
20 C=(ONE-EVD)/(ONE+EVD)
T=VM*F1
F1=(T+V1*C)/(V1+T*C)
IF(J.EQ.1) GO TO 40
J=J-1
VM=V1
GO TO 10
30 V1=VM
40 RETURN
END
SUBROUTINE F1NF3(R1,R2,F31,F32)
COMMON/FINERF/TOL,TIT,N,NEW,MEV,ES,LM
COMPLEX F31,F32,ZHANKS,ES
EXTERNAL F3
F31=ZHANKS(1,B1,F3,TOL,LM,1)
IF(B1.EQ.B2) GO TO 10
F32=ZHANKS(1,B2,F3,TOL,LM,1)
RETURN
10 F32=F31
RETURN
RETURN
END
COMPLEX FUNCTION FINITE(FUNC,BFIN)
C---COMPUTE FINITE INTEGRAL OVER (-L,L) OF COMPLEX FIELD FUNCTION
C BY LAG-CONVOLUTION AND QUINTIC SPLINE INTERPOLATION PRIOR TO
C AUTOMATIC INTEGRATION BY SUBR, CANC4.
C /FINITE/ CALLS /FUNC/ (WHICH CALLS /ZLAGH1 OR /ZLAGH0/), /QUINT/, AND
C /CANC4/ (WHICH CALLS /FUNINT/ AND /GPOINT/).
PARAMETERS:
C EXTERNAL DECLARED COMPLEX FUNCTION DEFINING THE DIPOLE FIELD
C FUNC = FUNCTION WITH CALLING SEQ: FUNC(B,NEW,R), WHERE
C B = ANY IND. NO.,
C NEW = 1 FIRST TIME, 0 OTHERWISE (REF: ZLAGH1 OR ZLAGH0)
C R = B*DEL FOR ANY B, OR DEL (SKIN DEPTH),
C BFIN = FIXED IND. NO. FOR THE FINITE INTEGRAL (BFIN*GT.0).
C---COMMON PARAMETERS (INPUT) REQUIRED:
C HAKTOL = REQUESTED HANKEL TRANSFORM (ZLAG) TOLERANCE.
C USE HAKTOL.LE.1.E-6*EPS, EPS=ACTUAL HANKEL REL. ERROR.
C FINIOL = REQUESTED FINITE INTEGRAL (CANC4) TOLERANCE.
C USE FINIOL.LE.1.E-3*EFP, EFP=ACTUAL FINITE REL. ERROR.
C INTYPE = INTEGRATION TYPE FOR CANC4 (NORMALLY, INTYPE=2 OR 4).
C NFIN = 1 TO USE 1-PASS ZLAG, =2 FOR 2-PASS, ETC.
NOTE: NFIN*GT.1 TAKES /NFIN TIMES/, AS LONG TO RUN, BUT
00009210 J=M-1
00009220 10 V1=CSQRT(CMPLX(G2,2.0*(J)))
00009230 EVD=-V1*(J)
00009240 EVD=CEXP(EVD)
00009250 20 C=(ONE-EVD)/(ONE+EVD)
00009260 T=VM*F1
00009270 F1=(T+V1*C)/(V1+T*C)
00009280 IF(J.EQ.1) GO TO 40
00009290 J=J-1
00009300 VM=V1
00009310 GO TO 10
00009320 30 V1=VM
00009330 40 RETURN
00009340 END
00009350 SUBROUTINE F1NF3(R1,R2,F31,F32)
00009360 COMMON/FINERF/TOL,TIT,N,NEW,MEV,ES,LM
00009370 COMPLEX F31,F32,ZHANKS,ES
00009380 EXTERNAL F3
00009390 F31=ZHANKS(1,B1,F3,TOL,LM,1)
00009400 IF(B1.EQ.B2) GO TO 10
00009410 F32=ZHANKS(1,B2,F3,TOL,LM,1)
00009420 RETURN
00009430 10 F32=F31
00009440 RETURN
00009450 END
00009460 COMPLEX FUNCTION FINITE(FUNC,BFIN)
00009470 C---COMPUTE FINITE INTEGRAL OVER (-L,L) OF COMPLEX FIELD FUNCTION
00009480 C BY LAG-CONVOLUTION AND QUINTIC SPLINE INTERPOLATION PRIOR TO
00009490 C AUTOMATIC INTEGRATION BY SUBR, CANC4.
00009500 C /FINITE/ CALLS /FUNC/ (WHICH CALLS /ZLAGH1 OR /ZLAGH0/), /QUINT/, AND
00009510 C /CANC4/ (WHICH CALLS /FUNINT/ AND /GPOINT/).
00009520 PARAMETERS:
00009530 C EXTERNAL DECLARED COMPLEX FUNCTION DEFINING THE DIPOLE FIELD
00009540 C FUNC = FUNCTION WITH CALLING SEQ: FUNC(B,NEW,R), WHERE
00009550 C B = ANY IND. NO.,
00009560 C NEW = 1 FIRST TIME, 0 OTHERWISE (REF: ZLAGH1 OR ZLAGH0)
00009570 C R = B*DEL FOR ANY B, OR DEL (SKIN DEPTH),
00009580 C BFIN = FIXED IND. NO. FOR THE FINITE INTEGRAL (BFIN*GT.0).
00009590 C---COMMON PARAMETERS (INPUT) REQUIRED:
00009600 C HAKTOL = REQUESTED HANKEL TRANSFORM (ZLAG) TOLERANCE.
00009610 C USE HAKTOL.LE.1.E-6*EPS, EPS=ACTUAL HANKEL REL. ERROR.
00009620 C FINIOL = REQUESTED FINITE INTEGRAL (CANC4) TOLERANCE.
00009630 C USE FINIOL.LE.1.E-3*EFP, EFP=ACTUAL FINITE REL. ERROR.
00009640 C INTYPE = INTEGRATION TYPE FOR CANC4 (NORMALLY, INTYPE=2 OR 4).
00009650 C NFIN = 1 TO USE 1-PASS ZLAG, =2 FOR 2-PASS, ETC.
00009660 NOTE: NFIN*GT.1 TAKES /NFIN TIMES/, AS LONG TO RUN, BUT
00009670 00009680 00009690 00009700
```

```

C      WILL GIVE ADDITIONAL ACCURACY, IF NEEDED. 00009710
C MEV = MAX. FUNCT EVAL'S FOR CANC4 (NORMALLY MEV>300) 00009720
C R1 = MAX SPACING FROM WIRE END TO RECEIVER POINT (XX,YY) 00009730
C     = SQRT((XX+L)**2+YY*YY) 00009740
C R2 = MIN SPACING FROM WIRE END TO RECEIVER POINT (XX,YY) 00009750
C     = SQRT((XX-L)**2+YY*YY) 00009760
C R0 = SPACING FROM WIRE CENTER TO RECEIVER POINT (XX,YY) 00009770
C     = SQRT(XX*XX+YY*YY) 00009780
C D(9) = THICKNESS OF M-LAYERS IN MODEL. (METERS) 00009790
C K(10) = CONDUCTIVITY RATIO SIG(I)/SIG(1) 00009800
C        FOR I=1,M 00009810
C M = NO. LAYERS IN MODEL (M.GE.1.AND.M.LT.10) 00009820
C 00009830
COMMON/FINERR/HAKTOL,FINTOL,INTYPE,NFIN,NEV,MEV,ESUM,LW 00009840
COMMON/SPLN80/FDR(80),AR(80),BR(80),CR(80),DR(80),ER(80), 00009850
& FDI(80),AI(80),BI(80),CI(80),DI(80),EI(80),RLM1,DELRLM,NB 00009860
COMMON/FIN/R1,R2,R0,L,SIG1,X,Y 00009870
COMMON/THICK/D(9) 00009880
COMMON/MODEL/K(10),DD(9),M 00009890
COMMON/CONST/DEL,DEL2,Z2DEL3 00009900
REAL L,K 00009910
COMPLEX FUNC,ESUM,FD,FUNINT,CANC4,Z2DEL3 00009920
EXTERNAL FUNINT 00009930
C ISIZE IS THE MAXIMUM POSSIBLE NUMBER OF NODES IN QUINTIC SPLINE. 00009940
DATA ISIZE/80/ 00009950
DEL=R0/SQRT(BFIN) 00009960
DEL2=DEL*DEL 00009970
Z2DEL3=CMPLX(0.0,2./((DEL2*DEL))) 00009980
M1=M-1 00009990
DO 1 I=1,M1 00010000
1 DD(I)=2.*D(I)/DEL 00010010
BMAX=R1/DEL 00010020
BMIN=R2/DEL 00010030
IF(X.LE.L) BMIN=Y/DEL 00010040
NB=AIN(5.*ALOG(BMAX/BMIN))+2 00010050
NB=MAX(NB,3) 00010060
X0=ALOG(BMIN)+NB*0.2 00010070
NB=NB+3 00010080
NRMAX=ISIZE/NB 00010090
IF(NFIN.LE.NRMAX) GO TO 3 00010100
IF(NRMAX.GT.0.0) GO TO 2 00010110
PRINT, 'ERROR IN FINITE: INSUFFICIENT SPLINE NODES' 00010120
STOP 00010130
2 NFIN=NRMAX 00010140
PRINT, 'ERROR IN FINITE: NFIN TOO LARGE, RESET TO ',NFIN 00010150
3 DELRLM=.2/FLOAT(NFIN) 00010160
X0=X0-DELRLM 00010170
DO 5 ITIME=1,NFIN 00010180
NEW=1 00010190
X0=X0+DELRLM 00010200
DO 5 J=1,NB 00010210
I=(NB+1)-J 00010220

```

```

I=NFIN*(I-1)+ITIME
XX=X0-0.2*KJ
BM=EXP(XX)
RM=BM*DEL
IF(I,EQ,1) RLM1=ALOG(RM)
FD=FUNC(BM,NEW,FM)
FDR(I)=REAL(FD)
FDI(I)=AIMAG(FD)
5 NEW=0
NB=NFIN*NB
CALL QUINT(NB,FDR,AR,BR,CR,DR,ER)
CALL QUINT(NB,FDI,AI,BI,CI,DI,EI)
IF(X,LT,L) GO TO 8
FINITE=CANC4(X-L,X+L,FINTOL,NEV,INTYPE,FUNINT,MEV,ESUM)
GO TO 10
8 FINITE=2.*CANC4(0.,ABS(X-L),FINTOL,NEV,INTYPE,FUNINT,MEV,ESUM)
IF(X,EQ,0.0) GO TO 10
FINITE=FINITE+CANC4(ABS(X-L),X+L,FINTOL,NEV,INTYPE,FUNINT,MEV,ESUM)
& ESUM)
10 RETURN
END

COMPLEX FUNCTION ZHY(B,NEW,R)
COMPLEX ZLAGH0,Z,I1K1,IKDIF
DOUBLE PRECISION B8
EXTERNAL F4
COMMON/PARM/ISTEP,A1,A2,A3,A4,A5,M,TOL
COMMON/CONST/DEL,DEL2,Z
ZHY=CMPLX(0.0,0.0)
IF(M,EQ,1) GO TO 2
ZHY=ZLAGH0(ALOG(B),F4,TOL,LW,NEW)/B
2 B8=0.7071067811865400*B
CALL IKS(B8,I1K1,IKDIF)
IF(ISTEP,EQ,0) GO TO 5
  REMOVE THE PRIMARY FIELD IF ISTEP=1
  I1K1=I1K1-CMPLX(0.5,0.0)
  IKDIF=IKDIF-CMPLX(1.0,0.0)
5 ZHY=ZHY*CMPLX(1./DEL2,0.0)-((IKDIF-2.*I1K1)/R)**2
RETURN
END

COMPLEX FUNCTION ZHZ(B,NEW,R)
COMPLEX ZLAGH1,HAF,ONE,Z
EXTERNAL F4
COMMON/PARM/ISTEP,A1,A2,A3,A4,A5,M,TOL
COMMON/CONST/DEL,DEL2,Z
DATA ONE/(1.0,0.0)/
ZHZ=CMPLX(0.0,0.0)
IF(M,EQ,1) GO TO 2
ZHZ=ZLAGH1(ALOG(B),F4,TOL,LW,NEW)*CMPLX(2./(B**R*DEL2),0.0)
2 HAF=ONE-(ONE+CMPLX(B,B)+CMPLX(0.0,2.*B*B/3.))*CEXF(CMPLX(-B,-B))
ZHZ=ZHZ-HAF*CMPLX(0.0,3./(R**3*B*B))

```

```

00010230
00010240
00010250
00010260
00010270
00010280
00010290
00010300
00010310
00010320
00010330
00010340
00010350
00010360
00010370
00010380
00010390
00010400
00010410
00010420
00010430

00010440
00010450
00010460
00010470
00010480
00010490
00010500
00010510
00010520
00010530
00010540
00010550
00010560
00010570
00010580
00010590
00010600
00010610

00010620
00010630
00010640
00010650
00010660
00010670
00010680
00010690
00010700
00010710
00010720

```





```

SUBROUTINE SPLINE(M,H,X,Y,A,B,C,IT,D,F,S)
C---ONE DIMENSIONAL CUBIC SPLINE COEFFICIENT DETERMINATION.
C
C BY W.L.ANDERSON, U.S. GEOLOGICAL SURVEY, DENVER, COLORADO
C
C PARMS--- M= NUMBER OF DATA POINTS, GT. 2
C           H= EQUAL INTERVAL OPTION WHEN H.GT.0. (USE DUMMY X HERE),
C           UNEQUAL INTERVALS IF H=0. (X REQUIRED STORAGE)
C           X= INDEF.VAR WHEN H=0. (DIM,GE,M).
C           Y= DEPENDENT VARIABLE (DIM,GE,M).
C           A,B,C=COEFF.ARRAYS (EACH DIM,GE,M)
C           RESULTS ARE RETURNED IN 1ST(M-1) ELEMENTS OF A,B,C.
C           ALSO USED AS WORK ARRAYS DURING EXECUTION.
C           IT= TYPE OF BOUNDARY CONDITION SUPPLIED IN D ARRAY. USE
C           IT=1 IF 1ST DERIVATIVES GIVEN AT END POINTS, OR
C           IT=0 IF 2ND DERIVATIVES GIVEN AT END POINTS.
C           D= BOUNDARY ARRAY (DIM 2) AT POINT 1 AND M RESPECTIVELY.
C           F,S= WORK ARRAYS (EACH DIM=M).
C---ERROR RETURN WITH M=-(ABS(M)) IF ANY PARAM OUT OF RANGE.
C THE RESULTING CUBIC SPLINE IS OF THE FORM:
C           Y=A(I)+A(I)*(X-X(I))+B(I)*(X-X(I))**2+C(I)*(X-X(I))**3
C           FOR I=1,2,...,M-1
C
C REAL*4 X(1),Y(1),A(1),B(1),C(1),D(2),F(1),S(1),MUL
C           IF(IT.LT.0,OR,IT.GT.1,OR,H.LT.0,OR,M.LT.3) GO TO 999
C           N=M-1
C           IF(IT.EQ.0) GO TO 20
C---1ST DERIVATIVE BOUNDARIES GIVEN
C           NE=N-1
C           IF(H) 999,11,1
C---EQUAL SPACING H.GT.0, AND IT=1
C           1 HH=3.0/H
C           DO 2 I=1,NE
C           B(I)=4.0
C           C(I)=1.0
C           A(I)=1.0
C           2 P(I)=HH*(Y(I+2)-Y(I))
C           F(1)=F(1)-D(1)
C           P(NE)=F(NE)-D(2)
C---SOLUTION OF TRIANGULAR MATRIX EQ. OF ORDER NE
C           3 C(1)=C(1)/B(1)
C           P(1)=F(1)/B(1)
C           DO 4 I=2,NE
C           MUL=1.0/(B(I)-A(I)*C(I-1))
C           C(I)=MUL*C(I)
C           F(I)=MUL*(F(I)-A(I)*F(I-1))
C---OBTAIN SPLINE COEFFICIENTS
C           A(NE+1)=F(NE)
C           I=NE-1
C           5 A(I+1)=F(I)-C(I)*A(I+1+1)
00011230
00011230 SUBROUTINE SPLINE COEFFICIENT DETERMINATION.
00011240
00011250 BY W.L.ANDERSON, U.S. GEOLOGICAL SURVEY, DENVER, COLORADO
00011260
00011270 M= NUMBER OF DATA POINTS, GT. 2
00011280 H= EQUAL INTERVAL OPTION WHEN H.GT.0. (USE DUMMY X HERE),
00011290 UNEQUAL INTERVALS IF H=0. (X REQUIRED STORAGE)
00011300 X= INDEF.VAR WHEN H=0. (DIM,GE,M).
00011310 Y= DEPENDENT VARIABLE (DIM,GE,M).
00011320 A,B,C=COEFF.ARRAYS (EACH DIM,GE,M)
00011330 RESULTS ARE RETURNED IN 1ST(M-1) ELEMENTS OF A,B,C.
00011340 ALSO USED AS WORK ARRAYS DURING EXECUTION.
00011350 IT= TYPE OF BOUNDARY CONDITION SUPPLIED IN D ARRAY. USE
00011360 IT=1 IF 1ST DERIVATIVES GIVEN AT END POINTS, OR
00011370 IT=0 IF 2ND DERIVATIVES GIVEN AT END POINTS.
00011380 D= BOUNDARY ARRAY (DIM 2) AT POINT 1 AND M RESPECTIVELY.
00011390 F,S= WORK ARRAYS (EACH DIM=M).
00011400 ---ERROR RETURN WITH M=-(ABS(M)) IF ANY PARAM OUT OF RANGE.
00011410 THE RESULTING CUBIC SPLINE IS OF THE FORM:
00011420 Y=A(I)+A(I)*(X-X(I))+B(I)*(X-X(I))**2+C(I)*(X-X(I))**3
00011430 FOR I=1,2,...,M-1
00011440
00011450 REAL*4 X(1),Y(1),A(1),B(1),C(1),D(2),F(1),S(1),MUL
00011460 IF(IT.LT.0,OR,IT.GT.1,OR,H.LT.0,OR,M.LT.3) GO TO 999
00011480 N=M-1
00011490 IF(IT.EQ.0) GO TO 20
00011500 ---1ST DERIVATIVE BOUNDARIES GIVEN
00011510 NE=N-1
00011520 IF(H) 999,11,1
00011530 ---EQUAL SPACING H.GT.0, AND IT=1
00011540 1 HH=3.0/H
00011550 DO 2 I=1,NE
00011560 B(I)=4.0
00011570 C(I)=1.0
00011580 A(I)=1.0
00011590 2 P(I)=HH*(Y(I+2)-Y(I))
00011600 F(1)=F(1)-D(1)
00011610 P(NE)=F(NE)-D(2)
00011620 ---SOLUTION OF TRIANGULAR MATRIX EQ. OF ORDER NE
00011630 3 C(1)=C(1)/B(1)
00011640 P(1)=F(1)/B(1)
00011650 DO 4 I=2,NE
00011660 MUL=1.0/(B(I)-A(I)*C(I-1))
00011670 C(I)=MUL*C(I)
00011680 F(I)=MUL*(F(I)-A(I)*F(I-1))
00011690 ---OBTAIN SPLINE COEFFICIENTS
00011700 A(NE+1)=F(NE)
00011710 I=NE-1
00011720 5 A(I+1)=F(I)-C(I)*A(I+1+1)
00011730

```

I=I-1	00011740
IF(I.GE.1) GO TO 5	00011750
IF(IT.EQ.0) GO TO 6	00011760
A(1)=D(1)	00011770
A(M)=D(2)	00011780
6 IF(H.EQ.0.) GO TO 14	00011790
HH=1.0/H	00011800
DO 7 I=1,N	00011810
MUL=HH*(Y(I+1)-Y(I))	00011820
B(I)=HH*(3.0*MUL-(A(I+1)+2.0*A(I)))	00011830
7 C(I)=HH*HH*(-2.0*MUL+A(I+1)+A(I))	00011840
RETURN	00011850
C--UNEQUAL SPACING H=0., AND IT=1	00011860
11 DO 12 I=1,N	00011870
12 S(I+1)=X(I+1)-X(I)	00011880
DO 13 I=1,NE	00011890
B(I)=2.0*(S(I+1)+S(I+2))	00011900
C(I)=S(I+1)	00011910
A(I)=S(I+2)	00011920
13 P(I)=3.0*(S(I+1)**2*(Y(I+2)-Y(I+1))+S(I+2)**2*(Y(I+1)-Y(I)))/	00011930
\$ (S(I+1)*S(I+2))	00011940
P(1)=P(1)-S(3)*D(1)	00011950
P(NE)=P(NE)-S(N)*D(2)	00011960
GO TO 3	00011970
14 DO 15 I=1,N	00011980
HH=1.0/S(I+1)	00011990
MUL=(Y(I+1)-Y(I))*HH**2	00012000
B(I)=3.0*MUL-(A(I+1)+2.0*A(I))*HH	00012010
15 C(I)=-2.0*MUL*HH+(A(I+1)+A(I))*HH**2	00012020
RETURN	00012030
C--2ND DERIVATIVE BOUNDARIES GIVEN	00012040
20 NE=N+1	00012050
IF(H) 999,31,21	00012060
C--EQUAL SPACING H .GT. 0 AND IT=0	00012070
21 HH=3.0/H	00012080
DO 22 I=2,N	00012090
B(I)=4.0	00012100
C(I)=1.0	00012110
A(I)=1.0	00012120
22 F(I)=HH*(Y(I+1)-Y(I-1))	00012130
B(1)=2.0	00012140
B(NE)=2.0	00012150
C(1)=1.0	00012160
C(NE)=1.0	00012170
A(NE)=1.0	00012180
F(1)=HH*(Y(2)-Y(1))-0.5*H*D(1)	00012190
F(NE)=HH*(Y(M)-Y(N))+0.5*H*D(2)	00012200
GO TO 3	00012210
C--UNEQUAL SPACING H=0 AND IT=0	00012220
31 DO 32 I=1,N	00012230
32 S(I+1)=X(I+1)-X(I)	00012240
N1=N-1	00012250

```

DO 33 I=1,N1                                00012260
B(I+1)=2.0*(S(I+1)+S(I+2))                  00012270
C(I+1)=S(I+1)                               00012280
A(I+1)=S(I+2)                               00012290
33 P(I+1)=3.0*(S(I+1)**2*(Y(I+2)-Y(I+1))+S(I+2)**2*(Y(I+1)-Y(I)))/ 00012300
*      (S(I+1)*S(I+2))                      00012310
B(1)=2.0                                    00012320
B(NE)=2.0                                    00012330
C(1)=1.0                                    00012340
C(NE)=1.0                                    00012350
A(NE)=1.0                                    00012360
P(1)=3.0*(Y(2)-Y(1))/S(2)-0.5*S(2)*D(1)    00012370
P(NE)=3.0*(Y(M)-Y(N))/S(M)+0.5*S(M)*D(2)   00012380
GO TO 3                                       00012390
999 M=-IABS(M)                               00012400
RETURN                                       00012410
END                                           00012420

```

```

      COMPLEX FUNCTION ZLAGHO(X,FUN,TOL,L,NEW) 00012430
C---*** A SPECIAL LAGGED* CONVOLUTION METHOD TO COMPUTE THE 00012440
C INTEGRAL FROM 0 TO INFINITY OF 'FUN(G)*JO(G*B)*DG' DEFINED AS THE 00012450
C COMPLEX HANKEL TRANSFORM OF ORDER 0 AND ARGUMENT X(=ALOG(B)) 00012460
C BY CONVOLUTION FILTERING WITH COMPLEX FUNCTION 'FUN'---AND 00012470
C USING A VARIABLE CUT-OFF METHOD WITH EXTENDED FILTER TAILS... 00012480
C 00012490
C---REF: ANDERSON, W.L., 1975, NTIS REPT, PB-242-800, 00012500
C 00012510
C---PARAMETERS: 00012520
C 00012530
C * X = REAL ARGUMENT(=ALOG(B) AT CALL) OF THE HANKEL TRANSFORM 00012540
C 'ZLAGHO' IS USEFUL ONLY WHEN X=(LAST X)-.20 *** I.E., 00012550
C SPACED SAME AS FILTER USED---IF THIS IS NOT CONVENIENT, 00012560
C THEN SUBPROGRAM 'ZHANKO' IS ADVISED FOR GENERAL USE. 00012570
C (ALSO SEE PARM 'NEW' & NOTES (2)-(4) BELOW). 00012580
C FUN(G)= EXTERNAL DECLARED COMPLEX FUNCTION NAME (USER SUPPLIED) 00012590
C OF A REAL ARGUMENT G. 00012600
C NOTE: IF PARMS OTHER THAN G ARE REQUIRED, USE COMMON IN 00012610
C CALLING PROGRAM AND IN SUBPROGRAM FUN. 00012620
C THE COMPLEX FUNCTION FUN SHOULD BE A MONOTONE 00012630
C DECREASING FUNCTION AS THE ARGUMENT G BECOMES LARGE...00012640
C FOR REAL-ONLY FUNCTIONS, SUBPROGRAM 'RLAGHO' IS ADVISED;00012650
C HOWEVER, TWO REAL-FUNCTIONS F1(G),F2(G) MAY BE 00012660
C INTEGRATED IN PARALLEL BY WRITING FUN=CMPLX(F1(G),F2(G))00012670
C TOL= REAL TOLERANCE EXCEPTED AT CONVOLVED TAILS---I.E., 00012680
C IF FILTER*FUN<TOL*MAX, THEN REST OF TAIL IS TRUNCATED,00012690
C THIS IS DONE AT BOTH ENDS OF FILTER. TYPICALLY, 00012700
C TOL <= .0001 IS USUALLY OK---BUT THIS DEPENDS ON 00012710
C THE FUNCTION FUN AND PARAMETER X...IN GENERAL, 00012720
C A 'SMALLER TOL' WILL USUALLY RESULT IN 'MORE ACCURACY' 00012730
C BUT WITH 'MORE WEIGHTS' BEING USED. TOL IS NOT DIRECTLY00012740
C RELATED TO TRUNCATION ERROR, BUT GENERALLY SERVES AS AN 00012750
C APPROXIMATION INDICATOR... FOR VERY LARGE OR SMALL B, 00012760

```

```

C      ONE SHOULD USE A SMALLER TOL THAN RECOMMENDED ABOVE... 00012770
C      L=      RESULTING NO. FILTER WTS. USED IN THE VARIABLE 00012780
C              CONVOLUTION (L DEPENDS ON TOL AND FUN). 00012790
C      MIN,L=20 AND MAX,L=193--WHICH COULD 00012800
C              OCCUR IF TOL IS VERY SMALL AND/OR FUN NOT DECREASING 00012810
C              VERY FAST... 00012820
C      * NEW= 1 IS NECESSARY 1ST TIME OR BRAND NEW X. 00012830
C              0 FOR ALL SUBSEQUENT CALLS WHERE X=(LAST X)-0.20 00012840
C              IS ASSUMED INTERNALLY BY THIS ROUTINE. 00012850
C              NOTE: IF THIS IS NOT TRUE, ROUTINE WILL 00012860
C              STILL ASSUME X=(LAST X)-0.20 ANYWAY... 00012870
C              IT IS THE USERS RESPONSIBILITY TO NORMALIZE 00012880
C              BY CORRECT B=EXP(X) OUTSIDE OF CALL (SEE USAGE BELOW). 00012890
C      THE LAGGED CONVOLUTION METHOD PICKS UP SIGNIFICANT 00012900
C      TIME IMPROVEMENTS WHEN THE KERNEL IS NOT A 00012910
C      SIMPLE ELEMENTARY FUNCTION...DUE TO INTERNALLY SAVING 00012920
C      ALL KERNEL FUNCTION EVALUATIONS WHEN NEW=1... 00012930
C      THEN WHEN NEW=0, ALL PREVIOUSLY CALCULATED 00012940
C      KERNELS WILL BE USED IN THE LAGGED CONVOLUTION 00012950
C      WHERE POSSIBLE, ONLY ADDING NEW KERNEL EVALUATIONS 00012960
C      WHEN NEEDED (DEPENDS ON PARMS TOL AND FUN) 00012970
C      00012980
C---THE RESULTING COMPLEX CONVOLUTION SUM IS GIVEN IN ZLAGHO; THE HANKEL 00012990
C TRANSFORM IS THEN ZLAGHO/B WHICH IS TO BE COMPUTED AFTER EXIT FROM 00013000
C THIS ROUTINE.... WHERE B=EXP(X), X=ARGUMENT USED IN CALL... 00013010
C 00013020
C---USAGE--- 'ZLAGHO' IS CALLED AS FOLLOWS: 00013030
C 00013040
C      ... 00013050
C      COMPLEX Z,ZLAGHO,ZF 00013060
C      EXTERNAL ZF 00013070
C      ... 00013080
C      Z=ZLAGHO(ALOG(B),ZF,TOL,L,NEW)/B 00013090
C      ... 00013100
C      END 00013110
C      COMPLEX FUNCTION ZF(G) 00013120
C      ...USER SUPPLIED CODE... 00013130
C      END 00013140
C 00013150
C---NOTES: 00013160
C      (1). EXP-UNDERFLOW'S MAY OCCUR IN EXECUTING THE SUBPROGRAM 00013170
C      BELOW; HOWEVER, THIS IS OK PROVIDED THE MACHINE SYSTEM SETS 00013180
C      ANY & ALL EXP-UNDERFLOW'S TO 0.0..... 00013190
C      (2). AS AN AID TO UNDERSTANDING & USING THE LAGGED CONVOLUTION 00013200
C      METHOD, LET BMAX>=BMIN>0 BE GIVEN. THEN IT CAN BE SHOWN 00013210
C      THAT THE ACTUAL NUMBER OF B'S IS NB=AINT(5.*ALOG(BMAX/BMIN))+1, 00013220
C      PROVIDED BMAX/BMIN>=1. THE USER MAY THEN ASSUME AN 'ADJUSTED' 00013230
C      BMINA=BMAX*EXP(-.2*(NB-1)). THE METHOD GENERATES THE DECREASING 00013240
C      ARGUMENTS SPACED AS X=ALOG(BMAX),X-.2,X-.2*2,...,ALOG(BMINA). 00013250
C      FOR EXAMPLE, ONE MAY CONTROL THIS WITH THE CODE: 00013260
C      ... 00013270
C      NB=AINT(5.*ALOG(BMAX/BMIN))+1 00013280
C      NB1=NB+1
  
```

```

C      X0=ALOG(BMAX)+.2      00013290
C      NEW=1                00013300
C      DO 1 J=1,NB         00013310
C      I=NB1-J            00013320
C      X=X0-.2*J          00013330
C      ARG(I)=EXP(X)      00013340
C      Z(I)=ZLAGH0(X,ZF,TOL,L,NEW)/ARG(I) 00013350
C      1 NEW=0            00013360
C      ...                00013370
C      (3). IF RESULTS ARE STORED IN ARRAYS ARG(I),Z(I),I=1,NB FOR 00013380
C      ARG IN (BMINA,BMAX), THEN THESE ARRAYS MAY BE USED, FOR EXAMPLE, 00013390
C      TO SPLINE-INTERPOLATE AT A DIFFERENT (LARGER OR SMALLER) 00013400
C      SPACING THAN USED IN THE LAGGED CONVOLUTION METHOD. 00013410
C      (4). IF A DIFFERENT RANGE OF B IS DESIRED, THEN ONE MAY 00013420
C      ALWAYS RESTART THE ABOVE PROCEDURE IN (2) WITH A NEW 00013430
C      BMAX,BMIN AND BY SETTING NEW=1,... 00013440
C      (5). ABSCISSA CORRESPONDING TO WEIGHT IS GENERATED TO SAVE STORAGE 00013450
C      00013460
C      00013470
C      COMPLEX FUN,C,CMAX,SAVE 00013480
C      DIMENSION KEY(193),SAVE(193),T(2),TMAX(2) 00013490
C      DIMENSION YT(193),Y1(76),Y2(76),Y3(41) 00013500
C      EQUIVALENCE (C,T(1)),(CMAX,TMAX(1)) 00013510
C      EQUIVALENCE (YT(1),Y1(1)),(YT(77),Y2(1)),(YT(153),Y3(1)) 00013520
C---JO-EXTENDED FILTER WEIGHT ARRAYS: 00013530
C      DATA Y1/ 00013540
C      1 5.8565723E-08, 7.1143477E-11,-7.8395565E-11, 8.7489547E-11, 00013550
C      2-8.9007811E-11, 9.8790055E-11,-9.8675347E-11, 1.1118797E-10, 00013560
C      3-1.0893474E-10, 1.2543400E-10,-1.1979399E-10, 1.4200767E-10, 00013570
C      4-1.3106341E-10, 1.6153229E-10,-1.4238602E-10, 1.8486236E-10, 00013580
C      5-1.5315381E-10, 2.1319755E-10,-1.6238115E-10, 2.4824144E-10, 00013590
C      6-1.6850378E-10, 2.9243813E-10,-1.6909302E-10, 3.4934366E-10, 00013600
C      7-1.6043759E-10, 4.2417082E-10,-1.3690001E-10, 5.2458440E-10, 00013610
C      8-8.9946096E-11, 6.6188220E-10,-6.6964033E-12, 8.5276151E-10, 00013620
C      9 1.3222770E-10, 1.1219600E-09, 3.5591442E-10, 1.5061956E-09, 00013630
C      1 7.0795382E-10, 2.0600379E-09, 1.2535947E-09, 2.8646623E-09, 00013640
C      2 2.0904225E-09, 4.0409101E-09, 3.3642886E-09, 5.7687700E-09, 00013650
C      3 5.2930786E-09, 8.3164338E-09, 8.2021809E-09, 1.2083635E-08, 00013660
C      4 1.2577400E-08, 1.7666303E-08, 1.9143895E-08, 2.5953011E-08, 00013670
C      5 2.8983953E-08, 3.8268851E-08, 4.3712685E-08, 5.6590075E-08, 00013680
C      6 6.5740136E-08, 8.3864288E-08, 9.8662323E-08, 1.2448811E-07, 00013690
C      7 1.4784461E-07, 1.8501974E-07, 2.2129198E-07, 2.7524203E-07, 00013700
C      8 3.3094739E-07, 4.0974828E-07, 4.9462868E-07, 6.1030809E-07, 00013710
C      9 7.3891802E-07, 9.0939667E-07, 1.1034727E-06, 1.3554600E-06, 00013720
C      1 1.6474556E-06, 2.0207696E-06, 2.4591294E-06, 3.0131400E-06/ 00013730
C      DATA Y2/ 00013740
C      1 3.6701680E-06, 4.4934101E-06, 5.4770076E-06, 6.7015208E-06, 00013750
C      2 8.1726989E-06, 9.9954201E-06, 1.2194425E-05, 1.4909101E-05, 00013760
C      3 1.8194388E-05, 2.2239184E-05, 2.7145562E-05, 3.3174088E-05, 00013770
C      4 4.0499452E-05, 4.9486730E-05, 6.0421440E-05, 7.3822001E-05, 00013780
C      5 9.0141902E-05, 1.1012552E-04, 1.3448017E-04, 1.6428337E-04, 00013790
C      6 2.0062570E-04, 2.4507680E-04, 2.9930366E-04, 3.6560582E-04, 00013800

```

```

7 4.4651421E-04, 5.4541300E-04, 6.6612648E-04, 8.1365181E-04, 00013810
8 9.9374786E-04, 1.2138120E-03, 1.4824945E-03, 1.8107657E-03, 00013820
9 2.2115938E-03, 2.7012675E-03, 3.2991969E-03, 4.0295817E-03, 00013830
1 4.9214244E-03, 6.0106700E-03, 7.3405529E-03, 8.9643708E-03, 00013840
2 1.0946310E-02, 1.3365017E-02, 1.6314985E-02, 1.9910907E-02, 00013850
3 2.4289325E-02, 2.9612896E-02, 3.6070402E-02, 4.3876936E-02, 00013860
4 5.3264829E-02, 6.4465091E-02, 7.7664144E-02, 9.2918324E-02, 00013870
5 1.1000121E-01, 1.2811102E-01, 1.4543025E-01, 1.5832248E-01, 00013880
6 1.6049224E-01, 1.4170064E-01, 8.8788108E-02, -1.1330934E-02, 00013890
7 -1.5331864E-01, -2.9094670E-01, -2.9084655E-01, -2.9708834E-02, 00013900
8 3.9009601E-01, 1.7999785E-01, -4.1858139E-01, 1.5317216E-01, 00013910
9 6.5184953E-02, -1.0751806E-01, 7.8429567E-02, -4.6019124E-02, 00013920
1 2.5309571E-02, -1.3904823E-02, 7.8187120E-03, -4.5190369E-03, 00013930
  DATA Y3/ 00013940
1 2.6729062E-03, -1.6073718E-03, 9.7715622E-04, -5.9804407E-04, 00013950
2 3.6749320E-04, -2.2635296E-04, 1.3960805E-04, -8.6172618E-05, 00013960
3 5.3212947E-05, -3.2867888E-05, 2.0304203E-05, -1.2543926E-05, 00013970
4 7.7499633E-06, -4.7882430E-06, 2.9584108E-06, -1.8278645E-06, 00013980
5 1.1293571E-06, -6.9778174E-07, 4.3113019E-07, -2.6637753E-07, 00013990
6 1.6458373E-07, -1.0168954E-07, 6.2829807E-08, -3.8819969E-08, 00014000
7 2.3985272E-08, -1.4819520E-08, 9.1563774E-09, -5.6573541E-09, 00014010
8 3.4954514E-09, -2.1597005E-09, 1.3343946E-09, -8.2447148E-10, 00014020
9 5.0941033E-10, -3.1474631E-10, 1.9447072E-10, -1.2015685E-10, 00014030
1 7.4241055E-11, -4.5871468E-11, 2.8343095E-11, -1.7513137E-11, 00014040
2 6.9049613E-12/ 00014050
C---$$$ENDATA 00014060
C 00014070
  IF(NEW) 10,30,10 00014080
10 LAG=-1 00014090
  X0=-X-26.30455704 00014100
  DO 20 IR=1,193 00014110
20 KEY(IR)=0 00014120
30 LAG=LAG+1 00014130
  ZLAGH0=(0.0,0.0) 00014140
  CMAX=(0.0,0.0) 00014150
  L=0 00014160
  ASSIGN 110 TO M 00014170
  I=129 00014180
  GO TO 200 00014190
110 TMAX(1)=AMAX1(ABS(T(1)),TMAX(1)) 00014200
  TMAX(2)=AMAX1(ABS(T(2)),TMAX(2)) 00014210
  I=I+1 00014220
  IF(I.LE.146) GO TO 200 00014230
  IF(TMAX(1).EQ.0.0.AND.TMAX(2).EQ.0.0) GO TO 150 00014240
  CMAX=TOL*CMAX 00014250
  ASSIGN 120 TO M 00014260
  I=128 00014270
  GO TO 200 00014280
120 IF(ABS(T(1)).LE.TMAX(1).AND.ABS(T(2)).LE.TMAX(2)) GO TO 130 00014290
  I=I-1 00014300
  IF(I.GT.0) GO TO 200 00014310
130 ASSIGN 140 TO M 00014320

```

```

    I=147                                00014330
    GO TO 200                             00014340
140  IF(ABS(T(1)).LE.TMAX(1).AND.ABS(T(2)).LE.TMAX(2)) GO TO 190 00014350
    I=I+1                                 00014360
    IF(I.LE.193) GO TO 200                00014370
    GO TO 190                             00014380
150  ASSIGN 160 TO M                      00014390
    I=1                                    00014400
    GO TO 200                             00014410
160  IF(T(1).EQ.0.0.AND.T(2).EQ.0.0) GO TO 170 00014420
    I=I+1                                 00014430
    IF(I.LE.128) GO TO 200                00014440
170  ASSIGN 180 TO M                      00014450
    I=193                                  00014460
    GO TO 200                             00014470
180  IF(T(1).EQ.0.0.AND.T(2).EQ.0.0) GO TO 190 00014480
    I=I-1                                  00014490
    IF(I.GE.147) GO TO 200                00014500
190  RETURN                               00014510
C--STORE/RETRIEVE ROUTINE (DONE INTERNALLY TO SAVE CALL'S) 00014520
200  LOOK=I+LAG                           00014530
    IQ=LOOK/194                           00014540
    IR=MOD(LOOK,194)                       00014550
    IF(IR.EQ.0) IR=1                       00014560
    IROLL=IQ*193                           00014570
    IF(KEY(IR).LE.IROLL) GO TO 220         00014580
210  C=SAVE(IR)*YT(I)                     00014590
    ZLAGH0=ZLAGH0+C                        00014600
    L=L+1                                   00014610
    GO TO M,(110,120,140,160,180)         00014620
220  KEY(IR)=IROLL+IR                     00014630
    SAVE(IR)=FUN(EXP(X0+FLOAT(LOOK)*.20)) 00014640
    GO TO 210                              00014650
    END                                    00014660

```

COMPLEX FUNCTION ZLAGH1(X,FUN,TOL,L,NEW)

```

C--*** A SPECIAL LAGGED* CONVOLUTION METHOD TO COMPUTE THE 00014670
C INTEGRAL FROM 0 TO INFINITY OF 'FUN(G)*J1(G*B)*DG' DEFINED AS THE 00014680
C COMPLEX HANKEL TRANSFORM OF ORDER 1 AND ARGUMENT X(=ALOG(B)) 00014690
C BY CONVOLUTION FILTERING WITH COMPLEX FUNCTION 'FUN'--AND 00014700
C USING A VARIABLE CUT-OFF METHOD WITH EXTENDED FILTER TAILS... 00014710
C 00014720
C 00014730
C--REF: ANDERSON, W.L., 1975, NTIS REPT. PB-242-800. 00014740
C 00014750
C--PARAMETERS: 00014760
C 00014770
C * X = REAL ARGUMENT(=ALOG(B) AT CALL) OF THE HANKEL TRANSFORM 00014780
C 'ZLAGH1' IS USEFUL ONLY WHEN X=(LAST X)-.20 *** I.E., 00014790
C SPACED SAME AS FILTER USED--IF THIS IS NOT CONVENIENT, 00014800
C THEN SUBPROGRAM 'ZHANK1' IS ADVISED FOR GENERAL USE. 00014810
C (ALSO SEE PARM 'NEW' & NOTES (2)-(3) BELOW). 00014820
C FUN(G)= EXTERNAL DECLARED COMPLEX FUNCTION NAME (USER SUPPLIED) 00014830

```

```

C      OF A REAL ARGUMENT G.                                00014840
C      NOTE: IF FARMS OTHER THAN G ARE REQUIRED, USE COMMON IN 00014850
C      CALLING PROGRAM AND IN SUBPROGRAM FUN.              00014860
C      THE COMPLEX FUNCTION FUN SHOULD BE A MONOTONE       00014870
C      DECREASING FUNCTION AS THE ARGUMENT G BECOMES LARGE..00014880
C      FOR REAL-ONLY FUNCTIONS, SUBPROGRAM 'ZLAGH1' IS ADVISED;00014890
C      HOWEVER, TWO REAL-FUNCTIONS F1(G),F2(G) MAY BE     00014900
C      INTEGRATED IN PARALLEL BY WRITING FUN=CMPLX(F1(G),F2(G))00014910
C      TOL= REAL TOLERANCE EXCEPTED AT CONVOLVED TAILS--I.E., 00014920
C      IF FILTER*FUN<TOL*MAX, THEN REST OF TAIL IS TRUNCATED.00014930
C      THIS IS DONE AT BOTH ENDS OF FILTER. TYPICALLY,    00014940
C      TOL <= .0001 IS USUALLY OK--BUT THIS DEPENDS ON   00014950
C      THE FUNCTION FUN AND PARAMETER X...IN GENERAL,     00014960
C      A 'SMALLER TOL' WILL USUALLY RESULT IN 'MORE ACCURACY' 00014970
C      BUT WITH 'MORE WEIGHTS' BEING USED. TOL IS NOT DIRECTLY00014980
C      RELATED TO TRUNCATION ERROR, BUT GENERALLY SERVES AS AN 00014990
C      APPROXIMATION INDICATOR... FOR VERY LARGE OR SMALL B, 00015000
C      ONE SHOULD USE A SMALLER TOL THAN RECOMMENDED ABOVE... 00015010
C      L= RESULTING NO. FILTER WTS. USED IN THE VARIABLE   00015020
C      CONVOLUTION (L DEPENDS ON TOL AND FUN).             00015030
C      MIN.L=15 AND MAX.L=236--WHICH COULD                00015040
C      OCCUR IF TOL IS VERY SMALL AND/OR FUN NOT DECREASING 00015050
C      VERY FAST...                                       00015060
C      * NEW= 1 IS NECESSARY 1ST TIME OR BRAND NEW X.     00015070
C      0 FOR ALL SUBSEQUENT CALLS WHERE X=(LAST X)-0.20   00015080
C      IS ASSUMED INTERNALLY BY THIS ROUTINE.             00015090
C      NOTE: IF THIS IS NOT TRUE, ROUTINE WILL            00015100
C      STILL ASSUME X=(LAST X)-0.20 ANYWAY...             00015110
C      IT IS THE USERS RESPONSIBILITY TO NORMALIZE        00015120
C      BY CORRECT B=EXP(X) OUTSIDE OF CALL (SEE USAGE BELOW).00015130
C      THE LAGGED CONVOLUTION METHOD PICKS UP SIGNIFICANT   00015140
C      TIME IMPROVEMENTS WHEN THE KERNEL IS NOT A        00015150
C      SIMPLE ELEMENTARY FUNCTION...DUE TO INTERNALLY SAVING 00015160
C      ALL KERNEL FUNCTION EVALUATIONS WHEN NEW=1...     00015170
C      THEN WHEN NEW=0, ALL PREVIOUSLY CALCULATED        00015180
C      KERNELS WILL BE USED IN THE LAGGED CONVOLUTION     00015190
C      WHERE POSSIBLE, ONLY ADDING NEW KERNEL EVALUATIONS 00015200
C      WHEN NEEDED (DEPENDS ON FARMS TOL AND FUN)        00015210
C      00015220
C--THE RESULTING COMPLEX CONVOLUTION SUM IS GIVEN IN ZLAGH1; THE HANKEL 00015230
C TRANSFORM IS THEN ZLAGH1/B WHICH IS TO BE COMPUTED AFTER EXIT FROM 00015240
C THIS ROUTINE.... WHERE B=EXP(X), X=ARGUMENT USED IN CALL... 00015250
C 00015260
C--USAGE-- 'ZLAGH1' IS CALLED AS FOLLOWS:                00015270
C ...                                                    00015280
C COMPLEX Z,ZLAGH1,ZF                                     00015290
C EXTERNAL ZF                                           00015300
C ...                                                    00015310
C Z=ZLAGH1(ALOG(B),ZF,TOL,L,NEW)/B                      00015320
C ...                                                    00015330
C END                                                    00015340
C COMPLEX FUNCTION ZF(G)                                00015350
  
```



```

C     ...USER SUPPLIED CODE...                                00015360
C     END                                                       00015370
C     END                                                       00015380
C--NOTES:                                                     00015390
C     (1). EXP--UNDERFLOW'S MAY OCCUR IN EXECUTING THE SUBPROGRAM 00015400
C     BELOW; HOWEVER, THIS IS OK PROVIDED THE MACHINE SYSTEM SETS 00015410
C     ANY & ALL EXP--UNDERFLOW'S TO 0.0....                    00015420
C     (2). AS AN AID TO UNDERSTANDING & USING THE LAGGED CONVOLUTION 00015430
C     METHOD, LET BMAX>=BMIN>0 BE GIVEN, THEN IT CAN BE SHOWN    00015440
C     THAT THE ACTUAL NUMBER OF B'S IS NB=AIN(5.*ALOG(BMAX/BMIN))+1, 00015450
C     PROVIDED BMAX/BMIN>=1. THE USER MAY THEN ASSUME AN 'ADJUSTED' 00015460
C     BMINA=BMAX*EXP(-.2*(NB-1)). THE METHOD GENERATES THE DECREASING 00015470
C     ARGUMENTS SPACED AS X=ALOG(BMAX),X-.2,X-.2*2,...,ALOG(BMINA), 00015480
C     FOR EXAMPLE, ONE MAY CONTROL THIS WITH THE CODE:          00015490
C     ...                                                       00015500
C     NB=AIN(5.*ALOG(BMAX/BMIN))+1                             00015510
C     NB1=NB+1                                                 00015520
C     X0=ALOG(BMAX)+.2                                         00015530
C     NEW=1                                                    00015540
C     DO 1 J=1,NB                                             00015550
C     I=NB1-J                                                 00015560
C     X=X0-.2*J                                               00015570
C     ARG(I)=EXP(X)                                           00015580
C     Z(I)=ZLAGH1(X,ZF,TOL,L,NEW)/ARG(I)                     00015590
C     1 NEW=0                                                 00015600
C     ...                                                       00015610
C     (3). IF RESULTS ARE STORED IN ARRAYS ARG(I),Z(I),I=1,NB FOR 00015620
C     ARG IN (BMINA,BMAX), THEN THESE ARRAYS MAY BE USED, FOR EXAMPLE, 00015630
C     TO SPLINE-INTERPOLATE AT A DIFFERENT (LARGER OR SMALLER)    00015640
C     SPACING THAN USED IN THE LAGGED CONVOLUTION METHOD,          00015650
C     (4). IF A DIFFERENT RANGE OF B IS DESIRED, THEN ONE MAY    00015660
C     ALWAYS RESTART THE ABOVE PROCEDURE IN (2) WITH A NEW      00015670
C     BMAX,BMIN AND BY SETTING NEW=1....                       00015680
C     (5). ABSCISSA CORRESPONDING TO WEIGHT IS GENERATED TO SAVE STORAGE 00015690
C     ...                                                       00015700
C     ...                                                       00015710
C     COMPLEX FUN,C,CMAX,SAVE                                  00015720
C     DIMENSION KEY(236),SAVE(236),T(2),TMAX(2)               00015730
C     DIMENSION WT(236),W1(76),W2(76),W3(76),W4(8)           00015740
C     EQUIVALENCE (C,T(1)),(CMAX,TMAX(1))                    00015750
C     EQUIVALENCE (WT(1),W1(1)),(WT(77),W2(1)),(WT(153),W3(1)), 00015760
C     1 (WT(229),W4(1))                                       00015770
C--J1-EXTENDED FILTER WEIGHT ARRAYS:                          00015780
C     DATA W1/                                               00015790
C     1-8.8863805E-10, 1.1293811E-09,-1.2050872E-09, 1.2696232E-09, 00015800
C     2-1.3223909E-09, 1.3642393E-09,-1.3969439E-09, 1.4225941E-09, 00015810
C     3-1.4427475E-09, 1.4580582E-09,-1.4682563E-09, 1.4732179E-09, 00015820
C     4-1.4735606E-09, 1.4719870E-09,-1.4727091E-09, 1.4828225E-09, 00015830
C     5-1.5102619E-09, 1.5667752E-09,-1.6634522E-09, 1.8172900E-09, 00015840
C     6-2.0412753E-09, 2.3595230E-09,-2.7861077E-09, 3.3592871E-09, 00015850
C     7-4.0940172E-09, 5.0571015E-09,-6.2604109E-09, 7.8269461E-09, 00015860
C     8-9.7514701E-09, 1.2267639E-08,-1.5312389E-08, 1.9339924E-08, 00015870

```

```
9-2.4126297E-08, 3.0576829E-08,-3.8060204E-08, 4.8423732E-08, 00015880
1-6.0051116E-08, 7.6787475E-08,-9.4700993E-08, 1.2192844E-07, 00015890
2-1.4918997E-07, 1.9392737E-07,-2.3464786E-07, 3.0911127E-07, 00015900
3-3.6815394E-07, 4.9413800E-07,-5.7554168E-07, 7.9301529E-07, 00015910
4-8.9502818E-07, 1.2794292E-06,-1.3811469E-06, 2.0789668E-06, 00015920
5-2.1069398E-06, 3.4103188E-06,-3.1584463E-06, 5.6639045E-06, 00015930
6-4.6059955E-06, 9.5561672E-06,-6.4142855E-06, 1.6440205E-05, 00015940
7-8.2010619E-06, 2.8945217E-05,-8.6348466E-06, 5.2317398E-05, 00015950
8-3.9915035E-06, 9.7273612E-05, 1.5220520E-05, 1.8614373E-04, 00015960
9 7.2023760E-05, 3.6620099E-04, 2.2062958E-04, 7.3874539E-04, 00015970
1 5.8623480E-04, 1.5226779E-03, 1.4538718E-03, 3.1930365E-03/ 00015980
DATA W2/ 00015990
1 3.4640868E-03, 6.7790882E-03, 8.0328420E-03, 1.4484339E-02, 00016000
2 1.8201316E-02, 3.0866143E-02, 4.0106549E-02, 6.4527872E-02, 00016010
3 8.4285526E-02, 1.2773175E-01, 1.6020907E-01, 2.1948043E-01, 00016020
4 2.3636305E-01, 2.4895051E-01, 1.2586300E-01,-5.1060445E-02, 00016030
5-3.4376222E-01,-2.9042175E-01, 1.1564736E-01, 4.9253231E-01, 00016040
6-4.6748595E-01, 1.5280945E-01, 3.3348541E-02,-8.2485252E-02, 00016050
7.7.9740630E-02,-6.6934498E-02, 5.5150465E-02,-4.5868721E-02, 00016060
8 3.8651958E-02,-3.2935834E-02, 2.8303994E-02,-2.4475127E-02, 00016070
9 2.1259541E-02,-1.8526278E-02, 1.6182037E-02,-1.4158101E-02, 00016080
1 1.2402225E-02,-1.0873526E-02, 9.5392016E-03,-8.3723743E-03, 00016090
2 7.3506490E-03,-6.4551136E-03, 5.6696335E-03,-4.9803353E-03, 00016100
3 4.3752213E-03,-3.8438703E-03, 3.3772023E-03,-2.9672872E-03, 00016110
4 2.6071877E-03,-2.2908274E-03, 2.0128794E-03,-1.7686706E-03, 00016120
5 1.5540998E-03,-1.3655666E-03, 1.1999089E-03,-1.0543497E-03, 00016130
6 9.2644973E-04,-8.1406593E-04, 7.1531559E-04,-6.2854459E-04, 00016140
7 5.5229955E-04,-4.8530352E-04, 4.2643446E-04,-3.7470650E-04, 00016150
8 3.2925334E-04,-2.8931382E-04, 2.5421910E-04,-2.2338147E-04, 00016160
9 1.9628455E-04,-1.7247455E-04, 1.5155278E-04,-1.3316889E-04, 00016170
1 1.1701502E-04,-1.0282066E-04, 9.0348135E-05,-7.9388568E-05/ 00016180
DATA W3/ 00016190
1 6.9758436E-05,-6.1296474E-05, 5.3860978E-05,-4.7327436E-05, 00016200
2 4.1586435E-05,-3.6541840E-05, 3.2109174E-05,-2.8214208E-05, 00016210
3 2.4791718E-05,-2.1784390E-05, 1.9141864E-05,-1.6819888E-05, 00016220
4 1.4779578E-05,-1.2986765E-05, 1.1411426E-05,-1.0027182E-05, 00016230
5 8.8108499E-06,-7.7420630E-06, 6.8029235E-06,-5.9777053E-06, 00016240
6 5.2525892E-06,-4.6154325E-06, 4.0555653E-06,-3.5636118E-06, 00016250
7 3.1313335E-06,-2.7514911E-06, 2.4177236E-06,-2.1244417E-06, 00016260
8 1.8667342E-06,-1.6402859E-06, 1.4413051E-06,-1.2664597E-06, 00016270
9 1.1128220E-06,-9.7781908E-07, 8.5919028E-07,-7.5494920E-07, 00016280
1 6.6335060E-07,-5.8286113E-07, 5.1213358E-07,-4.4998431E-07, 00016290
2 3.9537334E-07,-3.4738689E-07, 3.0522189E-07,-2.6817250E-07, 00016300
3 2.3561831E-07,-2.0701397E-07, 1.8188012E-07,-1.5979545E-07, 00016310
4 1.4038968E-07,-1.2333746E-07, 1.0835294E-07,-9.5185048E-08, 00016320
5 8.3613184E-08,-7.3443411E-08, 6.4505118E-08,-5.6648167E-08, 00016330
6 4.9740428E-08,-4.3665572E-08, 3.8321109E-08,-3.3616717E-08, 00016340
7 2.9472836E-08,-2.5819439E-08, 2.2594957E-08,-1.9745353E-08, 00016350
8 1.7223359E-08,-1.4987869E-08, 1.3003472E-08,-1.1240058E-08, 00016360
9 9.6723739E-09,-8.2794392E-09, 7.0438407E-09,-5.9509676E-09, 00016370
1 4.9882405E-09,-4.1443813E-09, 3.4088114E-09,-2.7712762E-09/ 00016380
DATA W4/ 00016390
```

1 2.221731E-09,-1.7504755E-09, 1.3485207E-09,-1.0080937E-09,  
2 7.2300885E-10,-4.8860666E-10, 3.0121413E-10,-9.1649798E-11/  
C--\$ENDATA

```
C  
IF(NEW) 10,30,10  
10 LAG=-1  
X0=-X-17.0  
DO 20 IR=1,236  
20 KEY(IR)=0  
30 LAG=LAG+1  
ZLAGH1=(0.0,0.0)  
CMAX=(0.0,0.0)  
L=0  
ASSIGN 110 TO M  
I=86  
GO TO 200  
110 TMAX(1)=AMAX1(ABS(T(1)),TMAX(1))  
TMAX(2)=AMAX1(ABS(T(2)),TMAX(2))  
I=I+1  
IF(I.LE.98) GO TO 200  
IF(TMAX(1).EQ.0.0.AND.TMAX(2).EQ.0.0) GO TO 150  
CMAX=TOL*CMAX  
ASSIGN 120 TO M  
I=85  
GO TO 200  
120 IF(ABS(T(1)).LE.TMAX(1).AND.ABS(T(2)).LE.TMAX(2)) GO TO 130  
I=I-1  
IF(I.GT.0) GO TO 200  
ASSIGN 140 TO M  
I=99  
GO TO 200  
140 IF(ABS(T(1)).LE.TMAX(1).AND.ABS(T(2)).LE.TMAX(2)) GO TO 190  
I=I+1  
IF(I.LE.236) GO TO 200  
GO TO 190  
ASSIGN 160 TO M  
I=1  
GO TO 200  
150 IF(T(1).EQ.0.0.AND.T(2).EQ.0.0) GO TO 170  
I=I+1  
IF(I.LE.85) GO TO 200  
ASSIGN 180 TO M  
I=236  
GO TO 200  
180 IF(T(1).EQ.0.0.AND.T(2).EQ.0.0) GO TO 190  
I=I-1  
IF(I.GE.99) GO TO 200  
RETURN  
190 C--STORE/RETRIEVE ROUTINE (DONE INTERNALLY TO SAVE CALL'S)  
LOOK=I+LAG  
IQ=LOOK/237  
IR=MOD(LOOK,237)  
00016910  
00016900  
00016890  
00016880  
00016870  
00016860  
00016850  
00016840  
00016830  
00016820  
00016810  
00016800  
00016790  
00016780  
00016770  
00016760  
00016750  
00016740  
00016730  
00016720  
00016710  
00016700  
00016690  
00016680  
00016670  
00016660  
00016650  
00016640  
00016630  
00016620  
00016610  
00016600  
00016590  
00016580  
00016570  
00016560  
00016550  
00016540  
00016530  
00016520  
00016510  
00016500  
00016490  
00016480  
00016470  
00016460  
00016450  
00016440  
00016430  
00016420  
00016410  
00016400
```

```

IF(IR.EQ.0) IR=1                                00016920
IROLL=IQ*236                                     00016930
IF(KEY(IR).LE.IROLL) GO TO 220                   00016940
210 C=SAVE(IR)*WT(I)                             00016950
ZLAGH1=ZLAGH1+C                                 00016960
L=L+1                                            00016970
GO TO M,(110,120,140,160,180)                   00016980
220 KEY(IR)=IROLL+IR                             00016990
SAVE(IR)=FUN(EXP(X0+FLOAT(LOOK)*.20))           00017000
GO TO 210                                        00017010
END                                              00017020
  
```

COMPLEX FUNCTION ZLAGFO(X,FUN,TOL,L,NEW) 00017030

```

C---*** A SPECIAL LAGGED* CONVOLUTION METHOD TO COMPUTE THE 00017040
C INTEGRAL FROM 0 TO INFINITY OF 'FUN(G)*COS(G*B)*DG' DEFINED AS THE 00017050
C COMPLEX FOURIER COSINE TRANSFORM WITH ARGUMENT X(=ALOG(B)) 00017060
C BY CONVOLUTION FILTERING WITH COMPLEX FUNCTION 'FUN'---AND 00017070
C USING A VARIABLE CUT-OFF METHOD WITH EXTENDED FILTER TAILS... 00017080
C 00017090
C---REF: ANDERSON, W.L., 1975, NTIS REPT. PB-242-800. 00017100
C 00017110
C---PARAMETERS: 00017120
C 00017130
C * X = REAL ARGUMENT(=ALOG(B) AT CALL) OF THE FOURIER TRANSFORM 00017140
C 'ZLAGFO' IS USEFUL ONLY WHEN X=(LAST X)-.20 *** I.E., 00017150
C SPACED SAME AS FILTER USED---IF THIS IS NOT CONVENIENT, 00017160
C THEN SUBPROGRAM 'ZFOURO' IS ADVISED FOR GENERAL USE. 00017170
C (ALSO SEE PARM 'NEW' & NOTES (2)-(4) BELOW). 00017180
C FUN(G)= EXTERNAL DECLARED COMPLEX FUNCTION NAME (USER SUPPLIED) 00017190
C OF A REAL ARGUMENT G. 00017200
C NOTE: IF PARMS OTHER THAN G ARE REQUIRED, USE COMMON IN 00017210
C CALLING PROGRAM AND IN SUBPROGRAM FUN. 00017220
C THE COMPLEX FUNCTION FUN SHOULD BE A MONOTONE 00017230
C DECREASING FUNCTION AS THE ARGUMENT G BECOMES LARGE... 00017240
C FOR REAL-ONLY FUNCTIONS, SUBPROGRAM 'RLAGFO' IS ADVISED; 00017250
C HOWEVER, TWO REAL-FUNCTIONS F1(G),F2(G) MAY BE 00017260
C INTEGRATED IN PARALLEL BY WRITING FUN=CMPLX(F1(G),F2(G)) 00017270
C TOL= REAL TOLERANCE EXCEPTED AT CONVOLVED TAILS---I.E., 00017280
C IF FILTER*FUN<TOL*MAX, THEN REST OF TAIL IS TRUNCATED. 00017290
C THIS IS DONE AT BOTH ENDS OF FILTER. TYPICALLY, 00017300
C TOL <= .0001 IS USUALLY OK---BUT THIS DEPENDS ON 00017310
C THE FUNCTION FUN AND PARAMETER X...IN GENERAL, 00017320
C A 'SMALLER TOL' WILL USUALLY RESULT IN 'MORE ACCURACY' 00017330
C BUT WITH 'MORE WEIGHTS' BEING USED. TOL IS NOT DIRECTLY 00017340
C RELATED TO TRUNCATION ERROR, BUT GENERALLY SERVES AS AN 00017350
C APPROXIMATION INDICATOR... FOR VERY LARGE OR SMALL B, 00017360
C ONE SHOULD USE A SMALLER TOL THAN RECOMMENDED ABOVE... 00017370
C L= RESULTING NO. FILTER WTS. USED IN THE VARIABLE 00017380
C CONVOLUTION (L DEPENDS ON TOL AND FUN). 00017390
C MIN.L=24 AND MAX.L=281---WHICH COULD 00017400
C OCCUR IF TOL IS VERY SMALL AND/OR FUN NOT DECREASING 00017410
C VERY FAST... 00017420
  
```

```

C      * NEW=      1 IS NECESSARY 1ST TIME OR BRAND NEW X.      00017430
C      0 FOR ALL SUBSEQUENT CALLS WHERE X=(LAST X)-0.20      00017440
C      IS ASSUMED INTERNALLY BY THIS ROUTINE.      00017450
C      NOTE: IF THIS IS NOT TRUE, ROUTINE WILL      00017460
C      STILL ASSUME X=(LAST X)-0.20 ANYWAY...      00017470
C      IT IS THE USERS RESPONSIBILITY TO NORMALIZE      00017480
C      BY CORRECT B=EXP(X) OUTSIDE OF CALL (SEE USAGE BELOW). 00017490
C      THE LAGGED CONVOLUTION METHOD PICKS UP SIGNIFICANT      00017500
C      TIME IMPROVEMENTS WHEN THE KERNEL IS NOT A      00017510
C      SIMPLE ELEMENTARY FUNCTION...DUE TO INTERNALLY SAVING 00017520
C      ALL KERNEL FUNCTION EVALUATIONS WHEN NEW=1...      00017530
C      THEN WHEN NEW=0, ALL PREVIOUSLY CALCULATED      00017540
C      KERNELS WILL BE USED IN THE LAGGED CONVOLUTION      00017550
C      WHERE POSSIBLE, ONLY ADDING NEW KERNEL EVALUATIONS      00017560
C      WHEN NEEDED (DEPENDS ON PARMS TOL AND FUN)      00017570
C      00017580
C--THE RESULTING COMPLEX CONVOLUTION SUM IS GIVEN IN ZLAGFO; THE FOURIER00017590
C TRANSFORM IS THEN ZLAGFO/B WHICH IS TO BE COMPUTED AFTER EXIT FROM 00017600
C THIS ROUTINE.... WHERE B=EXP(X), X=ARGUMENT USED IN CALL... 00017610
C 00017620
C--USAGE-- 'ZLAGFO' IS CALLED AS FOLLOWS:      00017630
C      ...      00017640
C      COMPLEX Z ZLAGFO,ZF      00017650
C      EXTERNAL ZF      00017660
C      ...      00017670
C      Z=ZLAGFO(ALOG(B),ZF,TOL,L,NEW)/B      00017680
C      ...      00017690
C      END      00017700
C      COMPLEX FUNCTION ZF(G)      00017710
C      ...USER SUPPLIED CODE...      00017720
C      END      00017730
C 00017740
C--NOTES:      00017750
C      (1). EXP-UNDERFLOW'S MAY OCCUR IN EXECUTING THE SUBPROGRAM 00017760
C      BELOW; HOWEVER, THIS IS OK PROVIDED THE MACHINE SYSTEM SETS 00017770
C      ANY & ALL EXP-UNDERFLOW'S TO 0.0....      00017780
C      (2). AS AN AID TO UNDERSTANDING & USING THE LAGGED CONVOLUTION 00017790
C      METHOD, LET BMAX>=BMIN>0 BE GIVEN, THEN IT CAN BE SHOWN      00017800
C      THAT THE ACTUAL NUMBER OF B'S IS NB=AIN(5.*ALOG(BMAX/BMIN))+1, 00017810
C      PROVIDED BMAX/BMIN>=1. THE USER MAY THEN ASSUME AN 'ADJUSTED' 00017820
C      BMINA=BMAX*EXP(-.2*(NB-1)). THE METHOD GENERATES THE DECREASING 00017830
C      ARGUMENTS SPACED AS X=ALOG(BMAX),X-.2,X-.2*2,...,ALOG(BMINA). 00017840
C      FOR EXAMPLE, ONE MAY CONTROL THIS WITH THE CODE:      00017850
C      ...      00017860
C      NB=AIN(5.*ALOG(BMAX/BMIN))+1      00017870
C      NB1=NB+1      00017880
C      X0=ALOG(BMAX)+.2      00017890
C      NEW=1      00017900
C      DO 1 J=1,NB      00017910
C      I=NB1-J      00017920
C      X=X0-.2*J      00017930
C      ARG(I)=EXP(X)      00017940

```

```

C          Z(I)=ZLAGFO(X,ZF,TOL,L,NEW)/ARG(I)          00017950
C          1          NEW=0                              00017960
C          ...                                          00017970
C          (3). IF RESULTS ARE STORED IN ARRAYS ARG(I),Z(I),I=1,NB FOR 00017980
C          ARG IN (BMINA,BMAX), THEN THESE ARRAYS MAY BE USED, FOR EXAMPLE, 00017990
C          TO SPLINE-INTERPOLATE AT A DIFFERENT (LARGER OR SMALLER) 00018000
C          SPACING THAN USED IN THE LAGGED CONVOLUTION METHOD. 00018010
C          (4). IF A DIFFERENT RANGE OF B IS DESIRED, THEN ONE MAY 00018020
C          ALWAYS RESTART THE ABOVE PROCEDURE IN (2) WITH A NEW 00018030
C          BMAX,BMIN AND BY SETTING NEW=1,... 00018040
C          (5). ABSCISSA CORRESPONDING TO WEIGHT IS GENERATED TO SAVE STORAGE 00018050
C          00018060
C          00018070
C          COMPLEX FUN,C,CMAX,SAVE 00018080
C          DIMENSION KEY(281),SAVE(281),T(2),TMAX(2) 00018090
C          DIMENSION YT(281),Y1(76),Y2(76),Y3(76),Y4(53) 00018100
C          EQUIVALENCE (C,T(1)),(CMAX,TMAX(1)) 00018110
C          EQUIVALENCE (YT(1),Y1(1)),(YT(77),Y2(1)),(YT(153),Y3(1)), 00018120
C          1 (YT(229),Y4(1)) 00018130
C--COS-EXTENDED FILTER WEIGHT ARRAYS: 00018140
C          DATA Y1/ 00018150
C          1 5.1178101E-14, 2.9433849E-14, 2.5492522E-14, 1.9034819E-14, 00018160
C          2 6.4179780E-14, 1.3085746E-15, 1.1989957E-13, -1.2216234E-14, 00018170
C          3 1.7534103E-13, 7.9373498E-15, 2.1235658E-13, 7.9981520E-14, 00018180
C          4 2.3815757E-13, 1.9714260E-13, 2.8920132E-13, 3.4161340E-13, 00018190
C          5 4.0349917E-13, 5.2203885E-13, 5.9837223E-13, 7.8015306E-13, 00018200
C          6 8.8911655E-13, 1.1709731E-12, 1.3165595E-12, 1.7578463E-12, 00018210
C          7 1.9538564E-12, 2.6289768E-12, 2.9167697E-12, 3.9044344E-12, 00018220
C          8 4.3927341E-12, 5.7526904E-12, 6.6569552E-12, 8.4555678E-12, 00018230
C          9 1.0063229E-11, 1.2487964E-11, 1.5134682E-11, 1.8501488E-11, 00018240
C          1 2.2720051E-11, 2.7452598E-11, 3.4025443E-11, 4.0875985E-11, 00018250
C          2 5.0751668E-11, 6.1094382E-11, 7.5492982E-11, 9.1445759E-11, 00018260
C          3 1.1227336E-10, 1.3676464E-10, 1.6720269E-10, 2.0423244E-10, 00018270
C          4 2.4932743E-10, 3.0470661E-10, 3.7198526E-10, 4.5449934E-10, 00018280
C          5 5.5502537E-10, 6.7793669E-10, 8.2810001E-10, 1.0112626E-09, 00018290
C          6 1.2354800E-09, 1.5085255E-09, 1.8432253E-09, 2.2503397E-09, 00018300
C          7 2.7499027E-09, 3.3569525E-09, 4.1025670E-09, 5.0077487E-09, 00018310
C          8 6.1205950E-09, 7.4703399E-09, 9.1312760E-09, 1.1143911E-08, 00018320
C          9 1.3622929E-08, 1.6623917E-08, 2.0324094E-08, 2.4798610E-08, 00018330
C          1 3.0321709E-08, 3.6992986E-08, 4.5237482E-08, 5.5183434E-08/ 00018340
C          DATA Y2/ 00018350
C          1 6.7491070E-08, 8.2317946E-08, 1.0069271E-07, 1.2279375E-07, 00018360
C          2 1.5022907E-07, 1.8316969E-07, 2.2413747E-07, 2.7322865E-07, 00018370
C          3 3.3441046E-07, 4.0756197E-07, 4.9894278E-07, 6.0793233E-07, 00018380
C          4 7.4443665E-07, 9.0679753E-07, 1.1107379E-06, 1.3525651E-06, 00018390
C          5 1.6573073E-06, 2.0174273E-06, 2.4728798E-06, 3.0090445E-06, 00018400
C          6 3.6898816E-06, 4.4879625E-06, 5.5059521E-06, 6.6935820E-06, 00018410
C          7 8.2160716E-06, 9.9828691E-06, 1.2260527E-05, 1.4888061E-05, 00018420
C          8 1.8296530E-05, 2.2202672E-05, 2.7305154E-05, 3.3109672E-05, 00018430
C          9 4.0751046E-05, 4.9372484E-05, 6.0820947E-05, 7.3619571E-05, 00018440
C          1 9.0780005E-05, 1.0976837E-04, 1.3550409E-04, 1.6365676E-04, 00018450
C          2 2.0227521E-04, 2.4398338E-04, 3.0197018E-04, 3.6370760E-04, 00018460

```

```
3 4.5083748E-04, 5.4213338E-04, 6.7315347E-04, 8.0800951E-04, 00018470
4 1.0051938E-03, 1.2041401E-03, 1.5011708E-03, 1.7942344E-03, 00018480
5 2.2421056E-03, 2.6730676E-03, 3.3490681E-03, 3.9815050E-03, 00018490
6 5.0028666E-03, 5.9285668E-03, 7.4730905E-03, 8.8233510E-03, 00018500
7 1.1160132E-02, 1.3119627E-02, 1.6653199E-02, 1.9472767E-02, 00018510
8 2.4800811E-02, 2.8793704E-02, 3.6762063E-02, 4.2228780E-02, 00018520
9 5.3905163E-02, 6.0804660E-02, 7.7081738E-02, 8.3874501E-02, 00018530
1 1.0377190E-01, 1.0377718E-01, 1.1892208E-01, 9.0437429E-02/ 00018540
DATA Y3/ 00018550
1 7.1685138E-02,-3.9473064E-02,-1.5078720E-01,-4.0489859E-01, 00018560
2-5.6018995E-01,-6.8050388E-01,-1.5094224E-01, 6.6304064E-01, 00018570
3 1.3766748E+00,-8.0373222E-01,-1.0869629E+00, 1.2812892E+00, 00018580
4-5.0341082E-01,-4.4274455E-02, 2.0913102E-01,-1.9999661E-01, 00018590
5 1.5207664E-01,-1.0920260E-01, 7.8169956E-02,-5.6651561E-02, 00018600
6 4.1611799E-02,-3.0880012E-02, 2.3072559E-02,-1.7311631E-02, 00018610
7 1.3021442E-02,-9.8085025E-03, 7.3943529E-03,-5.5769518E-03, 00018620
8 4.2073164E-03,-3.1745026E-03, 2.3954154E-03,-1.8076122E-03, 00018630
9 1.3640816E-03,-1.0293934E-03, 7.7682952E-04,-5.8623518E-04, 00018640
1 4.4240399E-04,-3.3386183E-04, 2.5195025E-04,-1.9013541E-04, 00018650
2 1.4348659E-04,-1.0828284E-04, 8.1716174E-05,-6.1667509E-05, 00018660
3 4.6537684E-05,-3.5119887E-05, 2.6503388E-05,-2.0000904E-05, 00018670
4 1.5093768E-05,-1.1390572E-05, 8.5959318E-06,-6.4869407E-06, 00018680
5 4.8953713E-06,-3.6942830E-06, 2.7878625E-06,-2.1038241E-06, 00018690
6 1.5875917E-06,-1.1980090E-06, 9.0398030E-07,-6.8208296E-07, 00018700
7 5.1458650E-07,-3.8817581E-07, 2.9272267E-07,-2.2067921E-07, 00018710
8 1.6623514E-07,-1.2514102E-07, 9.4034535E-08,-7.0556837E-08, 00018720
9 5.2741581E-08,-3.9298610E-08, 2.9107255E-08,-2.1413893E-08, 00018730
1 1.5742032E-08,-1.1498608E-08, 8.7561571E-09,-7.2959446E-09/ 00018740
DATA Y4/ 00018750
1 6.8816619E-09,-8.9679825E-09, 1.4258275E-08,-1.9564299E-08, 00018760
2 2.0235313E-08,-1.4725545E-08, 5.4632820E-09, 3.5995580E-09, 00018770
3-9.5287133E-09, 1.1460041E-08,-1.0250532E-08, 7.4641748E-09, 00018780
4-4.4703465E-09, 2.0499053E-09,-4.4806353E-10,-4.0374336E-10, 00018790
5 7.0321001E-10,-6.7067960E-10, 4.9130404E-10,-2.8840747E-10, 00018800
6 1.2373144E-10,-1.5260443E-11,-4.2027559E-11, 6.1885474E-11, 00018810
7-5.9273937E-11, 4.6588766E-11,-3.2054182E-11, 1.9831637E-11, 00018820
8-1.1210098E-11, 5.9567021E-12,-3.2427812E-12, 2.1353868E-12, 00018830
9-1.8476851E-12, 1.8438474E-12,-1.8362842E-12, 1.7241847E-12, 00018840
1-1.5161479E-12, 1.2627657E-12,-1.0129176E-12, 7.9578625E-13, 00018850
2-6.2131435E-13, 4.8745900E-13,-3.8703630E-13, 3.1172547E-13, 00018860
3-2.5397802E-13, 2.0824130E-13,-1.7123163E-13, 1.4113344E-13, 00018870
4-1.1687986E-13, 9.7664016E-14,-8.2977176E-14, 7.2515267E-14, 00018880
5-5.6047478E-14/ 00018890
C--$$ENDATA 00018900
IF(NEW) 10,30,10 00018910
10 LAG=-1 00018920
X0=-X-30.30251236 00018930
DO 20 IR=1,281 00018940
20 KEY(IR)=0 00018950
30 LAG=LAG+1 00018960
ZLAGF0=(0,0,0,0) 00018970
CMAX=(0,0,0,0) 00018980
```

	L=0	00018990
	ASSIGN 110 TO M	00019000
	I=149	00019010
	GO TO 200	00019020
110	TMAX(1)=AMAX1(ABS(T(1)),TMAX(1))	00019030
	TMAX(2)=AMAX1(ABS(T(2)),TMAX(2))	00019040
	I=I+1	00019050
	IF(I.LE.170) GO TO 200	00019060
	IF(TMAX(1).EQ.0.0.AND.TMAX(2).EQ.0.0) GO TO 150	00019070
	CMAX=TOL*CMAX	00019080
	ASSIGN 120 TO M	00019090
	I=148	00019100
	GO TO 200	00019110
120	IF(ABS(T(1)).LE.TMAX(1).AND.ABS(T(2)).LE.TMAX(2)) GO TO 130	00019120
	I=I-1	00019130
	IF(I.GT.0) GO TO 200	00019140
130	ASSIGN 140 TO M	00019150
	I=171	00019160
	GO TO 200	00019170
140	IF(ABS(T(1)).LE.TMAX(1).AND.ABS(T(2)).LE.TMAX(2)) GO TO 190	00019180
	I=I+1	00019190
	IF(I.LE.281) GO TO 200	00019200
	GO TO 190	00019210
150	ASSIGN 160 TO M	00019220
	I=1	00019230
	GO TO 200	00019240
160	IF(T(1).EQ.0.0.AND.T(2).EQ.0.0) GO TO 170	00019250
	I=I+1	00019260
	IF(I.LE.148) GO TO 200	00019270
170	ASSIGN 180 TO M	00019280
	I=281	00019290
	GO TO 200	00019300
180	IF(T(1).EQ.0.0.AND.T(2).EQ.0.0) GO TO 190	00019310
	I=I-1	00019320
	IF(I.GE.171) GO TO 200	00019330
190	RETURN	00019340
	C--STORE/RETRIEVE ROUTINE (DONE INTERNALLY TO SAVE CALL'S)	00019350
200	LOOK=I+LAG	00019360
	IQ=LOOK/282	00019370
	IR=MOD(LOOK,282)	00019380
	IF(IR.EQ.0) IR=1	00019390
	IROLL=IQ*281	00019400
	IF(KEY(IR).LE.IROLL) GO TO 220	00019410
210	C=SAVE(IR)*YT(I)	00019420
	ZLAGFO=ZLAGFO+C	00019430
	L=L+1	00019440
	GO TO M,(110,120,140,160,180)	00019450
220	KEY(IR)=IROLL+IR	00019460
	SAVE(IR)=FUN(EXP(X0+FLOAT(LOOK)*.20))	00019470
	GO TO 210	00019480
	END	00019490



```

COMPLEX FUNCTION ZLAGF1(X,FUN,TOL,L,NEW)                                00019500
C---*** A SPECIAL LAGGED* CONVOLUTION METHOD TO COMPUTE THE          00019510
C INTEGRAL FROM 0 TO INFINITY OF 'FUN(G)*SIN(G*B)*DG' DEFINED AS THE 00019520
C COMPLEX FOURIER SINE TRANSFORM WITH ARGUMENT X(=ALOG(B))          00019530
C BY CONVOLUTION FILTERING WITH COMPLEX FUNCTION 'FUN'---AND       00019540
C USING A VARIABLE CUT-OFF METHOD WITH EXTENDED FILTER TAILS....    00019550
C                                                                    00019560
C---REF: ANDERSON, W.L., 1975, NTIS REPT. PB-242-800.              00019570
C                                                                    00019580
C---PARAMETERS:                                                    00019590
C                                                                    00019600
C   * X      = REAL ARGUMENT(=ALOG(B) AT CALL) OF THE FOURIER TRANSFORM00019610
C             'ZLAGF1' IS USEFUL ONLY WHEN X=(LAST X)-.20 *** I.E., 00019620
C             SPACED SAME AS FILTER USED---IF THIS IS NOT CONVENIENT, 00019630
C             THEN SUBPROGRAM 'ZFOUR1' IS ADVISED FOR GENERAL USE.   00019640
C             (ALSO SEE PARM 'NEW' & NOTES (2)--(4) BELOW).          00019650
C   FUN(G)=  EXTERNAL DECLARED COMPLEX FUNCTION NAME (USER SUPPLIED) 00019660
C             OF A REAL ARGUMENT G.                                  00019670
C   NOTE: IF PARMS OTHER THAN G ARE REQUIRED, USE COMMON IN          00019680
C           CALLING PROGRAM AND IN SUBPROGRAM FUN.                  00019690
C           THE COMPLEX FUNCTION FUN SHOULD BE A MONOTONE            00019700
C           DECREASING FUNCTION AS THE ARGUMENT G BECOMES LARGE... 00019710
C           FOR REAL-ONLY FUNCTIONS, SUBPROGRAM 'RLAGF1' IS ADVISED;00019720
C           HOWEVER, TWO REAL-FUNCTIONS F1(G),F2(G) MAY BE          00019730
C           INTEGRATED IN PARALLEL BY WRITING FUN=CMLX(F1(G),F2(G))00019740
C   TOL=     REAL TOLERANCE EXCEPTED AT CONVOLVED TAILS---I.E.,   00019750
C           IF FILTER*FUN<TOL*MAX, THEN REST OF TAIL IS TRUNCATED.00019760
C           THIS IS DONE AT BOTH ENDS OF FILTER. TYPICALLY,        00019770
C           TOL <= .0001 IS USUALLY OK---BUT THIS DEPENDS ON      00019780
C           THE FUNCTION FUN AND PARAMETER X...IN GENERAL,         00019790
C           A 'SMALLER TOL' WILL USUALLY RESULT IN 'MORE ACCURACY' 00019800
C           BUT WITH 'MORE WEIGHTS' BEING USED. TOL IS NOT DIRECTLY00019810
C           RELATED TO TRUNCATION ERROR, BUT GENERALLY SERVES AS AN 00019820
C           APPROXIMATION INDICATOR... FOR VERY LARGE OR SMALL B,  00019830
C           ONE SHOULD USE A SMALLER TOL THAN RECOMMENDED ABOVE... 00019840
C   L=       RESULTING NO. FILTER WTS. USED IN THE VARIABLE        00019850
C           CONVOLUTION (L DEPENDS ON TOL AND FUN).                 00019860
C           MIN.L=20 AND MAX.L=266---WHICH COULD                   00019870
C           OCCUR IF TOL IS VERY SMALL AND/OR FUN NOT DECREASING  00019880
C           VERY FAST...                                           00019890
C   * NEW=   1 IS NECESSARY 1ST TIME OR BRAND NEW X.               00019900
C           0 FOR ALL SUBSEQUENT CALLS WHERE X=(LAST X)-0.20       00019910
C           IS ASSUMED INTERNALLY BY THIS ROUTINE.                 00019920
C           NOTE: IF THIS IS NOT TRUE, ROUTINE WILL                00019930
C           STILL ASSUME X=(LAST X)-0.20 ANYWAY...                 00019940
C           IT IS THE USERS RESPONSIBILITY TO NORMALIZE            00019950
C           BY CORRECT B=EXP(X) OUTSIDE OF CALL (SEE USAGE BELOW).00019960
C           THE LAGGED CONVOLUTION METHOD PICKS UP SIGNIFICANT      00019970
C           TIME IMPROVEMENTS WHEN THE KERNEL IS NOT A            00019980
C           SIMPLE ELEMENTARY FUNCTION...DUE TO INTERNALLY SAVING  00019990
C           ALL KERNEL FUNCTION EVALUATIONS WHEN NEW=1...         00020000
C           THEN WHEN NEW=0, ALL PREVIOUSLY CALCULATED             00020010
  
```

```

00020020 KERNELS WILL BE USED IN THE LAGGED CONVOLUTION
00020030 WHERE POSSIBLE, ONLY ADDING NEW KERNEL EVALUATIONS
00020040 WHEN NEEDED (DEPENDS ON PARAMS TOL AND FUN)
00020050
00020060 C---THE RESULTING COMPLEX CONVOLUTION SUM IS GIVEN IN ZLAGF1; THE FOURIER
00020070 C TRANSFORM IS THEN ZLAGF1/B WHICH IS TO BE COMPUTED AFTER EXIT FROM
00020080 C THIS ROUTINE... WHERE B=EXP(X)* X=ARGUMENT USED IN CALL...
00020090
00020100 C---USAGE--- ZLAGF1, IS CALLED AS FOLLOWS:
00020110
00020120 COMPLEX Z,ZLAGF1,ZF
00020130 EXTERNAL ZF
00020140
00020150 Z=ZLAGF1(ALOG(B),ZF,TOL,L,NEW)/B
00020160
00020170
00020180 COMPLEX FUNCTION ZF(G)
00020190
00020200 END
00020210
00020220 C---NOTES:
00020230 (1), EXP-UNDERFLOW'S MAY OCCUR IN EXECUTING THE SUBPROGRAM
00020240 BELOW; HOWEVER, THIS IS OK PROVIDED THE MACHINE SYSTEM SETS
00020250 ANY & ALL EXP-UNDERFLOW'S TO 0.0...
00020260 (2), AS AN AID TO UNDERSTANDING & USING THE LAGGED CONVOLUTION
00020270 METHOD, LET BMAX>=BMIN>0 BE GIVEN, THEN IT CAN BE SHOWN
00020280 THAT THE ACTUAL NUMBER OF B'S IS NB=AINT(5,*ALOG(BMAX/BMIN))+1,
00020290 PROVIDED BMAX/BMIN>=1, THE USER MAY THEN ASSUME AN ADJUSTED
00020300 BMINA=BMAX*EXP(-.2*(NB-1)). THE METHOD GENERATES THE DECREASING
00020310 ARGUMENTS SPACED AS X=ALOG(BMAX)*X-.2*X-.*2*2,...,ALOG(BMINA).
00020320 FOR EXAMPLE, ONE MAY CONTROL THIS WITH THE CODE:
00020330
00020340 NB=AINT(5,*ALOG(BMAX/BMIN))+1
00020350 NBI=NB+1
00020360 X0=ALOG(BMAX)+.2
00020370
00020380 DO 1 J=1,NB
00020390 I=NB1-J
00020400 X=X0-.2*I
00020410 ARG(I)=EXP(X)
00020420 Z(I)=ZLAGF1(X,ZF,TOL,L,NEW)/ARG(I)
00020430
00020440
00020450 (3), IF RESULTS ARE STORED IN ARRAYS ARG(I),Z(I),I=1,NB FOR
00020460 ARG IN (BMINA,BMAX), THEN THESE ARRAYS MAY BE USED, FOR EXAMPLE,
00020470 TO SPLINE-INTERPOLATE AT A DIFFERENT (LARGER OR SMALLER)
00020480 SPACING THAN USED IN THE LAGGED CONVOLUTION METHOD.
00020490 (4), IF A DIFFERENT RANGE OF B IS DESIRED, THEN ONE MAY
00020500 ALWAYS RESTART THE ABOVE PROCEDURE IN (2) WITH A NEW
00020510 BMAX,BMIN AND BY SETTING NEW=1...
00020520 (5), ABSCISSA CORRESPONDING TO WEIGHT IS GENERATED TO SAVE STORAGE
00020530

```

```

C
COMPLEX FUN,C,CMAX,SAVE                                00020540
DIMENSION KEY(266),SAVE(266),T(2),TMAX(2)             00020550
DIMENSION WT(266),W1(76),W2(76),W3(76),W4(38)        00020560
EQUIVALENCE (C,T(1)),(CMAX,TMAX(1))                  00020570
EQUIVALENCE (WT(1),W1(1)),(WT(77),W2(1)),(WT(153),W3(1)), 00020580
1 (WT(229),W4(1))                                     00020590
C---SIN-EXTENDED FILTER WEIGHT ARRAYS:                00020600
DATA W1/                                               00020610
1-1.1113940E-09,-1.3237246E-12, 1.5091739E-12,-1.6240954E-12, 00020620
2 1.7236636E-12,-1.8227727E-12, 1.9255992E-12,-2.0335514E-12, 00020630
3 2.1473541E-12,-2.2675549E-12, 2.3946842E-12,-2.5292661E-12, 00020640
4 2.6718110E-12,-2.8227693E-12, 2.9825171E-12,-3.1514006E-12, 00020650
5 3.3297565E-12,-3.5179095E-12, 3.7163306E-12,-3.9256378E-12, 00020660
6 4.1464798E-12,-4.3794552E-12, 4.6252131E-12,-4.8845227E-12, 00020670
7 5.1582809E-12,-5.4474462E-12, 5.7530277E-12,-6.0760464E-12, 00020680
8 6.4175083E-12,-6.7783691E-12, 7.1595239E-12,-7.5618782E-12, 00020690
9 7.9864477E-12,-8.4344110E-12, 8.9072422E-12,-9.4067705E-12, 00020700
1 9.9349439E-12,-1.0493731E-11, 1.1084900E-11,-1.1709937E-11, 00020710
2 1.2370354E-11,-1.3067414E-11, 1.3802200E-11,-1.4575980E-11, 00020720
3 1.5390685E-11,-1.6249313E-11, 1.7155934E-11,-1.8115250E-11, 00020730
4 1.9131898E-11,-2.0209795E-11, 2.1352159E-11,-2.2561735E-11, 00020740
5 2.3840976E-11,-2.5192263E-11, 2.6618319E-11,-2.8122547E-11, 00020750
6 2.9709129E-11,-3.1382870E-11, 3.3149030E-11,-3.5013168E-11, 00020760
7 3.6981050E-11,-3.9058553E-11, 4.1251694E-11,-4.3566777E-11, 00020770
8 4.6010537E-11,-4.8590396E-11, 5.1314761E-11,-5.4193353E-11, 00020780
9 5.7236720E-11,-6.0455911E-11, 6.3861222E-11,-6.7461492E-11, 00020790
1 7.1265224E-11,-7.5279775E-11, 7.9512249E-11,-8.3971327E-11/ 00020800
DATA W2/                                               00020810
1 8.8668961E-11,-9.3621900E-11, 9.8851764E-11,-1.0438319E-10, 00020820
2 1.1024087E-10,-1.1644680E-10, 1.2301979E-10,-1.2997646E-10, 00020830
3 1.3733244E-10,-1.4510363E-10, 1.5330772E-10,-1.6196550E-10, 00020840
4 1.7110130E-10,-1.8074257E-10, 1.9091922E-10,-2.0166306E-10, 00020850
5 2.1300756E-10,-2.2498755E-10, 2.3763936E-10,-2.5100098E-10, 00020860
6 2.6511250E-10,-2.8001616E-10, 2.9575691E-10,-3.1238237E-10, 00020870
7 3.2994314E-10,-3.4849209E-10, 3.6808529E-10,-3.8878042E-10, 00020880
8 4.1063982E-10,-4.3372666E-10, 4.5811059E-10,-4.8386049E-10, 00020890
9 5.1105728E-10,-5.3977672E-10, 5.7011632E-10,-6.0215516E-10, 00020900
1 6.3601273E-10,-6.7175964E-10, 7.0955028E-10,-7.4942601E-10, 00020910
2 7.9161025E-10,-8.3606980E-10, 8.8317110E-10,-9.3270330E-10, 00020920
3 9.8533749E-10,-1.0404508E-09, 1.0993731E-09,-1.1605442E-09, 00020930
4 1.2267391E-09,-1.2942905E-09, 1.3691677E-09,-1.4429912E-09, 00020940
5 1.5288164E-09,-1.6077524E-09, 1.7085998E-09,-1.7890471E-09, 00020950
6 1.9129068E-09,-1.9857116E-09, 2.1491608E-09,-2.1926779E-09, 00020960
7 2.4312660E-09,-2.3959044E-09, 2.7872500E-09,-2.5610596E-09, 00020970
8 3.2762318E-09,-2.6082940E-09, 4.0261453E-09,-2.3560563E-09, 00020980
9 5.3176554E-09,-1.3960161E-09, 7.7708747E-09, 1.1853546E-09, 00020990
1 1.2760851E-08, 7.4264707E-09, 2.3342187E-08, 2.1869851E-08/ 00021000
DATA W3/                                               00021010
1 4.6306744E-08, 5.4631686E-08, 9.6763087E-08, 1.2823337E-07, 00021020
2 2.0832812E-07, 2.9280540E-07, 4.5580888E-07, 6.5992437E-07, 00021030
3 1.0056815E-06, 1.4779183E-06, 2.2284335E-06, 3.2994604E-06, 00021040
00021050
  
```

```

4 4.9485823E-06, 7.3545473E-06, 1.1001083E-05, 1.6380539E-05, 00021060
5 2.4469550E-05, 3.6469246E-05, 5.4441527E-05, 8.1176726E-05, 00021070
6 1.2113828E-04, 1.8066494E-04, 2.6954609E-04, 4.0202288E-04, 00021080
7 5.9969995E-04, 8.9437312E-04, 1.3338166E-03, 1.9886697E-03, 00021090
8 2.9643943E-03, 4.4168923E-03, 6.5773518E-03, 9.7855105E-03, 00021100
9 1.4539361E-02, 2.1558670E-02, 3.1871864E-02, 4.6903518E-02, 00021110
1 6.8559512E-02, 9.9170152E-02, 1.4120770E-01, 1.9610835E-01, 00021120
2 2.6192603E-01, 3.2743321E-01, 3.6407406E-01, 3.1257559E-01, 00021130
3 9.0460168E-02, -3.6051039E-01, -8.6324760E-01, -8.1178720E-01, 00021140
4 5.2205241E-01, 1.5449873E+00, -1.1817933E+00, -2.6759896E-01, 00021150
5 8.0869203E-01, -6.2757149E-01, 3.4062630E-01, -1.5885304E-01, 00021160
6 7.0472984E-02, -3.1624462E-02, 1.4894068E-02, -7.4821176E-03, 00021170
7 4.0035936E-03, -2.2543784E-03, 1.3160358E-03, -7.8636604E-04, 00021180
8 4.7658745E-04, -2.9125817E-04, 1.7885105E-04, -1.1012416E-04, 00021190
9 6.7910334E-05, -4.1914054E-05, 2.5881544E-05, -1.5985851E-05, 00021200
1 9.8751880E-06, -6.1008526E-06, 3.7692543E-06, -2.3287953E-06/ 00021210
  DATA W4/ 00021220
1 1.4388425E-06, -8.8899353E-07, 5.4926991E-07, -3.3937048E-07, 00021230
2 2.0968284E-07, -1.2955437E-07, 8.0046336E-08, -4.9457371E-08, 00021240
3 3.0557711E-08, -1.8880390E-08, 1.1665454E-08, -7.2076428E-09, 00021250
4 4.4533423E-09, -2.7515696E-09, 1.7001092E-09, -1.0504494E-09, 00021260
5 6.4904567E-10, -4.0102999E-10, 2.4778763E-10, -1.5310321E-10, 00021270
6 9.4600354E-11, -5.8453314E-11, 3.6119400E-11, -2.2320056E-11, 00021280
7 1.3793460E-11, -8.5242656E-12, 5.2675102E-12, -3.2543076E-12, 00021290
8 2.0097689E-12, -1.2405412E-12, 7.6530538E-13, -4.7191929E-13, 00021300
9 2.9084993E-13, -1.7923661E-13, 1.1018948E-13, -6.7885902E-14, 00021310
1 4.2025050E-14, -2.1314731E-14/ 00021320
C---##ENDATA 00021330
C 00021340
  IF(NEW) 10,30,10 00021350
10 LAG=-1 00021360
  X0=-X-38.30455704 00021370
  DO 20 IR=1,266 00021380
20 KEY(IR)=0 00021390
30 LAG=LAG+1 00021400
  ZLAGF1=(0.0,0.0) 00021410
  CMAX=(0.0,0.0) 00021420
  L=0 00021430
  ASSIGN 110 TO M 00021440
  I=191 00021450
  GO TO 200 00021460
110 TMAX(1)=AMAX1(ABS(T(1)),TMAX(1)) 00021470
  TMAX(2)=AMAX1(ABS(T(2)),TMAX(2)) 00021480
  I=I+1 00021490
  IF(I.LE.208) GO TO 200 00021500
  IF(TMAX(1).EQ.0.0.AND.TMAX(2).EQ.0.0) GO TO 150 00021510
  CMAX=TOL*CMAX 00021520
  ASSIGN 120 TO M 00021530
  I=190 00021540
  GO TO 200 00021550
120 IF(ABS(T(1)).LE.TMAX(1).AND.ABS(T(2)).LE.TMAX(2)) GO TO 130 00021560
  I=I-1 00021570

```

```

130 IF(I.GT.0) GO TO 200                                00021580
    ASSIGN 140 TO M                                    00021590
    I=209                                              00021600
    GO TO 200                                          00021610
140 IF(ABS(T(1)).LE.TMAX(1).AND.ABS(T(2)).LE.TMAX(2)) GO TO 190 00021620
    I=I+1                                              00021630
    IF(I.LE.266) GO TO 200                             00021640
    GO TO 190                                          00021650
150 ASSIGN 160 TO M                                    00021660
    I=1                                                00021670
    GO TO 200                                          00021680
160 IF(T(1).EQ.0.0.AND.T(2).EQ.0.0) GO TO 170        00021690
    I=I+1                                              00021700
    IF(I.LE.190) GO TO 200                             00021710
170 ASSIGN 180 TO M                                    00021720
    I=266                                              00021730
    GO TO 200                                          00021740
180 IF(T(1).EQ.0.0.AND.T(2).EQ.0.0) GO TO 190        00021750
    I=I-1                                              00021760
    IF(I.GE.209) GO TO 200                             00021770
190 RETURN                                             00021780
C--STORE/RETRIEVE ROUTINE (DONE INTERNALLY TO SAVE CALL'S) 00021790
200 LOOK=I+LAG                                        00021800
    IQ=LOOK/267                                        00021810
    IR=MOD(LOOK,267)                                   00021820
    IF(IR.EQ.0) IR=1                                  00021830
    IROLL=IQ*266                                       00021840
    IF(KEY(IR).LE.IROLL) GO TO 220                     00021850
210 C=SAVE(IR)*WT(I)                                   00021860
    ZLAGF1=ZLAGF1+C                                    00021870
    L=L+1                                              00021880
    GO TO M,(110,120,140,160,180)                     00021890
220 KEY(IR)=IROLL+IR                                  00021900
    SAVE(IR)=FUN(EXP(X0+FLOAT(LOOK)*.20))              00021910
    GO TO 210                                          00021920
    END                                                00021930

    SUBROUTINE IKS(B8,I1K1,IKDIF)                       00021940
C--COMPUTE MODIFIED BESSEL FUNCTION (I & K) SPECIAL COMBINATIONS FOR 00021950
C PARAMETERS                                           00021960
C   B8      = DOUBLE PRECISION ARGUMENT (=B/DSQRT(2.DO) HERE) 00021970
C   I1K1    = I1*K1 COMPLEX RESULT                    00021980
C   IKDIF   = 4*I1*K1-(B8*DSQRT(I))*(I0*K1-I1*K0) COMPLEX RESULT DONE IN 00021990
C             DP BEFORE CMPLX"ING.                    00022000
C--SUBROUTINE KELVIN CALLED                             00022010
C                                                       00022020
    DOUBLE PRECISION B8,BB(8),BETA,Q1,Q2,R1,R2        00022030
    COMPLEX I1K1,IKDIF,CAMBDA,DENOM,DENOM1,TERMO,TERM1,TERM11 00022040
    COMPLEX S11,S10,S11,SK0,SK1,ONE                  00022050
    DATA ONE/(1.0,0.0)/                              00022060
    IF(B8.GT.20.DO) GO TO 10                           00022070
    CALL KELVIN(B8,8,BB)                                00022080

```

```
00022090 Q1=-BB(6)*BB(8)+BB(5)*BB(7)
00022100 Q2= BB(5)*BB(8)+BB(6)*BB(7)
00022110 IK1=CMLX(SNGL(Q1),SNGL(Q2))
00022120 R1=-BB(1)*BB(8)-BB(2)*BB(7) -
00022130 BB(6)*BB(3)-BB(5)*BB(4)
00022140 R2=-BB(2)*BB(8)+BB(1)*BB(7) +
00022150 BB(5)*BB(3)-BB(6)*BB(4)
00022160 BETA=.7071067811865475D0*BB
00022170 Q1=4.0D0*Q1-BETA*(R1-R2)
00022180 Q2=4.0D0*Q2-BETA*(R1+R2)
00022190 IKDIF=CMLX(SNGL(Q1),SNGL(Q2))
00022200 RETURN
00022210 10 B=SNGL(BB/0.7071067811865475D0)
00022220 TOL=1.E-6
00022230 C--FOR LARGE ARGUMENTS, USE ARRAMOWITZ AND STEGUN
00022240 C ASYMPTOTIC FORMULAS FOR LARGE ARGUMENTS
00022250 C 9.7.1 THROUGH 9.7.5, P. 377-378.
00022260 CAMBDA=B*CMLX(1.0+1.0)/2.
00022270 IKDIF=CMLX(100.0,0.)
00022280 ISIGN=1
00022290 DENOM=8.*CAMBDA
00022300 DENOM1=(2.*CAMBDA)**2
00022310 NDD=1
00022320 TERM0=ONE
00022330 TERM1=ONE
00022340 TERM11=ONE
00022350 S11=ONE
00022360 S10=ONE
00022370 S11=ONE
00022380 SK0=ONE
00022390 SK1=ONE
00022400 NDD2=NDD*NDD
00022410 OIKDIF=CABS(IKDIF)
00022420 TERM1=TERM1*CMLX(4.-NDD2,0.)/DENOM
00022430 TERM0=TERM0*CMLX(-FLOAT(NDD2),0.)/DENOM
00022440 TERM11=TERM11*CMLX(NDD*(4.-NDD2)/(NDD+1),0.)/DENOM1
00022450 ISIGN=-ISIGN
00022460 S11=S11+ISIGN*TERM11
00022470 S10=S10+ISIGN*TERM0
00022480 S11=S11+ISIGN*TERM1
00022490 SK0=SK0+TERM0
00022500 SK1=SK1+TERM1
00022510 IKDIF=S10*SK1-SK0*S11
00022520 NDD=NDD+2
00022530 IF (ABS(OIKDIF-CABS(IKDIF)).GT.TOL) GO TO 1
00022540 IK1=S11/(CAMBDA*CMLX(2.0,0.))
00022550 IKDIF=CMLX(4.0,0.)*IK1-IKDIF/CMLX(2.0,0.)
00022560 RETURN
00022570 END
00022580 SUBROUTINE KELVIN(X;M;B)
00022590 C--COMPUTES W(LE;B) KELVIN FUNCTIONS (ORDERS 0,1) CONSECUTIVELY STORED
```

```
C IN ARRAY B(M) WHERE:
C
C X = DF-ARGUMENT, GT, 0.0D0 (ASYMPTOTIC FORM USED IF X,GE,8.0)
C M = NUMBER OF B'S TO COMPUTE AS DEFINED BELOW (1,GE,M,LE,8)
C B(M) = COMPUTED DF-FUNCTIONS WHERE B IS DEFINED:
C B(1) = BER(X) --- ORDER 0
C B(2) = BEI(X) --- ORDER 0
C B(3) = KER(X) --- ORDER 0
C B(4) = KEI(X) --- ORDER 0
C B(5) = BER1(X) --- ORDER 1
C B(6) = BEI1(X) --- ORDER 1
C B(7) = KER1(X) --- ORDER 1
C B(8) = KEI1(X) --- ORDER 1
C ** ACCURACY GOOD TO AT LEAST 14 FIGURES FOR ALL X **
C NOTE: THIS METHOD OF GENERATING MULTIPLE KELVIN FUNCTIONS WAS CHOSEN
C TO REDUCE TOTAL CPU-TIME SINCE MOST APPLICATIONS REQUIRE
C MULTIPLE FUNCTION USE AND IS THEREFORE ACCOMPLISHED BY ONE CALL.
C E.G: TO OBTAIN BER(X),BEI(X),KER(X) AND KEI(X); CALL KELVIN(X,4,B)
C IF X OR M OUT OF RANGE, ROUTINE EXITS WITHOUT ACTION.
C IMPLICIT REAL*8 (A-H,O-Z)
C REAL*8 B(8),CN(8),SN(8)
C DATA CN /,7071067811865475D0,0.0D0,-,7071067811865475D0,
* -1.0D0,-,7071067811865475D0,0.0D0,7071067811865475D0,1.0D0/,
* SN /,7071067811865475D0,1.0D0,7071067811865475D0,0.0D0,
* -,7071067811865475D0,-1.0D0,-,7071067811865475D0,0.0D0/
C DATA P14/,7853981633974483D0/,R22/,7071067811865475D0/,
* E/O,5D-14/,
* P11/,3183098861837907D0/
C IF(M,LT,1,OR,M,GT,8,OR,X,LE,0.0D0) GO TO 9
C IF(X,GE,8.0D0) GO TO 8
C--SERIES METHODS (X,GT,0.0,AND,X,LT,8.0D0)
X2=0.5D0*X
X4=X2**4
T1=-0.25D0*X4
S1=T1
T2=0.0D0
T3=0.0D0
T4=0.0D0
T5=0.0D0
T6=0.0D0
T7=0.0D0
T8=0.0D0
T9=0.0D0
T10=0.0D0
T11=0.0D0
T12=0.0D0
T13=0.0D0
T14=0.0D0
T15=0.0D0
T16=0.0D0
T17=0.0D0
T18=0.0D0
IF(M,EG,1) GO TO 100
T2=X2**2
S2=T2
IF(M,EG,2) GO TO 100
T5=1.5D0
S5=T1*T5
IF(M,EG,3) GO TO 100
T9=1.0D0
S9=T2
S9=T2
00022600
00022610
00022620 = DF-ARGUMENT, GT, 0.0D0 (ASYMPTOTIC FORM USED IF X,GE,8.0)
00022630 = NUMBER OF B'S TO COMPUTE AS DEFINED BELOW (1,GE,M,LE,8)
00022640 B(M) = COMPUTED DF-FUNCTIONS WHERE B IS DEFINED:
00022650 B(1) = BER(X) --- ORDER 0
00022660 B(2) = BEI(X) --- ORDER 0
00022670 B(3) = KER(X) --- ORDER 0
00022680 B(4) = KEI(X) --- ORDER 0
00022690 B(5) = BER1(X) --- ORDER 1
00022700 B(6) = BEI1(X) --- ORDER 1
00022710 B(7) = KER1(X) --- ORDER 1
00022720 B(8) = KEI1(X) --- ORDER 1
00022730 C ** ACCURACY GOOD TO AT LEAST 14 FIGURES FOR ALL X **
00022740 C NOTE: THIS METHOD OF GENERATING MULTIPLE KELVIN FUNCTIONS WAS CHOSEN
00022750 C TO REDUCE TOTAL CPU-TIME SINCE MOST APPLICATIONS REQUIRE
00022760 C MULTIPLE FUNCTION USE AND IS THEREFORE ACCOMPLISHED BY ONE CALL.
00022770 C E.G: TO OBTAIN BER(X),BEI(X),KER(X) AND KEI(X); CALL KELVIN(X,4,B)
00022780 C IF X OR M OUT OF RANGE, ROUTINE EXITS WITHOUT ACTION.
00022790 C IMPLICIT REAL*8 (A-H,O-Z)
00022800 C REAL*8 B(8),CN(8),SN(8)
00022810 C DATA CN /,7071067811865475D0,0.0D0,-,7071067811865475D0,
00022820 * -1.0D0,-,7071067811865475D0,0.0D0,7071067811865475D0,1.0D0/,
00022830 * SN /,7071067811865475D0,1.0D0,7071067811865475D0,0.0D0,
00022840 * -,7071067811865475D0,-1.0D0,-,7071067811865475D0,0.0D0/
00022850 C DATA P14/,7853981633974483D0/,R22/,7071067811865475D0/,
00022860 * E/O,5D-14/,
00022870 * P11/,3183098861837907D0/
00022880 C IF(M,LT,1,OR,M,GT,8,OR,X,LE,0.0D0) GO TO 9
00022890 C IF(X,GE,8.0D0) GO TO 8
00022900 C--SERIES METHODS (X,GT,0.0,AND,X,LT,8.0D0)
00022910 X2=0.5D0*X
00022920 X4=X2**4
00022930 T1=-0.25D0*X4
00022940 S1=T1
00022950 T2=0.0D0
00022960 T3=0.0D0
00022970 T4=0.0D0
00022980 T5=0.0D0
00022990 T6=0.0D0
00023000 T7=0.0D0
00023010 T8=0.0D0
00023020 T9=0.0D0
00023030 T10=0.0D0
00023040 T11=0.0D0
00023050 T12=0.0D0
00023060 T13=0.0D0
00023070 T14=0.0D0
00023080 T15=1.5D0
00023090 S5=T1*T5
00023100 IF(M,EG,3) GO TO 100
00023110 T9=1.0D0
00023120 S9=T2
00023130 S9=T2
00023140
```

IF(M.EQ.4) GO TO 100	00023120
T3=-0.5D0*X2**3	00023130
S3=T3	00023140
T4=X2	00023150
S4=T4	00023160
IF(M.LE.6) GO TO 100	00023170
T7=-0.25D0*X2**3	00023180
S7=2.0D0*T7*T5	00023190
T8=X2	00023200
S8=T8	00023210
100 TK=2.0D0	00023220
101 TK2=TK+TK	00023230
TK21=TK2-1.0D0	00023240
TK22=TK2-2.0D0	00023250
RK2=1.0D0/TK2	00023260
RK21=1.0D0/TK21	00023270
RK22=1.0D0/TK22	00023280
R1=-X4*(RK21**RK2)**2	00023290
T1=T1*R1	00023300
S1=S1+T1	00023310
IF(M.EQ.1) GO TO 200	00023320
R2=-X4*(RK22**RK21)**2	00023330
T2=T2*R2	00023340
S2=S2+T2	00023350
IF(M.EQ.2) GO TO 200	00023360
T5=T5+RK21+RK2	00023370
T15=T1*T5	00023380
S5=S5+T15	00023390
IF(M.EQ.3) GO TO 200	00023400
T6=T6+RK22+RK21	00023410
T26=T2*T6	00023420
S6=S6+T26	00023430
IF(M.EQ.4) GO TO 200	00023440
T3=T3*(-X4*(RK22**RK21**2**RK2))	00023450
S3=S3+T3	00023460
T4=T4*(-X4**RK22**2**RK21/(TK2-3.0D0))	00023470
S4=S4+T4	00023480
IF(M.LE.6) GO TO 200	00023490
T7=T7*R1	00023500
T75=TK2*T7*T5	00023510
S7=S7+T75	00023520
T8=T8*R2	00023530
T86=TK21*T8*T6	00023540
S8=S8+T86	00023550
200 TK=TK+1.0D0	00023560
IF(DABS(T1).GT.E.OR.DABS(T2).GT.E.OR.DABS(T15).GT.E.OR.	00023570
* DABS(T26).GT.E.OR.DABS(T3).GT.E.OR.DABS(T4).GT.E.OR.	00023580
* DABS(T75).GT.E.OR.DABS(T86).GT.E) GO TO 101	00023590
B(1)=1.0D0+S1	00023600
IF(M.EQ.1) GO TO 9	00023610
B(2)=S2	00023620
IF(M.EQ.2) GO TO 9	00023630





```

FM=FM+T2
T3=TP*SN(N)
GP=GP+T3
T4=TM*SN(N)
GM=GM+T4
K=K+1
IF(K.GT.MAXK) GO TO 3
GO TO 2
21 FP=FP-T1
FM=FM-T2
GP=GP-T3
GM=GM-T4
3 FP=FP+1.0D0
FM=FM+1.0D0
B(N4+4)=C1*(-FM*SB-GM*CB)
B(N4+3)=C1*(FM*CB-GM*SB)
B(N4+2)=C2*(FP*SA-GP*CA)+PI1*B(N4+3)
B(N4+1)=C2*(FP*CA+GP*SA)-PI1*B(N4+4)
IF(NU.EQ.1.OR.M.LE.4) GO TO 9
NU=1
GO TO 1
END

```

00024160  
 00024170  
 00024180  
 00024190  
 00024200  
 00024210  
 00024220  
 00024230  
 00024240  
 00024250  
 00024260  
 00024270  
 00024280  
 00024290  
 00024300  
 00024310  
 00024320  
 00024330  
 00024340  
 00024350  
 00024360  
 00024370

```

SUBROUTINE QUINT(NY,Y,B,C,D,E,F)
C---COMPUTES COEFFICIENTS OF A QUINTIC NATURAL SPLINE S(X) GIVEN
C THE ORDINATES Y(I) AT ASSUMED EQUIDISTANT POINTS X(I),I=1 TO NY.
C
C TRANSLATED FROM ALGOL TO FORTRAN BY
C W.L. ANDERSON, U.S. GEOLOGICAL SURVEY, DENVER, COLORADO.
C REF: ACM TRANSACTIONS ON MATH. SOFTWARE, SEPT 1976, V.2, N. 3,
C PP.281-289.
C
C PARAMETERS:
C
C NY = NUMBER OF DATA POINTS GIVEN IN Y(NY), NY.GT.2.
C Y()= ARRAY OF NY GIVEN ORDINATES (DIM.GE.NY).
C Y() POINTS ASSUMED EQUALLY SPACED IN X-DIRECTION.
C B,C,D,E,F() = RESULTING ARRAYS (EACH DIM.GE.NY) OF
C QUINTIC SPLINE COEFFICIENTS, WHERE
C FOR ANY XX IN (X(I),X(I+1)):
C S(XX)=((((F(I)*T+E(I))*T+D(I))*T+C(I))*T+B(I))*T+Y(I) WITH
C T=(XX-X(I))/DELX, DELX=(X(I+1)-X(I)) FOR ANY I.
C NOTE: SEE PROC 'QPOINT' TO EVAL THE QUINTIC SPLINE AFTER
C 'QUINT' IS CALLED.
C
C
C DIMENSION Y(1),B(1),C(1),D(1),E(1),F(1)
C IF(NY.LE.2) GO TO 4
C N=NY-3
C P=0.0
C Q=0.0
C R=0.0
C S=0.0

```

00024380  
 00024390  
 00024400  
 00024410  
 00024420  
 00024430  
 00024440  
 00024450  
 00024460  
 00024470  
 00024480  
 00024490  
 00024500  
 00024510  
 00024520  
 00024530  
 00024540  
 00024550  
 00024560  
 00024570  
 00024580  
 00024590  
 00024600  
 00024610  
 00024620  
 00024630  
 00024640  
 00024650  
 00024660

```
00024670 T=0.0  
00024680 DO 1 I=1,N  
00024690 U=F*R  
00024700 B(I)=1.0/(66.0-U*R-Q)  
00024710 R=26.0-U  
00024720 C(I)=R  
00024730 D(I)=(Y(I+3)-3.0*(Y(I+2)-Y(I+1))-Y(I))-U*S-Q*T  
00024740 Q=F  
00024750 P=B(I)  
00024760 T=S  
00024770 S=D(I)  
00024780 CONTINUE  
00024790 D(N+2)=0.0  
00024800 N1=N+1  
00024810 D(N1)=0.0  
00024820 DO 2 J=1,N  
00024830 I=N1-J  
00024840 D(I)=(D(I)-C(I))*D(I+1)-D(I+2))*B(I)  
00024850 CONTINUE  
00024860 N=N1-1  
00024870 Q=0.0  
00024880 V=D(I)  
00024890 T=V  
00024900 R=V  
00024910 DO 3 I=2,N  
00024920 F=Q  
00024930 G=R  
00024940 R=D(I)  
00024950 S=T  
00024960 T=P-Q-Q+R  
00024970 F(I)=T  
00024980 U=S.0*(-F+Q)  
00024990 E(I)=U  
00025000 D(I)=10.0*(F+Q)  
00025010 C(I)=0.5*(Y(I+1)+Y(I-1))+S-T)-Y(I)-U  
00025020 B(I)=0.5*(Y(I+1)-Y(I-1))-S-T)-D(I)  
00025030 CONTINUE  
00025040 F(I)=V  
00025050 E(I)=0.0  
00025060 E(N1)=0.0  
00025070 D(I)=0.0  
00025080 D(N1)=0.0  
00025090 C(I)=C(2)-10.0*K  
00025100 C(N1)=C(N1)+10.0*K  
00025110 B(1)=Y(2)-Y(1)-C(1)-V  
00025120 B(N1)=Y(N1)-Y(N1-1)+C(N1)-T  
00025130 RETURN  
00025140 END  
00025150 SUBROUTINE GPOINT(N1,Y,B,C,D,E,F,X1,DELX,XX,YY)  
00025160 C GIVEN THE QUINTIC SPLINE COEFF'S B(*),C(*),D(*),E(*),F(*) AS  
00025170 C OBTAINED FROM SUBR, GUNIT, AND GIVEN NY OBS, DATA Y(NY) EQUALLY
```

```

C SPACED BY DELX STARTING AT X1, THEN 'QPOINT' INTERPOLATES 00025180
C YY AT ANY XX IN (X1,X1+(NY-1)*DELX). 00025190
C 00025200
C   DIMENSION Y(1),B(1),C(1),D(1),E(1),F(1) 00025210
C   XMAX=X1+(NY-1)*DELX 00025220
C   IF(XX.LT.X1.OR.XX.GT.XMAX) GO TO 2 00025230
C   I=(XX-X1)/DELX+1 00025240
C   XI=X1+(I-1)*DELX 00025250
C   T=(XX-XI)/DELX 00025260
C   YY=(((F(I)*T+E(I))*T+D(I))*T+C(I))*T+B(I))*T+Y(I) 00025270
1   RETURN 00025280
2   WRITE(6,3) XX,X1,XMAX 00025290
3   FORMAT('QPOINT ERROR--- XX=',E16.8,' NOT IN CLOSED INTERVAL (' 00025300
C   & E16.8,' ',E16.8,')') 00025310
C   GO TO 1 00025320
C   END 00025330

C   COMPLEX FUNCTION CANC4(A1,B1,EP,M,N,FUN,MF,ESUM) 00025340
C--COMPLEX FUNCTION DEFINITE INTEGRATION BY 00025350
C   ADAPTIVE QUADRATURE USING NEWTON-COTES NO. 4 (N=1,2,3), OR 00025360
C   AUTOMATIC GAUSSIAN QUADRATURE (N=4,5)..... 00025370
C   A GENERAL ROUTINE IN SINGLE-PRECISION COMPLEX... 00025380
C   BY W.L.ANDERSON, U.S.GEOLOGICAL SURVEY, DENVER, COLORADO. 00025390
C--PARAMETERS-- 00025400
C   A1 = LOWER LIMIT OF INTEGRATION (REAL*4) 00025410
C   B1 = UPPER LIMIT OF INTEGRATION (REAL*4) 00025420
C   EP = DESIRED REL. ERROR (REAL*4) IN COMPLEX RESULT 'CANC4'. 00025430
C   (FOR BOTH RE & IM PARTS OF CANC4) 00025440
C   M = RESULTING NUMBER OF COMPLEX 'FUN EVALUATIONS' 00025450
C   N = ERROR TEST TYPE: 00025460
C   = 1 FOR ABS. ERROR TEST (NOT GENERALLY RECOMMENDED) 00025470
C   = 2 FOR 'L-ONE' ERROR TEST 00025480
C   = 3 FOR 'L-INFINITY' ERROR TEST 00025490
C   = 4 FOR REL. ERROR TEST USING ADAPTIVE GAUSSIAN QUADRATURE 00025500
C   = 5 FOR REL. ERROR TEST USING NON-ADAPTIVE GAUSSIAN QUAD... 00025510
C   NOTE: N=1,2,OR 3 USES ADAPTIVE NEWTON-COTES QUADRATURE, AND 00025520
C   N=4 OR 5 USES ADAPTIVE OR NON-ADAPTIVE GAUSS QUADRATURES 00025530
C   FUN = EXTERNAL COMPLEX FUNCTION NAME (COMPLEX*8) 00025540
C   MF = MAX. FUN EVALUATIONS ALLOWED BEFORE ACCEPTING COMPLEX 00025550
C   RESULT 'CANC4' WITH M.GE.MF.... 00025560
C   ESUM = ACTUAL COMPLEX ERROR ACHIEVED AT EXIT.... 00025570
C--SUBPROGRAMS CALLED: CQSUBA,CQSUB (WHICH CALLS CQUAD) 00025580
C   (THESE ARE FOR N=4 OR 5 GAUSSIAN QUADRATURES) 00025590
C 00025600
C   COMPLEX FUN,ESUM,TSUM,FA, F,X,Z, CQSUBA,CQSUB, 00025610
1 F1,FS,F3,FM,F2,FT,F4,FB,FTP,FBP,FMAX,FTST,EST,AEST,EST1,EST2,AEST 00025620
21,AEST2,ABSAR,DELTA,DIFF,DAFT,SUM 00025630
C   DIMENSION F2(30),F4(30),FTP(30),FBP(30),FTST(5),EST2(30),NRTR(30) 00025640
C   DIMENSION AEST2(30),XB(30) 00025650
C   DIMENSION FMX(2) 00025660
C   EXTERNAL FUN 00025670
C   EQUIVALENCE (FMX(1),FMAX) 00025680

```

C--STATEMENT FUNCTION GOES HERE ON HONEYWELL MULTICS SYSTEM

```
00025690 F(X)=CMPLX(ABS(REAL(X)),ABS(AIMAG(X)))  
00025700 THE PARAMETER SETUP FOR THE INITIAL CALL  
00025710 IF(N.LE.0)GO TO 210  
00025720 IF(N.GT.5)GO TO 211  
00025730 GO TO (10,10,10,10,400,500),N  
00025740 A=A1  
00025750 B=B1  
00025760 EPS=EP*63.  
00025770 ESUM=(0.0,0.0)  
00025780 TSUM=(0.0,0.0)  
00025790 LVL=1  
00025800 DA=B-A  
00025810 FA=FUN(A)  
00025820 FS=FUN((3.0 *A+B)/4.0 )  
00025830 FM=FUN((A+B)*0.5 )  
00025840 FT=FUN((A+3.0 *B)/4.0 )  
00025850 FB=FUN(B)  
00025860 M=5  
00025870 FMAX=F(A)  
00025880 FST(1)=FMAX  
00025890 FST(2)=F(FS)  
00025900 FST(3)=F(FM)  
00025910 FST(4)=F(FT)  
00025920 FST(5)=F(FB)  
00025930 DO 100 I=2,5  
00025940 IF(FMX(1).GE.REAL(FST(I)))GO TO 101  
00025950 IF(FMX(1).GE.REAL(FST(I)))GO TO 101  
00025960 FMX(1)=REAL(FST(I))  
00025970 IF(FMX(2).GE.AIMAG(FST(I)))GO TO 100  
00025980 FMX(2)=AIMAG(FST(I))  
00025990 CONTINUE  
00026000 EST=(7.0 *(F+FB)+32.0 *(FS+FT)+12.0 *FM)*DA/90.0  
00026010 ABSAR=(7.0 *(FST(1)+FST(5))+32.0 *(FST(2)+FST(4))+12.0 *FST(3))*DA/90.0  
00026020 IT(3)*DA/90.0  
00026030 AEST=ABSAR  
00026040 I=RECUR  
00026050 SX=(DA/(2.0 *LVL))/90.0  
00026060 F1=FUN((7.0 *A+B)/8.0 )  
00026070 F3=FUN((5.0 *A+3.0 *B)/8.0 )  
00026080 F2=LVL)=FUN((3.0 *A+5.0 *B)/8.0 )  
00026090 F4(LVL)=FUN((A+7.0 *B)/8.0 )  
00026100 EST1=SX*(7.0 *(F+FM)+32.0 *(F1+F3)+12.0 *FS)  
00026110 FFB(LVL)=FB  
00026120 FTP(LVL)=FT  
00026130 XB(LVL)=B  
00026140 EST2(LVL)=SX*(7.0 *(FM+FB)+32.0 *(F2(LVL)+F4(LVL))+12.0 *FT)  
00026150 SUM=EST1+EST2(LVL)  
00026160 FST(1)=F(FT)  
00026170 FST(2)=F(F2(LVL))  
00026180 FST(3)=F(F3)  
00026190 FST(4)=F(F4(LVL))  
00026200 FST(5)=F(FM)
```

```

00026210 AEST1=SX*(7.0 *(F(FA) +FTST(5))+32.0 *(FTST(1)+FTST(3))+12.0
00026220 X*(FS))
00026230 AEST2(LVL)=SX*(7.0 *(FTST(5)+F(FB) )+32.0 *(FTST(2)+FTST(4))+10026230
00026240 ABSAR=ABSAR-AEST1+AEST2(LVL)
00026250 M=M+4
00026260 IF(M*GE*MF) GO TO 5
00026270 GO TO (201,200,202)*N
00026280 DELTA=ABSAR
00026290 GO TO 205
00026300 WRITE(6,39)
00026310 FORMAT(' CANCA- ERROR RETURN-N,LE.0')
00026320 GO TO 999
00026330 WRITE(6,40)
00026340 FORMAT(' CANCA- ERROR RETURN-N,GT.5')
00026350 GO TO 999
00026360 DELTA=(1.0,1.0)
00026370 GO TO 205
00026380 DO 203 I=1,4
00026390 IF(FMX(1).GE.REAL(FTST(I)))GO TO 2031
00026400 FMX(1)=REAL(FTST(I))
00026410 IF(FMX(2).GE.AIMAG(FTST(I)))GO TO 2031
00026420 FMX(2)=AIMAG(FTST(I))
00026430 CONTINUE
00026440 DELTA=FMX
00026450 DAF1=EST-SUM
00026460 DIFF=F(DAF1)
00026470 DAF1=DAF1/63.0
00026480 Z=DIFF-EFS*DELTA
00026490 IF(REAL(Z).LE.0.0.AND.AIMAG(Z).LE.0.0) GO TO 6
00026500 IF(LVL-30)4,2,2
00026510 IF(LVL-1)2,4,2
00026520 Z=UP
00026530 A=H
00026540 ESUM=ESUM+DAF1
00026550 TSUM=TSUM+SUM
00026560 LVL=LVL-1
00026570 L=NRTN(LVL)
00026580 GO TO (11,12)*L
00026590 I1=R1,I2=R2
00026600 NRTN(LVL)=1
00026610 EST=EST1
00026620 AEST=AEST1
00026630 FB=FM
00026640 FT=F3
00026650 FM=FS
00026660 FS=F1
00026670 B=(A+B)/2.0
00026680 EFS=EFS/2.0
00026690 LVL=LVL+1
00026700 GO TO 1
00026710 NRTN(LVL)=2
00026720

```

```

FA=FB                                00026730
FS=F2(LVL)                            00026740
FM=FTP(LVL)                            00026750
FT=F4(LVL)                             00026760
FB=FBF(LVL)                            00026770
B=XB(LVL)                              00026780
EST=EST2(LVL)                          00026790
AEST=AEST2(LVL)                        00026800
GO TO 7                                00026810
12 EPS=2.0 *EPS                         00026820
   IF(LVL-1)5,5,9                       00026830
5   CANC4=TSUM-ESUM                     00026840
   GO TO 999                             00026850
400  CANC4=CQSUBA(A1,B1,EP,M,ICK,ESUM,FUN,MF) 00026860
   GO TO 999                             00026870
500  CANC4=CQSUB(A1,B1,EP,M,ICK,ESUM,FUN,MF) 00026880
999  RETURN                              00026890
END                                     00026900

SUBROUTINE CQUAD(A,B,RESULT,K,EPSIL,NPTS,ICHECK,F,MEV) 00026910
C---MODIFIED BY W.L.ANDERSON FOR COMPLEX FUNCTIONS---12/28/73. 00026920
  COMPLEX F,RESULT,FUNCT,FZERO,ACUM      00026930
  DIMENSION FUNCT(127), P(381), RESULT(8) 00026940
C THIS SUBROUTINE ATTEMPTS TO CALCULATE THE INTEGRAL OF F(X) 00026950
C OVER THE INTERVAL *A* TO *B* WITH RELATIVE ERROR NOT 00026960
C EXCEEDING *EPSIL*. 00026970
C THE RESULT IS OBTAINED USING A SEQUENCE OF 1,3,7,15,31,63, 00026980
C 127, AND 255 POINT INTERLACING FORMULAE(NO INTEGRAND 00026990
C EVALUATIONS ARE WASTED) OF RESPECTIVE DEGREE 1,5,11,23, 00027000
C 47,95,191 AND 383. THE FORMULAE ARE BASED ON THE OPTIMAL 00027010
C EXTENSION OF THE 3-POINT GAUSS FORMULA. DETAILS OF 00027020
C THE FORMULAE ARE GIVEN IN *THE OPTIMUM ADDITION OF POINTS 00027030
C TO QUADRATURE FORMULAE* BY T.N.L. PATTERSON,MATHS.COMP. 00027040
C VOL 22,847-856,1968. 00027050
C                                     *** INPUT *** 00027060
C A      LOWER LIMIT OF INTEGRATION. 00027070
C B      UPPER LIMIT OF INTEGRATION. 00027080
C EPSIL  RELATIVE ACCURACY REQUIRED. WHEN THE RELATIVE 00027090
C        DIFFERENCE OF TWO SUCCESSIVE FORMULAE DOES NOT 00027100
C        EXCEED *EPSIL* THE LAST FORMULA COMPUTED IS TAKEN 00027110
C        AS THE RESULT. 00027120
C F      F(X) IS THE INTEGRAND. 00027130
C                                     *** OUTPUT *** 00027140
C RESULT THIS ARRAY,WHICH SHOULD BE DECLARED TO HAVE AT 00027150
C        LEAST 8 ELEMENTS, HOLDS THE RESULTS OBTAINED BY 00027160
C        THE 1,3,7, ETC., POINT FORMULAE. THE NUMBER OF 00027170
C        FORMULAE COMPUTED DEPENDS ON *EPSIL*. 00027180
C K      RESULT(K) HOLDS THE VALUE OF THE INTEGRAL TO THE 00027190
C        SPECIFIED RELATIVE ACCURACY. 00027200
C NPTS   NUMBER INTEGRAND EVALUATIONS. 00027210
C ICHECK ON EXIT NORMALLY ICHECK=0. HOWEVER IF CONVERGENCE 00027220
C        TO THE ACCURACY REQUESTED IS NOT ACHIEVED ICHECK=1 00027230

```





\* 0.98868475754742947994E 00,0.61155068221172463397E-02, 00027760  
\* 0.97218287474858179658E 00,0.10498246909621321898E-01, 00027770  
\* 0.94634285837340290515E 00,0.15406750466559497802E-01, 00027780  
\* 0.91037115695700429250E 00,0.20594233915912711149E-01, 00027790  
\* 0.86390793819369047715E 00,0.25869679327214746911E-01, 00027800  
\* 0.80694053195021761186E 00,0.31073551111687964880E-01, 00027810  
\* 0.73975604435269475868E 00,0.36064432780782572640E-01, 00027820  
\* 0.66290966002478059546E 00,0.40715510116944318934E-01, 00027830  
\* 0.57719571005204581484E 00,0.44914531653632197414E-01, 00027840  
\* 0.48361802694584102756E 00,0.48564330406673198716E-01/ 00027850

DATA

\* P( 85),P( 86),P( 87),P( 88),P( 89),P( 90),P( 91), 00027860  
\* P( 92),P( 93),P( 94),P( 95),P( 96),P( 97),P( 98), 00027880  
\* P( 99),P(100),P(101),P(102),P(103),P(104),P(105), 00027890  
\* P(106),P(107),P(108),P(109),P(110),P(111),P(112)/ 00027900  
\* 0.38335932419873034692E 00,0.51583253952048458777E-01, 00027910  
\* 0.27774982202182431507E 00,0.53905499335266063927E-01, 00027920  
\* 0.16823525155220746498E 00,0.55481404356559363988E-01, 00027930  
\* 0.56344313046592789972E-01,0.56277699831254301273E-01, 00027940  
\* 0.56377628360384717388E-01,0.16801938574103865271E-01, 00027950  
\* 0.64519000501757369228E-02,0.25078569652949768707E-01, 00027960  
\* 0.21088152457266328793E-02,0.11615723319955134727E-01, 00027970  
\* 0.21438980012503867246E-01,0.27394605263981432516E-01, 00027980  
\* 0.63260731936263354422E-03,0.41115039786546930472E-02, 00027990  
\* 0.89892757840641357233E-02,0.14244877372916774306E-01, 00028000  
\* 0.19219905124727766019E-01,0.23406777495314006201E-01, 00028010  
\* 0.26417473395058259931E-01,0.27989218255238159704E-01, 00028020  
\* 0.18073956444538835782E-03,0.12895240826104173921E-02, 00028030  
\* 0.30577534101755311361E-02,0.52491234548088591251E-02/ 00028040

DATA

\* P(113),P(114),P(115),P(116),P(117),P(118),P(119), 00028050  
\* P(120),P(121),P(122),P(123),P(124),P(125),P(126), 00028070  
\* P(127),P(128),P(129),P(130),P(131),P(132),P(133), 00028080  
\* P(134),P(135),P(136),P(137),P(138),P(139),P(140)/ 00028090  
\* 0.77033752332797418482E-02,0.10297116957956355524E-01, 00028100  
\* 0.12934839663607373455E-01,0.15536775555843982440E-01, 00028110  
\* 0.18032216390391286320E-01,0.20357755058472159467E-01, 00028120  
\* 0.22457265826816098707E-01,0.24282165203336599358E-01, 00028130  
\* 0.25791626976024229388E-01,0.26952749667633031963E-01, 00028140  
\* 0.27740702178279681994E-01,0.28138849915627150636E-01, 00028150  
\* 0.99998243035489159858E 00,0.50536095207862517625E-04, 00028160  
\* 0.99959879967191068325E 00,0.37774664632698466027E-03, 00028170  
\* 0.99831663531840739253E 00,0.93836984854238150079E-03, 00028180  
\* 0.99572410469840718851E 00,0.16811428654214699063E-02, 00028190  
\* 0.99149572117810613240E 00,0.25687649437940203731E-02, 00028200  
\* 0.98537149959852037111E 00,0.35728927835172996494E-02, 00028210  
\* 0.97714151463970571416E 00,0.46710503721143217474E-02, 00028220  
\* 0.96663785155841656709E 00,0.58434498758356395076E-02/ 00028230

DATA

\* P(141),P(142),P(143),P(144),P(145),P(146),P(147), 00028240  
\* P(148),P(149),P(150),P(151),P(152),P(153),P(154), 00028250  
\* P(155),P(156),P(157),P(158),P(159),P(160),P(161), 00028260  
00028270

```
* F(162),F(163),F(164),F(165),F(166),F(167),F(168)/ 00028280
* 0.95373000642576113641E 00,0.70724899954335554680E-02, 00028290
* 0.93832039777959288365E 00,0.83428387539681577056E-02, 00028300
* 0.92034002547001242073E 00,0.96411777297025366953E-02, 00028310
* 0.89974489977694003664E 00,0.10955733387837901648E-01, 00028320
* 0.87651341448470526974E 00,0.12275830560082770087E-01, 00028330
* 0.850644449476835027976E 00,0.13591571009765546790E-01, 00028340
* 0.82215625436498040737E 00,0.14893641664815182035E-01, 00028350
* 0.79108493379984836143E 00,0.16173218729577719942E-01, 00028360
* 0.75748396638051363793E 00,0.17421930159464173747E-01, 00028370
* 0.72142308537009891548E 00,0.18631848256138790186E-01, 00028380
* 0.68298743109107922809E 00,0.19795495048097499488E-01, 00028390
* 0.64227664250975951377E 00,0.20905851445812023852E-01, 00028400
* 0.59940393024224289297E 00,0.21956366305317824939E-01, 00028410
* 0.55449513263193254887E 00,0.22940964229387748761E-01/ 00028420
DATA 00028430
* F(169),F(170),F(171),F(172),F(173),F(174),F(175), 00028440
* F(176),F(177),F(178),F(179),F(180),F(181),F(182), 00028450
* F(183),F(184),F(185),F(186),F(187),F(188),F(189), 00028460
* F(190),F(191),F(192),F(193),F(194),F(195),F(196)/ 00028470
* 0.50768775753371660215E 00,0.23854052106038540080E-01, 00028480
* 0.45913001198983233287E 00,0.24690524744487676909E-01, 00028490
* 0.40897982122988867241E 00,0.25445769965464765813E-01, 00028500
* 0.35740383783153215238E 00,0.26115673376706097680E-01, 00028510
* 0.30457644155671404334E 00,0.26696622927450359906E-01, 00028520
* 0.25067873030348317661E 00,0.27185513229624791819E-01, 00028530
* 0.19589750271110015392E 00,0.27579749566481873035E-01, 00028540
* 0.14042423315256017459E 00,0.27877251476613701609E-01, 00028550
* 0.84454040083710883710E-01,0.28076455793817246607E-01, 00028560
* 0.28184648949745694339E-01,0.28176319033016602131E-01, 00028570
* 0.28188814180192358694E-01,0.84009692870519326354E-02, 00028580
* 0.32259500250878684614E-02,0.12539284826474884353E-01, 00028590
* 0.10544076228633167722E-02,0.58078616599775673635E-02, 00028600
* 0.10719490006251933623E-01,0.13697302631990716258E-01/ 00028610
DATA 00028620
* F(197),F(198),F(199),F(200),F(201),F(202),F(203), 00028630
* F(204),F(205),F(206),F(207),F(208),F(209),F(210), 00028640
* F(211),F(212),F(213),F(214),F(215),F(216),F(217), 00028650
* F(218),F(219),F(220),F(221),F(222),F(223),F(224)/ 00028660
* 0.31630366082226447689E-03,0.20557519893273465236E-02, 00028670
* 0.44946378920320678616E-02,0.71224386864583871532E-02, 00028680
* 0.96099525623638830097E-02,0.11703388747657003101E-01, 00028690
* 0.13208736697529129966E-01,0.13994609127619079852E-01, 00028700
* 0.90372734658751149261E-04,0.64476204130572477933E-03, 00028710
* 0.15288767050877655684E-02,0.26245617274044295626E-02, 00028720
* 0.38516876166398709241E-02,0.51485584789781777618E-02, 00028730
* 0.64674198318036867274E-02,0.77683877779219912200E-02, 00028740
* 0.90161081951956431600E-02,0.10178877529236079733E-01, 00028750
* 0.11228632913408049354E-01,0.12141082601668299679E-01, 00028760
* 0.12895813488012114694E-01,0.13476374833816515982E-01, 00028770
* 0.13870351089139840997E-01,0.14069424957813575318E-01, 00028780
* 0.25157870384280661489E-04,0.18887326450650491366E-03, 00028790
```

\* 0.46918492424785040975E-03,0.84057143271072246365E-03/  
DATA

\* F(225),F(226),F(227),F(228),F(229),F(230),F(231),\*

\* F(232),F(233),F(234),F(235),F(236),F(237),F(238),\*

\* F(239),F(240),F(241),F(242),F(243),F(244),F(245),\*

\* F(246),F(247),F(248),F(249),F(250),F(251),F(252)/

\* 0.12843824718970101768E-02,0.17864463917586498247E-02,

\* 0.23355251860571608737E-02,0.29217249379178197538E-02,

\* 0.35362449977167777340E-02,0.41714193769840788528E-02,

\* 0.4820588648512683476E-02,0.54778666939189508240E-02,

\* 0.61379152800413850435E-02,0.67957855048827733948E-02,

\* 0.74468208324075910174E-02,0.80866093647888599710E-02,

\* 0.87109650797320868736E-02,0.93159241280693950932E-02,

\* 0.98977475240487497440E-02,0.10452925722906011926E-01,

\* 0.10978183152658912470E-01,0.11470482114693874380E-01,

\* 0.11927026053019270040E-01,0.12345262372243838455E-01,

\* 0.12722884982733382906E-01,0.13057836888353048840E-01,

\* 0.13348311463725179953E-01,0.13592756614812395910E-01,

\* 0.13789874783240936517E-01,0.139386257383068850804E-01,

\* 0.14038227896908623303E-01,0.14088159516508301065E-01/  
DATA

\* F(253),F(254),F(255),F(256),F(257),F(258),F(259),\*

\* F(260),F(261),F(262),F(263),F(264),F(265),F(266),\*

\* F(267),F(268),F(269),F(270),F(271),F(272),F(273),\*

\* F(274),F(275),F(276),F(277),F(278),F(279),F(280)/

\* 0.9999975963797484642E 00,0.69379364324108267170E-05,

\* 0.99994399620705437576E 00,0.53275293669780613125E-04,

\* 0.9997904992443204733E 00,0.13575491094922871973E-03,

\* 0.99938033802502358193E 00,0.24921240048299729402E-03,

\* 0.99874561446809511470E 00,0.38974528447328229322E-03,

\* 0.997805355449595727456E 00,0.55429531493037471492E-03,

\* 0.99651414591489027385E 00,0.74028280424450333046E-03,

\* 0.99483150280062100052E 00,0.94536151685852538246E-03,

\* 0.99272134428278861533E 00,0.11674841174299594077E-02,

\* 0.99015137040077015918E 00,0.14049079956551446427E-02,

\* 0.98709252795403406719E 00,0.16561127281544526052E-02,

\* 0.98351865757863272876E 00,0.19197129710138724125E-02,

\* 0.97940628167086268381E 00,0.219440692253638388388E-02,

\* 0.97473445975240266776E 00,0.247899582266575679307E-02/  
DATA

\* F(281),F(282),F(283),F(284),F(285),F(286),F(287),\*

\* F(288),F(289),F(290),F(291),F(292),F(293),F(294),\*

\* F(295),F(296),F(297),F(298),F(299),F(300),F(301),\*

\* F(302),F(303),F(304),F(305),F(306),F(307),F(308)/

\* 0.96948465950245923177E 00,0.27721957645934509940E-02,

\* 0.96364062156981213252E 00,0.30730184347025783234E-02,

\* 0.95718821610986096274E 00,0.33803979910869203823E-02,

\* 0.95011529752129487656E 00,0.36933779170256508183E-02,

\* 0.94241156519108305981E 00,0.40110687240750233989E-02,

\* 0.93406843615772578800E 00,0.433264096809298828545E-02,

\* 0.92507893290707565236E 00,0.46573172997568547773E-02,

\* 0.91543758715576504064E 00,0.49843645647655386012E-02,

0.91543758715576504064E 00,0.49843645647655386012E-02,

0.92507893290707565236E 00,0.46573172997568547773E-02,

0.93406843615772578800E 00,0.433264096809298828545E-02,

0.94241156519108305981E 00,0.40110687240750233989E-02,

0.95011529752129487656E 00,0.36933779170256508183E-02,

0.95718821610986096274E 00,0.33803979910869203823E-02,

0.96364062156981213252E 00,0.30730184347025783234E-02,

0.96948465950245923177E 00,0.27721957645934509940E-02,

0.97473445975240266776E 00,0.247899582266575679307E-02/

00028800  
00028810  
00028820  
00028830  
00028840  
00028850  
00028860  
00028870  
00028880  
00028890  
00028900  
00028910  
00028920  
00028930  
00028940  
00028950  
00028960  
00028970  
00028980  
00028990  
00029000  
00029010  
00029020  
00029030  
00029040  
00029050  
00029060  
00029070  
00029080  
00029090  
00029100  
00029110  
00029120  
00029130  
00029140  
00029150  
00029160  
00029170  
00029180  
00029190  
00029200  
00029210  
00029220  
00029230  
00029240  
00029250  
00029260  
00029270  
00029280  
00029290  
00029300  
00029310

```
* 0.90514035881326159519E 00,0.53130866051870565663E-02, 00029320
* 0.89418456833555902286E 00,0.56428181013844441585E-02, 00029330
* 0.88256884024734190684E 00,0.59729195655081658049E-02, 00029340
* 0.87029305554811390585E 00,0.63027734490857587172E-02, 00029350
* 0.85735831088623215653E 00,0.66317812429018878941E-02, 00029360
* 0.84376688267270860104E 00,0.69593614093904229394E-02/ 00029370
DATA 00029380
* P(309),P(310),P(311),P(312),P(313),P(314),P(315), 00029390
* P(316),P(317),P(318),P(319),P(320),P(321),P(322), 00029400
* P(323),P(324),P(325),P(326),P(327),P(328),P(329), 00029410
* P(330),P(331),P(332),P(333),P(334),P(335),P(336)/ 00029420
* 0.82952219463740140018E 00,0.72849479805538070639E-02, 00029430
* 0.81462878765513741344E 00,0.76079896657190565832E-02, 00029440
* 0.79909229096084140180E 00,0.79279493342948491103E-02, 00029450
* 0.78291939411828301639E 00,0.82443037630328680306E-02, 00029460
* 0.76611781930376009072E 00,0.85565435613076896192E-02, 00029470
* 0.74869629361693660282E 00,0.88641732094824942641E-02, 00029480
* 0.73066452124218126133E 00,0.91667111635607884067E-02, 00029490
* 0.71203315536225203459E 00,0.94636899938300652943E-02, 00029500
* 0.69281376977911470289E 00,0.97546565363174114611E-02, 00029510
* 0.67301883023041847920E 00,0.10039172044056840798E-01, 00029520
* 0.65266166541001749610E 00,0.10316812330947621682E-01, 00029530
* 0.63175643771119423041E 00,0.10587167904885197931E-01, 00029540
* 0.61031811371518640016E 00,0.10849844089337314099E-01, 00029550
* 0.58836243444766254143E 00,0.11104461134006926537E-01/ 00029560
DATA 00029570
* P(337),P(338),P(339),P(340),P(341),P(342),P(343), 00029580
* P(344),P(345),P(346),P(347),P(348),P(349),P(350), 00029590
* P(351),P(352),P(353),P(354),P(355),P(356),P(357), 00029600
* P(358),P(359),P(360),P(361),P(362),P(363),P(364)/ 00029610
* 0.56590588542365442262E 00,0.11350654315980596602E-01, 00029620
* 0.54296566649831149049E 00,0.11588074033043952568E-01, 00029630
* 0.51955966153745702199E 00,0.11816385890830235763E-01, 00029640
* 0.49570640791876146017E 00,0.12035270785279562630E-01, 00029650
* 0.47142506587165887693E 00,0.12244424981611985899E-01, 00029660
* 0.44673538766202847374E 00,0.12443560190714035263E-01, 00029670
* 0.42165768662616330006E 00,0.12632403643542078765E-01, 00029680
* 0.39621280605761593918E 00,0.12810698163877361967E-01, 00029690
* 0.37042208795007823014E 00,0.12978202239537399286E-01, 00029700
* 0.34430734159943802278E 00,0.13134690091960152836E-01, 00029710
* 0.31789081206847668318E 00,0.13279951743930530650E-01, 00029720
* 0.29119514851824668196E 00,0.13413793085110098513E-01, 00029730
* 0.26424337241092676194E 00,0.13536035934956213614E-01, 00029740
* 0.23705884558982972721E 00,0.13646518102571291428E-01/ 00029750
DATA 00029760
* P(365),P(366),P(367),P(368),P(369),P(370),P(371), 00029770
* P(372),P(373),P(374),P(375),P(376),P(377),P(378), 00029780
* P(379),P(380),P(381)/ 00029790
* 0.20966523824318119477E 00,0.13745093443001896632E-01, 00029800
* 0.18208649675925219825E 00,0.13831631909506428676E-01, 00029810
* 0.15434681148137810869E 00,0.13906019601325461264E-01, 00029820
* 0.12647058437230196685E 00,0.13968158806516938516E-01, 00029830
```

```

* 0.98482396598119202090E-01,0.14017968039456608810E-01,      00029840
* 0.70406976042855179063E-01,0.14055382072649964277E-01,      00029850
* 0.42269164765363603212E-01,0.14080351962553661325E-01,      00029860
* 0.14093886410782462614E-01,0.14092845069160408355E-01,      00029870
* 0.14094407090096179347E-01/      00029880
  ICHECK = 0      00029890
C CHECK FOR TRIVIAL CASE.      00029900
  IF (A.EQ.B) GO TO 70      00029910
C SCALE FACTORS.      00029920
  SUM = (B+A)/2.0      00029930
  DIFF = (B-A)/2.0      00029940
C 1-POINT GAUSS      00029950
  FZERO = F(SUM)      00029960
  RESULT(1) = 2.0*FZERO*DIFF      00029970
  I = 0      00029980
  IOLD = 0      00029990
  INEW = 1      00030000
  K = 2      00030010
  ACUM = (0.0,0.0)      00030020
  GO TO 30      00030030
10 IF (K.EQ.8) GO TO 50      00030040
  IF(INEW+IOLD.GE.MEV) GO TO 60      00030050
  K = K + 1      00030060
  ACUM = (0.0,0.0)      00030070
C CONTRIBUTION FROM FUNCTION VALUES ALREADY COMPUTED.      00030080
  DO 20 J=1,IOLD      00030090
    I = I + 1      00030100
    ACUM = ACUM + F(I)*FUNCT(J)      00030110
  20 CONTINUE      00030120
C CONTRIBUTION FROM NEW FUNCTION VALUES.      00030130
  30 IOLD = IOLD + INEW      00030140
  DO 40 J=INEW,IOLD      00030150
    I = I + 1      00030160
    X = F(I)*DIFF      00030170
    FUNCT(J) = F(SUM+X) + F(SUM-X)      00030180
    I = I + 1      00030190
    ACUM = ACUM + F(I)*FUNCT(J)      00030200
  40 CONTINUE      00030210
  INEW = IOLD + 1      00030220
  I = I + 1      00030230
  RESULT(K) = (ACUM+F(I)*FZERO)*DIFF      00030240
C CHECK FOR CONVERGENCE.      00030250
  IF (ABS(REAL(RESULT(K))-REAL(RESULT(K-1))),LE.EPSIL*      00030260
  $ABS(REAL(RESULT(K))),AND,      00030270
  $ ABS(AIMAG(RESULT(K))-AIMAG(RESULT(K-1))),LE.EPSIL*      00030280
  $ABS(AIMAG(RESULT(K)))) GO TO 60      00030290
  GO TO 10      00030300
C CONVERGENCE NOT ACHIEVED.      00030310
  50 ICHECK = 1      00030320
C NORMAL TERMINATION.      00030330
  60 NPTS = INEW + IOLD      00030340
  RETURN      00030350

```

```

C TRIVIAL CASE                                00030360
  70 K = 2                                    00030370
    RESULT(1) = (0.0,0.0)                    00030380
    RESULT(2) = (0.0,0.0)                    00030390
    NPTS = 0                                  00030400
    RETURN                                    00030410
    END                                        00030420

    COMPLEX FUNCTION CQSUB(A, B, EPSIL, NPTS, ICHECK, RELERR, F, MEV) 00030430
    COMPLEX RELERR, F, RESULT, ESTIM, COMP 00030440
C THIS FUNCTION ROUTINE PERFORMS AUTOMATIC INTEGRATION 00030450
C OVER A FINITE INTERVAL USING THE BASIC INTEGRATION 00030460
C ALGORITHM QUAD, TOGETHER WITH, IF NECESSARY, A NON- 00030470
C ADAPTIVE SUBDIVISION PROCESS. 00030480
C THE CALL TAKES THE FORM 00030490
C CQSUB(A,B, EPSIL, NPTS, ICHECK, RELERR, F, MEV) 00030500
C AND CAUSES F(X) TO BE INTEGRATED OVER (A,B) WITH RELATIVE 00030510
C ERROR HOPEFULLY NOT EXCEEDING EPSIL. SHOULD QUAD CONVERGE 00030520
C (ICHECK=0) THEN QSUB WILL RETURN THE VALUE OBTAINED BY IT 00030530
C OTHERWISE SUBDIVISION WILL BE INVOKED AS A RESCUE 00030540
C OPERATION IN A NON-ADAPTIVE MANNER. THE ARGUMENT RELERR 00030550
C GIVES A CRUDE ESTIMATE OF THE ACTUAL RELATIVE ERROR 00030560
C OBTAINED. 00030570
C THE SUBDIVISION STRATEGY IS AS FOLLOWS 00030580
C LET THE INTERVAL (A,B) BE DIVIDED INTO 2**N PANELS AT STEP 00030590
C N OF THE SUBDIVISION PROCESS. QUAD IS APPLIED FIRST TO 00030600
C THE SUBDIVIDED INTERVAL ON WHICH QUAD LAST FAILED TO 00030610
C CONVERGE AND IF CONVERGENCE IS NOW ACHIEVED THE REMAINING 00030620
C PANELS ARE INTEGRATED. SHOULD A CONVERGENCE FAILURE OCCUR 00030630
C ON ANY PANEL THE INTEGRATION AT THAT POINT IS TERMINATED 00030640
C AND THE PROCEDURE REPEATED WITH N INCREASED BY 1. THE 00030650
C STRATEGY INSURES THAT POSSIBLY DELINQUENT INTERVALS ARE 00030660
C EXAMINED BEFORE WORK, WHICH LATER MIGHT HAVE TO BE 00030670
C DISCARDED, IS INVESTED ON WELL BEHAVED PANELS. THE 00030680
C PROCESS IS COMPLETE WHEN NO CONVERGENCE FAILURE OCCURS ON 00030690
C ANY PANEL AND THE SUM OF THE RESULTS OBTAINED BY QUAD ON 00030700
C EACH PANEL IS TAKEN AS THE VALUE OF THE INTEGRAL. 00030710
C THE PROCESS IS VERY CAUTIOUS IN THAT THE SUBDIVISION OF 00030720
C THE INTERVAL (A,B) IS UNIFORM, THE FINENESS OF WHICH IS 00030730
C CONTROLLED BY THE SUCCESS OF QUAD. IN THIS WAY IT IS 00030740
C RATHER DIFFICULT FOR A SPURIOUS CONVERGENCE TO SLIP 00030750
C THROUGH. 00030760
C THE CONVERGENCE CRITERION OF QUAD IS SLIGHTLY RELAXED 00030770
C IN THAT A PANEL IS DEEMED TO HAVE BEEN SUCCESSFULLY 00030780
C INTEGRATED IF EITHER QUAD CONVERGES OR THE ESTIMATED 00030790
C ABSOLUTE ERROR COMMITTED ON THIS PANEL DOES NOT EXCEED 00030800
C EPSIL TIMES THE ESTIMATED ABSOLUTE VALUE OF THE INTEGRAL 00030810
C OVER (A,B). THIS RELAXATION IS TO TRY TO TAKE ACCOUNT OF 00030820
C A COMMON SITUATION WHERE ONE PARTICULAR PANEL CAUSES 00030830
C SPECIAL DIFFICULTY, PERHAPS DUE TO A SINGULARITY OF SOME 00030840
C TYPE. IN THIS CASE QUAD COULD OBTAIN NEARLY EXACT 00030850
C ANSWERS ON ALL OTHER PANELS AND SO THE RELATIVE ERROR FOR 00030860
  
```

C THE TOTAL INTEGRATION WOULD BE ALMOST ENTIRELY DUE TO THE	00030870
C DELINQUENT PANEL. WITHOUT THIS CONDITION THE COMPUTATION	00030880
C MIGHT CONTINUE DESPITE THE REQUESTED RELATIVE ERROR BEING	00030890
C ACHIEVED.	00030900
C THE OUTCOME OF THE INTEGRATION IS INDICATED BY ICHECK.	00030910
C ICHECK=0 - CONVERGENCE OBTAINED WITHOUT INVOKING	00030920
C SUBDIVISION. THIS CORRESPONDS TO THE	00030930
C DIRECT USE OF QUAD.	00030940
C ICHECK=1 - RESULT OBTAINED AFTER INVOKING SUBDIVISION.	00030950
C ICHECK=2 - AS FOR ICHECK=1 BUT AT SOME POINT THE	00030960
C RELAXED CONVERGENCE CRITERION WAS USED.	00030970
C THE RISK OF UNDERESTIMATING THE RELATIVE	00030980
C ERROR WILL BE INCREASED. IF NECESSARY,	00030990
C CONFIDENCE MAY BE RESTORED BY CHECKING	00031000
C EPSIL AND RELERR FOR A SERIOUS DISCREPANCY.	00031010
C ICHECK NEGATIVE	00031020
C IF DURING THE SUBDIVISION PROCESS THE	00031030
C ALLOWED UPPER LIMIT ON THE NUMBER OF PANELS	00031040
C THAT MAY BE GENERATED (PRESENTLY 4096) IS	00031050
C REACHED A RESULT IS OBTAINED WHICH MAY BE	00031060
C UNRELIABLE BY CONTINUING THE INTEGRATION	00031070
C WITHOUT FURTHER SUBDIVISION IGNORING	00031080
C CONVERGENCE FAILURES. THIS OCCURRENCE IS	00031090
C FLAGGED BY RETURNING ICHECK WITH NEGATIVE	00031100
C SIGN.	00031110
C THE RELIABILITY OF THE ALGORITHM WILL DECREASE FOR LARGE	00031120
C VALUES OF EPSIL. IT IS RECOMMENDED THAT EPSIL SHOULD	00031130
C GENERALLY BE LESS THAN ABOUT 0.001.	00031140
DIMENSION RESULT(8)	00031150
INTEGER BAD, OUT	00031160
LOGICAL RHS	00031170
EXTERNAL F	00031180
DATA NMAX/4096/	00031190
CALL CQUAD(A, B, RESULT, K, EPSIL, NPTS, ICHECK, F, MEV)	00031200
CQSUB = RESULT(K)	00031210
RELERR = (0.0,0.0)	00031220
IF (REAL(CQSUB).NE.0.0.AND.AIMAG(CQSUB).NE.0.0) RELERR=	00031230
\$ CMLPX(ABS(REAL(RESULT(K)-RESULT(K-1)))/REAL(CQSUB),	00031240
\$ ABS(AIMAG(RESULT(K)-RESULT(K-1)))/AIMAG(CQSUB))	00031250
C CHECK IF SUBDIVISION IS NEEDED.	00031260
IF (ICHECK.EQ.0) RETURN	00031270
C SUBDIVIDE	00031280
ESTIM=CQSUB*EPSIL	00031290
ESTIM=CMLPX(ABS(REAL(ESTIM)),ABS(AIMAG(ESTIM)))	00031300
IC = 1	00031310
RHS = .FALSE.	00031320
N = 1	00031330
H = B - A	00031340
BAD = 1	00031350
10 CQSUB = (0.0,0.0)	00031360
RELERR = (0.0,0.0)	00031370
H = H*0.5	00031380

00031390  
00031400  
00031410  
00031420  
00031430  
00031440  
00031450  
00031460  
00031470  
00031480  
00031490  
00031500  
00031510  
00031520  
00031530  
00031540  
00031550  
00031560  
00031570  
00031580  
00031590  
00031600  
00031610  
00031620  
00031630  
00031640  
00031650  
00031660  
00031670  
00031680  
00031690  
00031700  
00031710  
00031720  
00031730  
00031740  
00031750  
00031760  
00031770  
00031780  
00031790  
00031800  
00031810  
00031820  
00031830  
00031840  
00031850  
00031860  
00031870  
00031880  
00031890  
00031900

```
N = N + N  
C INTERVAL (A*B) DIVIDED INTO N EQUAL SUBINTERVALS.  
C INTEGRATE OVER SUBINTERVALS BAD TO (BAD+1) WHERE TROUBLE  
C HAS OCCURRED.  
M1 = BAD  
M2 = BAD + 1  
OUT = 1  
GO TO 50  
C INTEGRATE OVER SUBINTERVALS 1 TO (BAD-1)  
20 M1 = 1  
M2 = BAD - 1  
RHS = .FALSE.  
OUT = 2  
GO TO 50  
C INTEGRATE OVER SUBINTERVALS (BAD+2) TO N.  
30 M1 = BAD + 2  
M2 = N  
OUT = 3  
GO TO 50  
C SUBDIVISION RESULT  
40 ICHECK = IC  
RELERR=CMPLX(REAL(RELERR)/ABS(COSUB)),  
$ AIMAG(RELERR)/ABS(AIMAG(COSUB)))  
RETURN  
C INTEGRATE OVER SUBINTERVALS M1 TO M2.  
50 IF (M1.GT.M2) GO TO 90  
DO 80 JJ=M1,M2  
J = JJ  
C EXAMINE FIRST THE LEFT OR RIGHT HALF OF THE SUBDIVIDED  
C TROUBLE SOME INTERVAL DEPENDING ON THE OBSERVED TREND.  
IF (RHS) J = M2 + M1 - JJ  
ALPHA = A + H*(J-1)  
BETA = ALPHA + H  
CALL CQUAD(ALPHA, BETA, RESULT, M, EPSIL, NF, ICHECK, F, MEV)  
COMP = (RESULT(M)-RESULT(M-1))  
COMP=CMPLX(ABS(REAL(COMP)),ABS(AIMAG(COMP)))  
NPTS = NPTS + NF  
IF(NPTS.GE.MEV) GO TO 70  
IF(ICHECK.NE.1) GO TO 70  
IF(REAL(COMP).LE.REAL(ESTIM).AND.  
$ AIMAG(COMP).LE.AIMAG(ESTIM)) GO TO 100  
C SUBINTERVAL J HAS CAUSED TROUBLE.  
C CHECK IF FURTHER SUBDIVISION SHOULD BE CARRIED OUT.  
IF (N.EQ.NMAX) GO TO 60  
BAD = 2*J - 1  
RHS = .FALSE.  
IF ((J-2*(J/2)).EQ.0) RHS = .TRUE.  
GO TO 10  
60 IC = -ABS(IC)  
70 COSUB = COSUB + RESULT(M)  
80 CONTINUE  
RELERR = RELERR + COMP
```



```

90 GO TO (20,30,40), OUT                                00031910
C RELAXED CONVERGENCE                                    00031920
100 IC = ISIGN(2,IC)                                     00031930
  GO TO 70                                               00031940
  END                                                    00031950

  COMPLEX FUNCTION CQSUBA(A, B, EPSIL, NPTS, ICHECK, RELERR, F, MEV) 00031960
  COMPLEX RELERR, F, RESULT, ESTIM, COMP                 00031970
C THIS FUNCTION ROUTINE PERFORMS AUTOMATIC INTEGRATION 00031980
C OVER A FINITE INTERVAL USING THE BASIC INTEGRATION 00031990
C ALGORITHM QUAD TOGETHER WITH, IF NECESSARY AN ADAPTIVE 00032000
C SUBDIVISION PROCESS. IT IS GENERALLY MORE EFFICIENT THAN 00032010
C THE NON-ADAPTIVE ALGORITHM QSUB BUT IS LIKELY TO BE LESS 00032020
C RELIABLE(SEE COMP.J.,14,189,1971).                   00032030
C THE CALL TAKES THE FORM                               00032040
C CQSUBA(A,B, EPSIL, NPTS, ICHECK, RELERR, F, MEV)     00032050
C AND CAUSES F(X) TO BE INTEGRATED OVER (A,B) WITH RELATIVE 00032060
C ERROR HOPEFULLY NOT EXCEEDING EPSIL. SHOULD QUAD CONVERGE 00032070
C (ICHECK=0) THEN QSUBA WILL RETURN THE VALUE OBTAINED BY IT 00032080
C OTHERWISE SUBDIVISION WILL BE INVOKED AS A RESCUE    00032090
C OPERATION IN AN ADAPTIVE MANNER. THE ARGUMENT RELERR GIVES 00032100
C A CRUDE ESTIMATE OF THE ACTUAL RELATIVE ERROR OBTAINED. 00032110
C THE SUBDIVISION STRATEGY IS AS FOLLOWS              00032120
C AT EACH STAGE OF THE PROCESS AN INTERVAL IS PRESENTED FOR 00032130
C SUBDIVISION (INITIALLY THIS WILL BE THE WHOLE INTERVAL 00032140
C (A,B)). THE INTERVAL IS HALVED AND QUAD APPLIED TO EACH 00032150
C SUBINTERVAL. SHOULD QUAD FAIL ON THE FIRST SUBINTERVAL 00032160
C THE SUBINTERVAL IS STACKED FOR FUTURE SUBDIVISION AND THE 00032170
C SECOND SUBINTERVAL IMMEDIATELY EXAMINED. SHOULD QUAD FAIL 00032180
C ON THE SECOND SUBINTERVAL THE SUBINTERVAL IS        00032190
C IMMEDIATELY SUBDIVIDED AND THE WHOLE PROCESS REPEATED. 00032200
C EACH TIME A CONVERGED RESULT IS OBTAINED IT IS      00032210
C ACCUMULATED AS THE PARTIAL VALUE OF THE INTEGRAL. WHEN 00032220
C QUAD CONVERGES ON BOTH SUBINTERVALS THE INTERVAL LAST 00032230
C STACKED IS CHOSEN NEXT FOR SUBDIVISION AND THE PROCESS 00032240
C REPEATED. A SUBINTERVAL IS NOT EXAMINED AGAIN ONCE A 00032250
C CONVERGED RESULT IS OBTAINED FOR IT SO THAT A SPURIOUS 00032260
C CONVERGENCE IS MORE LIKELY TO SLIP THROUGH THAN FOR THE 00032270
C NON-ADAPTIVE ALGORITHM QSUB.                         00032280
C THE CONVERGENCE CRITERION OF QUAD IS SLIGHTLY RELAXED 00032290
C IN THAT A PANEL IS DEEMED TO HAVE BEEN SUCCESSFULLY 00032300
C INTEGRATED IF EITHER QUAD CONVERGES OR THE ESTIMATED 00032310
C ABSOLUTE ERROR COMMITTED ON THIS PANEL DOES NOT EXCEED 00032320
C EPSIL TIMES THE ESTIMATED ABSOLUTE VALUE OF THE INTEGRAL 00032330
C OVER (A,B). THIS RELAXATION IS TO TRY TO TAKE ACCOUNT OF 00032340
C A COMMON SITUATION WHERE ONE PARTICULAR PANEL CAUSES 00032350
C SPECIAL DIFFICULTY, PERHAPS DUE TO A SINGULARITY OF SOME 00032360
C TYPE. IN THIS CASE QUAD COULD OBTAIN NEARLY EXACT    00032370
C ANSWERS ON ALL OTHER PANELS AND SO THE RELATIVE ERROR FOR 00032380
C THE TOTAL INTEGRATION WOULD BE ALMOST ENTIRELY DUE TO THE 00032390
C DELINQUENT PANEL. WITHOUT THIS CONDITION THE COMPUTATION 00032400
C MIGHT CONTINUE DESPITE THE REQUESTED RELATIVE ERROR BEING 00032410

```

```

C ACHIEVED. 00032420
C THE OUTCOME OF THE INTEGRATION IS INDICATED BY ICHECK. 00032430
C ICHECK=0 - CONVERGENCE OBTAINED WITHOUT INVOKING SUB- 00032440
C DIVISION. THIS WOULD CORRESPOND TO THE 00032450
C DIRECT USE OF QUAD. 00032460
C ICHECK=1 - RESULT OBTAINED AFTER INVOKING SUBDIVISION. 00032470
C ICHECK=2 - AS FOR ICHECK=1 BUT AT SOME POINT THE 00032480
C RELAXED CONVERGENCE CRITERION WAS USED. 00032490
C THE RISK OF UNDERESTIMATING THE RELATIVE 00032500
C ERROR WILL BE INCREASED. IF NECESSARY, 00032510
C CONFIDENCE MAY BE RESTORED BY CHECKING 00032520
C EPSIL AND RELERR FOR A SERIOUS DISCREPANCY. 00032530
C ICHECK NEGATIVE 00032540
C IF DURING THE SUBDIVISION PROCESS THE STACK 00032550
C OF DELINQUENT INTERVALS BECOMES FULL (IT IS 00032560
C PRESENTLY SET TO HOLD AT MOST 100 NUMBERS) 00032570
C A RESULT IS OBTAINED BY CONTINUING THE 00032580
C INTEGRATION IGNORING CONVERGENCE FAILURES 00032590
C WHICH CANNOT BE ACCOMMODATED ON THE STACK. 00032600
C THIS OCCURRENCE IS FLAGGED BY RETURNING 00032610
C ICHECK WITH NEGATIVE SIGN. 00032620
C THE RELIABILITY OF THE ALGORITHM WILL DECREASE FOR LARGE 00032630
C VALUES OF EPSIL. IT IS RECOMMENDED THAT EPSIL SHOULD. 00032640
C GENERALLY BE LESS THAN ABOUT 0.001. 00032650
  DIMENSION RESULT(8), STACK(100) 00032660
  EXTERNAL F 00032670
  DATA ISMAX/100/ 00032680
  CALL CQUAD(A, B, RESULT, K, EPSIL, NPTS, ICHECK, F,MEV) 00032690
  CQSUBA = RESULT(K) 00032700
  RELERR = (0.0,0.0) 00032710
  IF(REAL(CQSUBA).NE.0.0.AND.AIMAG(CQSUBA).NE.0.0) RELERR= 00032720
  * CMPLX(ABS(REAL(RESULT(K)-RESULT(K-1)))/REAL(CQSUBA), 00032730
  * ABS(AIMAG(RESULT(K)-RESULT(K-1)))/AIMAG(CQSUBA)) 00032740
C CHECK IF SUBDIVISION IS NEEDED 00032750
  IF (ICHECK.EQ.0) RETURN 00032760
C SUBDIVIDE 00032770
  ESTIM=CQSUBA*EPSIL 00032780
  ESTIM=CMPLX(ABS(REAL(ESTIM)),ABS(AIMAG(ESTIM))) 00032790
  RELERR = (0.0,0.0) 00032800
  CQSUBA = (0.0,0.0) 00032810
  IS = 1 00032820
  IC = 1 00032830
  SUB1 = A 00032840
  SUB3 = B 00032850
10 SUB2 = (SUB1+SUB3)*0.5 00032860
  CALL CQUAD(SUB1, SUB2, RESULT, K, EPSIL, NF, ICHECK, F,MEV) 00032870
  NPTS = NPTS + NF 00032880
  IF(NPTS.GE.MEV) GO TO 50 00032890
  COMP = (RESULT(K)-RESULT(K-1)) 00032900
  COMP=CMPLX(ABS(REAL(COMP)),ABS(AIMAG(COMP))) 00032910
  IF (ICHECK.EQ.0) GO TO 30 00032920
  IF(REAL(COMP).LE.REAL(ESTIM).AND. 00032930

```

```

    * AIMAG(COMP).LE.AIMAG(ESTIM)) GO TO 70          00032940
    IF (IS.GE.ISMAX) GO TO 20                       00032950
C STACK SUBINTERVAL (SUB1,SUB2) FOR FUTURE EXAMINATION 00032960
    STACK(IS) = SUB1                               00032970
    IS = IS + 1                                    00032980
    STACK(IS) = SUB2                               00032990
    IS = IS + 1                                    00033000
    GO TO 40                                        00033010
20 IC = -IABS(IC)                                  00033020
30 CQSUBA = CQSUBA + RESULT(K)                     00033030
    RELERR = RELERR + COMP                         00033040
40 CALL CQUAD(SUB2, SUB3, RESULT, K, EPSIL, NF, ICHECK, F,MEV) 00033050
    NPTS = NPTS + NF                              00033060
    IF(NPTS.GE.MEV) GO TO 50                      00033070
    COMP = (RESULT(K)-RESULT(K-1))                00033080
    COMP=CMPLX(ABS(REAL(COMP)),ABS(AIMAG(COMP)))  00033090
    IF (ICHECK.EQ.0) GO TO 50                    00033100
    IF(REAL(COMP).LE.REAL(ESTIM).AND.           00033110
    * AIMAG(COMP).LE.AIMAG(ESTIM)) GO TO 80      00033120
C SUBDIVIDE INTERVAL (SUB2,SUB3)                  00033130
    SUB1 = SUB2                                    00033140
    GO TO 10                                       00033150
50 CQSUBA = CQSUBA + RESULT(K)                     00033160
    RELERR = RELERR + COMP                         00033170
    IF(NPTS.GE.MEV) RETURN                        00033180
    IF (IS.EQ.1) GO TO 60                         00033190
C SUBDIVIDE THE DELINQUENT INTERVAL LAST STACKED  00033200
    IS = IS - 1                                    00033210
    SUB3 = STACK(IS)                              00033220
    IS = IS - 1                                    00033230
    SUB1 = STACK(IS)                              00033240
    GO TO 10                                       00033250
C SUBDIVISION RESULT                              00033260
60 ICHECK = IC                                     00033270
    RELERR=CMPLX(REAL(RELERR)/ABS(REAL(CQSUBA)), 00033280
    * AIMAG(RELERR)/ABS(AIMAG(CQSUBA)))          00033290
    RETURN                                         00033300
C RELAXED CONVERGENCE                              00033310
70 IC = ISIGN(2,IC)                               00033320
    GO TO 30                                       00033330
80 IC = ISIGN(2,IC)                               00033340
    GO TO 50                                       00033350
    END                                           00033360

    SUBROUTINE IMSLMQ(SUBZ,SUBEND)                 00033370
C--IMSLMQ-- DERIVATIVE--FREE 'IMSL' MARQUARDT INVERSION--5/4/79 00033380
C FOR SOLVING GENERAL NONLINEAR LEAST SQUARES PROBLEMS. USER NEED ONLY 00033390
C WRITE SUBROUTINES 'FCODE, SUBZ, AND SUBEND' EXACTLY AS USED 00033400
C IN PROGRAM 'MARQRT'. ALSO, THE SAME PARAMETER FILE05 AND DATA 00033410
C FILE10 MAY BE USED BY 'IMSLMQ' AS IN 'MARQRT'. 00033420
C 00033430
C--NOTE: 'FCODE' CANNOT BE PASSED AS EXTERNAL DUE TO THE 00033440

```

```

C 'BLACK-BOX' NATURE OF IMSL ROUTINE 'ZXSSQ' (SEE IMSL DOC.). 00033450
C THUS, ONE SHOULD RENAME ACTUAL NAME TO 'FCODE' FOR USE HERE. 00033460
C (I.E., SEE CALL FCODE IN 'FPXSSQ'--EXTERNAL FUNCTION FOR ZXSSQ). 00033470
C 00033480
C--THE USER MUST DECLARE THE CALLING PARAMETERS 00033490
C SUBZ,SUBEND (ANY DESIRED NAMES MAY BE USED) AS EXTERNAL IN 00033500
C MAIN CALLING PROGRAM; E.G., 00033510
C 00033520
C     EXTERNAL SUBZ,SUBEND 00033530
C     CALL IMSLMQ(SUBZ,SUBEND) 00033540
C     STOP 00033550
C     END 00033560
C 00033570
C--THIS INTERFACE BETWEEN 'MARQRT' AND 'IMSLMQ' WAS WRITTEN BY 00033580
C W.L.ANDERSON, U.S. GEOLOGICAL SURVEY, DENVER, COLORADO. 00033590
C 00033600
C--SEE DOCUMENTATION OF 'MARQLOOPS', USGS OPEN-FILE REPT 79-240 (1979), 00033610
C FOR DETAILS ON CODING THE REQUIRED SUBROUTINES FCODE,SUBZ, AND 00033620
C SUBEND. ALSO SEE IMSL DOCUMENTATION FOR 'ZXSSQ'. 00033630
C 00033640
C--THE INPUT ORDER ON FILE05 (PARAMETER FILE) IS: 00033650
C 00033660
C 1. TITLE (MAX. 80-CHARACTER TITLE--ALWAYS REQUIRED). 00033670
C 2. #PARMS (SAME PARMS DEFINITIONS FOR PGM MARQRT FOR PARAMETERS: 00033680
C     N,K,IF,M,IALT,ISTOP,IWT,NITER,E,SCALEP,B(),IPRT, AND IB()). 00033690
C     PLUS, ADDITIONAL PARMS FOR 'ZXSSQ' (SEE IMSL DOC): 00033700
C     IOPT,NSIG,MAXFN,EPS,DELTA,PARM(4). 00033710
C 3. (OBJECT-TIME FORMAT STATEMENT) FOR READING THE DATA MATRIX 00033720
C     (Y(I),X(I,J),J=1,M*) ON FILE IALT (DEFAULT 10), WHERE M*=M+IWT. 00033730
C (3A. INSERT DATA MATRIX HERE ONLY IF IALT=5) 00033740
C 4. #INIT OPTIONAL NAMELIST FOR READING ADDITIONAL PARMS IN 00033750
C     SUBROUTINE SUBZ (WHICH MAY BE A DUMMY SUBROUTINE). 00033760
C 5. OPTIONALLY, REPEAT STEPS 1-4, IF ISTOP=0 WAS USED IN STEP 2. 00033770
C 00033780
C--OUTPUT IS GIVEN ON FILE06 (ON-LINE USUALLY), AND 00033790
C ON FILE16 (CONTAINS ALL PRINTABLE OUTPUT); FILE06 CONTAINS ONLY 00033800
C OUTPUT VIA PARM IPRT (0--ABBREVIATED, 1 OR -2 --DETAIL). 00033810
C 00033820
C----- 00033830
C--THE USER SHOULD ADD >IML>IMSL TO THE SEARCH RULES ON MULTICS. 00033840
C (IMSL ROUTINES USED: ZXSSQ,LEQTIP,LUDECP,LUELMP,UERTST,LINVIP) 00033850
C----- 00033860
C 00033870
C     DIMENSION B(20),GRAD(20),TITLE(16),H(270),COV(20,20),SE(20) 00033880
C     DIMENSION SQWT(200),IB(20),PRNT(5),C(20),INDEX(20), 00033890
C     & XJAC(200,20),XJTJ(210),WORK(710),PARM(4),F(200) 00033900
C     INTEGER SP,SCALEP,SY,SCALEY 00033910
C--THE FOLLOWING CHARACTER STATEMENTS ONLY FOR HONEYWELL MULTICS: 00033920
C CHARACTER*4 FMT(18) 00033930
C CHARACTER*5 TITLE 00033940
C COMMON/FIXDAT/Y(200),X(200,5),BFIX(20),YMAX,IIB(20),IIP,NOBS,K 00033950
C COMMON/PRT/IPRT 00033960

```

```

EXTERNAL FPXSSQ,LNXSSQ                                00033970
EQUIVALENCE (SQWT(1),X(1,5)),(SP,SCALEP),(N,NOBS),    00033980
& (M,NVARS),(NITER,LIMIT),(SY,SCALEY),(E,EPS),(SS,SSQ), 00033990
& (IB(1),IIB(1)),(IP,IIP),(B(1),BFIX(1))             00034000
  NAMELIST/PARMS/N,K,IP,M,IALT,NITER,IB,E,B,IWT,ISTOP,SP,SY, 00034010
  & SCALEP,SCALEY, IDER,IPRT,INON,FF,T,TAU,XL,MODLAM,GAMCR, 00034020
  & DEL,ZETA,IOUT,IOPT,NSIG,MAXFN,EPS,DELTA,PARM      00034030
C-- NOTE NAMELIST PARMS INCLUDED (BUT IGNORED) FOR COMPATIBILITY 00034040
C ARE: IDER,INON,FF,T,TAU,XL,MODLAM,GAMCR,DEL,ZETA,IOUT. 00034050
C ALSO, SP=SCALEP WILL BE CONSIDERED MODE=1 (ALOG OPTION) ONLY 00034060
C WHEN SP=2 OR 1 (AND MODE=0 LINEAR WHEN SP=0). SY=SCALEY IS 00034070
C IGNORED FOR THIS VERSION OF 'IMSLMQ'.              00034080
C                                                     00034090
C--READ IMSLMQ TITLE LINE                             00034100
  READ(5,4) TITLE                                     00034110
4  FORMAT(16A5)                                       00034120
C--PRESET DEFAULTS                                   00034130
  N=0                                                 00034140
  K=0                                                 00034150
  M=0                                                 00034160
  IP=0                                                00034170
  IPRT=0                                              00034180
  ISTOP=1                                             00034190
  IWT=0                                               00034200
  IALT=10                                             00034210
  NITER=10                                           00034220
  SP=0                                                00034230
  MODE=0                                              00034240
  FMIN=0.0                                           00034250
  IOPT=1                                              00034260
  NSIG=3                                              00034270
  MAXFN=0                                             00034280
  EPS=0.0                                             00034290
  DELTA=0.0                                          00034300
  PARM(1)=.01                                        00034310
  PARM(2)=2.                                         00034320
  PARM(3)=120.                                       00034330
  PARM(4)=.1                                         00034340
  DO 5 I=1,20                                       00034350
  IB(I)=0                                            00034360
  B(I)=0.0                                           00034370
5  GRAD(I)=0.0                                       00034380
C--READ $PARMS                                       00034390
6  READ(5,PARMS)                                     00034400
C--TEST $PARMS BEFORE PROCEEDING                    00034410
  IF(N.GT.200.OR.K.GT.20.OR.M.GT.4.OR.IWT.GT.1.OR.IP.GT.19.OR. 00034420
  & N.LT.1.OR.K.LT.1.OR.M.LT.1.OR.IWT.LT.0.OR.IP.LT.0.OR. 00034430
  & N.LT.K-IP.OR.IALT.EQ.6.OR.IALT.EQ.16)           00034440
  &CALL ERRMSG('SOME $PARMS OUT OF RANGE.',5,6,16) 00034450
  DO 7 I=1,K                                         00034460
  IF(B(I).EQ.0.0)CALL ERRMSG('SOME B(I)=0.0 ',3,6,16) 00034470
7  CONTINUE                                          00034480

```

```

00034490 IF(MAXFN,EG,0) MAXFN=2*K*NITER
00034500 DO 8 I=1,IF
00034510 IF(IR(I),GT,0) GO TO 8
00034520 CALL ERRMSG('IF,GT,1 BUT SOME IR(I),LE,0',6,6,16)
00034530 CONTINUE
00034540 IF(SF,NE,0) MODE=1
00034550 IF(IFRT,EG,1) IFRT=-2
00034560 C--READ OBJECT FORMAT FOR DATA MATRIX ON FILE IALT,
00034570 READ(5,10) (FMT(I),I=1,18)
00034580 FORMAT(18A4)
00034590 M1=M+IWT
00034600 YMAX=0.0
00034610 DO 11 I=1,N
00034620 READ(IALT,FMT) Y(I),(X(I),J),J=1,M1)
00034630 SQRT(I)=1.0
00034640 IF(IWT,EG,1,AND,X(I,M1),NE,0.0) SQRT(I)=1.0/X(I,M1)
00034650 IF(ABS(Y(I)),GT,YMAX) YMAX=ABS(Y(I))
00034660 CONTINUE
00034670 IF(IALT,NE,5) REMIND IALT
00034680 C--INITIALIZATION VIA CALL SUBZ (READ #INIT, TEST B*X,Y, ETC)
00034690 CALL SUBZ(Y,X,B,FRNT,NDUM,N,TITLE,1)
00034700 C--WRITE $PARMS ON UNIT 6 AND 16
00034710 WRITE(6,60) TITLE,N,K,IF,M,E,IALT,ISTOP,IWT,NITER,SP,IFRT,
00034720 & IOPT,NSIG,MAXFN,EFS,DELTA,PARM
00034730 FORMAT('11 M S L M 0 ---',16A5//N=',15,9X',K=',14,10X',IF=
00034740 & '14,9X',M=',13,11X',E=',E10.3// IALT=',13,8X',ISTOP=',12,8X',
00034750 & IWT=',12,10X',NITER=',16,4X',SCALEF=',12// IFRT=',13,
00034760 & 8X',IOPT=',12,9X',NSIG=',13,8X',MAXFN=',16,4X',EFS=',E10.3/
00034770 & / DELTA=',E10.3// PARM=',4E10.3)
00034780 WRITE(16,60) TITLE,N,K,IF,M,E,IALT,ISTOP,IWT,NITER,SP,IFRT,
00034790 & IOPT,NSIG,MAXFN,EFS,DELTA,PARM
00034800 IF(IF,EG,0) GO TO 661
00034810 WRITE(6,660) (IR(I),I=1,IF)
00034820 FORMAT('/',IR=',19I3)
00034830 WRITE(16,660) (IR(I),I=1,IF)
00034840 WRITE(6,662) FMT
00034850 WRITE(6,662) FMT
00034860 FORMAT('/',FMT=',18A4/)
00034870 WRITE(16,662) FMT
00034880 WRITE(6,6000) (B(I),I=1,K)
00034890 FORMAT('/',INITIAL PARAMETERS',//(5E16.8))
00034900 C--INTERFACE WITH ZXSSQ USING MARGRT FCODE
00034910 DO 1 I=1,20
00034920 INDEX(I)=1
00034930 KIP=K-IF
00034940 IF(IF,EG,0) GO TO 400
00034950 C--REORDER B TO C WHEN IF>0
00034960 IM=0
00034970 DO 202 I=1,K
00034980 DO 201 J=1,IF
00034990 IF(I,EG,IB(J)) GO TO 202
00035000

```

```

201 CONTINUE                                00035010
    IM=IM+1                                00035020
    C(IM)=B(I)                              00035030
    INDEX(IM)=I                             00035040
202 CONTINUE                                00035050
    WRITE(6,203) (I,I=1,K)                  00035060
203 FORMAT('/ PARAMETER INDEX:',2013)        00035070
    WRITE(16,203) (I,I=1,K)                 00035080
    WRITE(6,204) (INDEX(I),I=1,KIP)         00035090
204 FORMAT(' REORDERED AS...:',2013)        00035100
    WRITE(16,204) (INDEX(I),I=1,KIP)        00035110
    WRITE(6,206) (C(I),I=1,KIP)            00035120
206 FORMAT('/ REORDERED PARAMETERS'//(5E16.8)) 00035130
    WRITE(16,206) (C(I),I=1,KIP)           00035140
    GO TO 500                                00035150
400 DO 401 I=1,K                            00035160
401 C(I)=B(I)                              00035170
500 CONTINUE                                00035180
    IF(MODE.EQ.0) GO TO 12                  00035190
C--LOG PARAMETERS CHOSEN (MODE=1 OR SP.NE.0) 00035200
    DO 111 I=1,KIP                          00035210
    IF(C(I).LE.0.0)CALL ERRMSG('SP.NE.0 & SOME B(I).LE.0.',5,6,16) 00035220
111 C(I)=ALOG(C(I))                          00035230
    CALL ZXSSQ(LNXSSQ,N,KIP,NSIG,EPS,DELTA,MAXFN,IOPT,PARM, 00035240
    & C,SSQ,F,XJAC,200,XJTJ,WORK,INFER,IER) 00035250
    DO 1111 I=1,KIP                         00035260
1111 C(I)=EXP(C(I))                          00035270
    GO TO 21                                00035280
C--LINEAR PARAMETERS CHOSEN (MODE=0 OR SP=0) 00035290
12 CALL ZXSSQ(FPXSSQ,N,KIP,NSIG,EPS,DELTA,MAXFN,IOPT,PARM, 00035300
    & C,SSQ,F,XJAC,200,XJTJ,WORK,INFER,IER) 00035310
C--ZXSSQ ERRMSG CODE HANDLERS              00035320
21 IF(IER.EQ.0) GO TO 100                   00035330
    IF(IER.EQ.129)                          00035340
    &CALL ERRMSG('SINGULARITY DETECTED IN JACOBIAN & RECOVERY FAILED', 00035350
    & 10,6,16)                               00035360
    IF(IER.EQ.130)                          00035370
    &CALL ERRMSG('N,K-IP,IOPT,PARM(1) OR PARM(2) INCORRECT',8,6,16) 00035380
    IF(IER.EQ.131)                          00035390
    &CALL WARN('MARQUARDT PARAMETER EXCEEDED PARM(3) ',8,6,16,$100) 00035400
    IF(IER.EQ.132) CALL ERRMSG(              00035410
    & 'AFTER RECOVERY FROM SINGULAR JACOBIAN, B CYCLED BACK AGAIN..', 00035420
    & 12,6,16)                               00035430
    IF(IER.EQ.133)CALL WARN('MAXFN EXCEEDED.',3,6,16,$100) 00035440
    IF(IER.EQ.38)CALL WARN('JACOBIAN=0. SOLUTION IS STATIONARY POINT',00035450
    & 8,6,16,$100)                            00035460
100 WRITE(6,603) (WORK(I),I=1,5),INFER,IER,SSQ,(C(I),I=1,KIP) 00035470
    WRITE(16,603) (WORK(I),I=1,5),INFER,IER,SSQ,(C(I),I=1,KIP) 00035480
603 FORMAT('/// $$$ $ IMSLM CONVERGENCE INFORMATION: '/// 00035490
    & ' NORM OF GRADIENT',T32,E16.8/' FUNCTION EVALUATIONS',T32,E16.8/ 00035500
    & ' EST. SIGN. DIGITS',T32,E16.8/' MARQUARDT PARAMETER',T32, 00035510
    & E16.8/' NO. ITERATIONS',T32,E16.8/' TYPE CONVERGENCE (INFER)', 00035520
  
```

```

& T32,I3/' ERROR CODE (IER)',T32,I5/ 00035530
& ' RESIDUAL SUM-OF-SQUARES (SSQ)=' ,E16.8// 00035540
&' **** FINAL UNSCALED PARAMETERS'// 00035550
& (5E16.8)) 00035560
99 KK=MAX0((KIP+1)*KIP/2,5) 00035570
DO 80 I=1,KIP 00035580
80 GRAD(I)=2.*WORK(KK+I) 00035590
WRITE(6,82) (GRAD(I),I=1,KIP) 00035600
82 FORMAT(/' SCALED GRADIENT'// (5E16.8)) 00035610
WRITE(16,82) (GRAD(I),I=1,KIP) 00035620
IF(IPRT.EQ.-2) WRITE(6,699) 00035630
699 FORMAT(/3X,'I',4X,'OBS.Y(I)',6X,'CAL',11X,'RES',8X,'X(I,1)',8X, 00035640
& 'WT(I)') 00035650
WRITE(16,1699) 00035660
1699 FORMAT(/3X,'I',4X,'OBS.Y(I)',6X,'CAL',11X,'RES',8X,'X(I,1)',8X, 00035670
& 'X(I,2)',8X,'X(I,3)',8X,'X(I,4)',8X,'WT(I)') 00035680
SUMF2=0.0 00035690
DO 110 I=1,NOBS 00035700
RES=F(I)/SQWT(I) 00035710
YCAL=Y(I)-RES 00035720
WT=SQWT(I)*SQWT(I) 00035730
SUMF2=SUMF2+F(I)*F(I) 00035740
IF(IPRT.EQ.-2) WRITE(6,210) I,Y(I),YCAL,RES,X(I,1),WT 00035750
210 FORMAT(1X,I3,2E14.6,E11.3,2E14.6) 00035760
WRITE(16,211) I,Y(I),YCAL,RES,(X(I,J),J=1,4),WT 00035770
211 FORMAT(1X,I3,2E14.6,E11.3,5E14.6) 00035780
110 CONTINUE 00035790
IF(N.EQ.KIP) RMSERR=0.0 00035800
IF(N.GT.KIP) RMSERR=SQRT(SUMF2/(N-KIP)) 00035810
WRITE(6,604) RMSERR 00035820
604 FORMAT(/' **** RMSERR=' ,E16.8) 00035830
WRITE(16,604) RMSERR 00035840
C--PRINT ON FILE16 (ONLY) THE FINAL SCALED PARTIALS (JACOBIAN) 00035850
WRITE(16,605) 00035860
605 FORMAT(/' FINAL SCALED PARTIALS (JACOBIAN)') 00035870
DO 112 I=1,NOBS 00035880
WRITE(16,606) I,(XJAC(I,J),J=1,KIP) 00035890
606 FORMAT(1X,I3,5E16.8/(4X,5E16.8)) 00035900
112 CONTINUE 00035910
C--GET INVERSE JACOBIAN TRANSPOSE*JACOBIAN (FROM XJTJ SYMMETRIC MATRIX) 00035920
CALL LINVIP(XJTJ,KIP,H,5,D1,D2,IER) 00035930
IF(IER.GT.128) CALL ERRMSG('IN LINVIP CALL.',3,6,16) 00035940
C--FINAL STATISTICS 00035950
DO 301 I=1,KIP 00035960
DO 301 J=1,KIP 00035970
301 COV(I,J)=H(LOC(I,J)) 00035980
IF(IPRT.EQ.-2) WRITE(6,120) 00035990
120 FORMAT(/' SCALED COVARIANCE MATRIX (INVERSE OF XJTJ)') 00036000
WRITE(16,120) 00036010
DO 122 I=1,KIP 00036020
SE(I)=SQRT(ABS(COV(I,I))) 00036030
IF(IPRT.EQ.-2) WRITE(6,300) INDEX(I),(COV(I,J),J=1,KIP) 00036040

```



```

300 FORMAT(1X,I2,1OE12.4/(3X,1OE12.4))
122 WRITE(16,300) INDEX(I),(COV(I,J),J=1,KIP)
CONTINUE
IF(IPRT,EG,-2) WRITE(6,304)
FORMAT(//,CORRELATION MATRIX')
304 WRITE(16,304)
DO 131 I=1,KIP
IF(SE(I),EG,0.0) GO TO 132
DO 129 J=1,KIP
IF(SE(J),EG,0.0) GO TO 129
COV(I,J)=COV(I,J)/(SE(I)*SE(J))
CONTINUE
IF(IPRT,EG,-2) WRITE(6,300) INDEX(I),(COV(I,J),J=1,KIP)
GO TO 131
GO TO 131
COV(I,I)=1.0
GO TO 133
CONTINUE
125 WRITE(6,303)
303 FORMAT(/15H ** PARAMETER,3X,9HSTD ERROR,3X,
& 31HSTD ERROR/PARAMETER (UNSCALED))
WRITE(16,303)
DO 126 I=1,KIP
SE(I)=RMSERR*SE(I)
IF(SF,GT,0) SE(I)=C(I)*SE(I)
SEC=SE(I)/C(I)
WRITE(6,300) INDEX(I),C(I),SE(I),SEC
WRITE(16,300) INDEX(I),C(I),SE(I),SEC
CONTINUE
DO 600 I=1,KIP
H(I)=C(I)
IF(IF,EG,0) GO TO 601
IF(IF,EG,0) GO TO 601
C--PUT SOL C AND BFIX TOGETHER FOR SUBEND USE.
IM=0
DO 127 I=1,K
H(I)=B(I)
DO 128 J=1,IP
IF(I,EG,IB(J)) GO TO 127
CONTINUE
128 CONTINUE
IM=IM+1
H(I)=C(IM)
CONTINUE
CALL SUBEND(Y,X,H,K,N,TITLE,I)
GO TO 999
READ(5,4) TITLE
DO 1000 I=1,20
B(I)=0.0
GO TO 6
C--FOLLOWING CALL ONLY FOR HONEYWELL MULTICS SYSTEM:
999 CALL CLOSE_FILE(,-ALL)
STOP
RETURN
00036050
00036050 FORMAT(1X,I2,1OE12.4/(3X,1OE12.4))
00036060 WRITE(16,300) INDEX(I),(COV(I,J),J=1,KIP)
00036070 CONTINUE
00036080 IF(IPRT,EG,-2) WRITE(6,304)
00036090 FORMAT(//,CORRELATION MATRIX')
00036100 WRITE(16,304)
00036110 DO 131 I=1,KIP
00036120 IF(SE(I),EG,0.0) GO TO 132
00036130 DO 129 J=1,KIP
00036140 IF(SE(J),EG,0.0) GO TO 129
00036150 COV(I,J)=COV(I,J)/(SE(I)*SE(J))
00036160 CONTINUE
00036170 IF(IPRT,EG,-2) WRITE(6,300) INDEX(I),(COV(I,J),J=1,KIP)
00036180 GO TO 131
00036190 GO TO 131
00036200 COV(I,I)=1.0
00036210 GO TO 133
00036220 CONTINUE
00036230 125 WRITE(6,303)
00036240 303 FORMAT(/15H ** PARAMETER,3X,9HSTD ERROR,3X,
& 31HSTD ERROR/PARAMETER (UNSCALED))
00036250 WRITE(16,303)
00036260 DO 126 I=1,KIP
00036270 SE(I)=RMSERR*SE(I)
00036280 IF(SF,GT,0) SE(I)=C(I)*SE(I)
00036290 SEC=SE(I)/C(I)
00036300 WRITE(6,300) INDEX(I),C(I),SE(I),SEC
00036310 WRITE(16,300) INDEX(I),C(I),SE(I),SEC
00036320 CONTINUE
00036330 DO 600 I=1,KIP
00036340 H(I)=C(I)
00036350 IF(IF,EG,0) GO TO 601
00036360 IF(IF,EG,0) GO TO 601
00036370 C--PUT SOL C AND BFIX TOGETHER FOR SUBEND USE.
00036380 IM=0
00036390 DO 127 I=1,K
00036400 H(I)=B(I)
00036410 DO 128 J=1,IP
00036420 IF(I,EG,IB(J)) GO TO 127
00036430 CONTINUE
00036440 IM=IM+1
00036450 H(I)=C(IM)
00036460 CONTINUE
00036470 CALL SUBEND(Y,X,H,K,N,TITLE,I)
00036480 GO TO 999
00036490 READ(5,4) TITLE
00036500 DO 1000 I=1,20
00036510 B(I)=0.0
00036520 GO TO 6
00036530 C--FOLLOWING CALL ONLY FOR HONEYWELL MULTICS SYSTEM:
00036540 999 CALL CLOSE_FILE(,-ALL)
00036550 STOP
00036560 RETURN

```

```

END 00036570

SUBROUTINE FPXSSQ(C,N,KIP,F) 00036580
C--CALCULATES RESIDUAL VECTOR F(N) FOR 'ZXSSQ' (EXTERNAL FPXSSQ) FOR 00036590
C UNSCALED C(KIP) PARAMETER VECTOR AND DATA X(200,5),Y(200) IN FIXDAT. 00036600
C 00036610
C C= INPUT VECTOR OF PARAMETERS (LENGTH KIP) 00036620
C N= NO. OBS. <= 200 00036630
C KIP= NO. PARAMETERS =K-IP (IP>=0) 00036640
C F= OUTPUT VECTOR OF (WEIGHTED) FUNCTION RESIDUALS (LENGTH N) 00036650
C 00036660
C--CALLS 'FCODE' AS CODED FOR 'MARQRT' WITH FIXED DATA IN COMMON/FIXDAT/00036670
C 00036680
C DIMENSION C(1),F(1),PRNT(5),SQWT(200),BIP(20) 00036690
C COMMON/FIXDAT/Y(200),X(200,5),BFIX(20),YMAX,IIB(20),IIP,NOBS,K 00036700
C EQUIVALENCE (SQWT(1),X(1,5)) 00036710
C IF(IIP.GT.0) GO TO 2 00036720
C DO 1 I=1,N 00036730
C CALL FCODE(Y,X,C,PRNT,FF,I,1) 00036740
1 F(I)=SQWT(I)*(Y(I)-FF) 00036750
C RETURN 00036760
C 00036770
C IM=0 00036780
C DO 4 I=1,K 00036790
C BIP(I)=BFIX(I) 00036800
C DO 3 J=1,IIP 00036810
C IF(I.EQ.IIB(J)) GO TO 4 00036820
3 CONTINUE 00036830
C IM=IM+1 00036840
C BIP(I)=C(IM) 00036850
4 CONTINUE 00036860
C DO 5 I=1,N 00036870
C CALL FCODE(Y,X,BIP,PRNT,FF,I,1) 00036880
5 F(I)=SQWT(I)*(Y(I)-FF) 00036890
C RETURN 00036900
C END 00036910

SUBROUTINE LNXSSQ(C,N,KIP,F) 00036920
C--INTERFACES TO 'FPXSSQ' TO ALLOW PARMS TO BE IN LOG OR LINEAR 00036930
C SPACE OUTSIDE, BUT ALWAYS LINEAR WITHIN FPXSSQ. 00036940
C 00036950
C--CALLS 'FPSXXQ' (AND IN TURN 'FCODE') 00036960
C 00036970
C DIMENSION C(1),F(1),CTEM(20) 00036980
C COMMON/PRT/IPRT 00036990
C DO 1 I=1,KIP 00037000
1 CTEM(I)=EXP_(C(I)) 00037010
C CALL FPXSSQ(CTEM,N,KIP,F) 00037020
C RETURN 00037030
C END 00037040

SUBROUTINE SAVER(I,J) 00037050
C=====00037050

```

```

C---GENERAL UTILITY TO MODIFY COMMON/SAVE/ AS FOLLOWS          00037060
C CALL SAVER(I,J) WILL REPLACE FSAVE() ARRAY WITH             00037070
C FSAVE(K)=GSAVE(K)**I * FSAVE(K)**J                          00037080
C FOR K=1,NSAVE.                                              00037090
C                                                                00037100
C INPUT PARAMETERS (I,J) MAY BE NEGATIVE,ZERO, OR POSITIVE  00037110
C INTEGERS.                                                    00037120
C---SUBROUTINE SAVER MAY BE USED IN CONJUNCTION WITH SUBPROGRAM 00037130
C ZHANKS TO MODIFY SAVED KERNELS WHEN USING NEW=0 OPTION (SEE  00037140
C ZHANKS).                                                      00037150
C                                                                00037160
C   COMPLEX FSAVE                                              00037170
C   COMMON/SAVE/FSAVE(283),GSAVE(283),NSAVE                    00037180
C   DO 1 K=1,NSAVE                                             00037190
C   FSAVE(K)=CMPLX(GSAVE(K)**I,0.0)*(FSAVE(K)**J)              00037200
1  CONTINUE                                                    00037210
   RETURN                                                       00037220
   END                                                            00037220

C   COMPLEX FUNCTION ZHANKS(N,B,FUN,TOL,NF,NEW)                 00037230
C=====00037240
C COMPLEX HANKEL TRANSFORMS OF ORDER 0 OR 1 FOR RELATED (SAVED) 00037250
C KERNELS AND FIXED TRANSFORM ARGUMENT B.GT.0.                00037260
C                                                                00037270
C---REF: ANDERSON, W.L., 1979, GEOPHYSICS, VOL. 44, NO. 7, P. 00037280
C 1287-1305.                                                    00037290
C---SUBPROGRAM ZHANKS EVALUATES THE INTEGRAL FROM 0 TO INFINITY 00037300
C OF FUN(G)*JN(G*B)*DG, DEFINED AS THE COMPLEX HANKEL TRANSFORM 00037310
C OF ORDER N (=0 OR 1) AND TRANSFORM ARGUMENT B.GT.0. THE METHOD 00037320
C IS BY ADAPTIVE DIGITAL FILTERING OF THE COMPLEX KERNEL FUNCTION 00037330
C FUN, USING DIRECT AND/OR PREVIOUSLY SAVED KERNEL FUNCTION VALUES. 00037340
C                                                                00037350
C---PARAMETERS (ALL INPUT, EXCEPT NF)                        00037360
C                                                                00037370
C   N      = ORDER (=0 OR 1) OF THE HANKEL TRANSFORM TO BE EVALUATED. 00037380
C   B      = REAL TRANSFORM ARGUMENT B.GT.0.0 OF THE HANKEL TRANSFORM. 00037390
C           IF NEW=0, B IS ASSUMED EQUAL TO THE LAST B USED WHEN NEW=1 00037400
C           (SEE PARAMETER NEW AND SUBPROGRAM USAGE BELOW).        00037410
C   FUN(G)= EXTERNAL DECLARED COMPLEX FUNCTION NAME (USER SUPPLIED) 00037420
C           OF A REAL ARGUMENT G.GT.0. THIS REFERENCE MUST BE SUPPLIED 00037430
C           EVEN WHEN NEW=0, SINCE THE ADAPTIVE CONVOLUTION         00037440
C           MAY NEED SOME DIRECT FUNCTION CALLS (E.G. IF TOL REDUCED). 00037450
C           IF PARAMETERS OTHER THAN G ARE REQUIRED IN FUN, USE COMMON 00037460
C           IN THE CALLING PROGRAM AND IN SUBPROGRAM FUN. BOTH     00037470
C           REAL AND IMAGINARY PARTS OF THE COMPLEX FUNCTION FUN(G) 00037480
C           MUST BE CONTINUOUS BOUNDED FUNCTIONS FOR G.GT.0.0. FOR A 00037490
C           REAL FUNCTION F1(G), FUN=CMPLX(F1(G),0.0) MAY BE USED.  00037500
C           TWO INDEPENDENT REAL-FUNCTIONS F1(G),F2(G) MAY BE      00037510
C           INTEGRATED IN PARALLEL BY WRITING FUN=CMPLX(F1(G),F2(G)). 00037520
C   TOL    = REQUESTED REAL TRUNCATION TOLERANCE ACCEPTED AT THE FILTER 00037530
C           TAILS FOR ADAPTIVE FILTERING. A TRUNCATION CRITERION IS  00037540
C           DEFINED DURING CONVOLUTION IN A FIXED ABSCISSA RANGE AS  00037550
C           THE MAX. ABSOLUTE CONVOLVED PRODUCT TIMES TOL. TYPICALLY, 00037560

```

```

C      TOL.LE.0.00001 WOULD GIVE ABOUT .01 PER CENT ACCURACY      00037570
C      FOR WELL-BEHAVED KERNELS AND MODERATE VALUES OF B. FOR    00037580
C      VERY LARGE OR SMALL B, A VERY SMALL TOL SHOULD BE USED.    00037590
C      IN GENERAL, DECREASING THE TOLERANCE WOULD PRODUCE HIGHER  00037600
C      ACCURACY IN THE CONVOLUTION SINCE MORE FILTER WEIGHTS ARE   00037610
C      USED (UNLESS EXPONENT UNDERFLOWS OCCUR IN THE KERNEL       00037620
C      EVALUATION --- SEE NOTE (1) BELOW).                          00037630
C      FOR MAXIMUM ACCURACY POSSIBLE, TOL=0.0 MAY BE USED.        00037640
C      NF      = TOTAL NUMBER OF DIRECT FUN CALLS USED DURING CONVOLUTION 00037650
C      FOR ANY VALUE OF NEW (NF IS AN OUTPUT PARAMETER).          00037660
C      NF IS IN THE RANGE 21.LE.NF.LE.283 WHEN NEW=1. USUALLY,    00037670
C      NF IS MUCH LESS THAN 283 (OR 0) WHEN NEW=0.                00037680
C      NEW  =1 IS REQUIRED FOR THE VERY FIRST CALL TO ZHANKS, OR IF  00037690
C      FORCING DIRECT FUNCTION FUN(G) CALLS, E.G., IF USING      00037700
C      ZHANKS FOR UNRELATED KERNELS.                               00037710
C      NEW=1 INITIALIZES COMMON/SAVE/FSAVE(283),GSAVE(283),NSAVE  00037720
C      FOR NSAVE COMPLEX KERNEL VALUES IN FSAVE AND CORRESPONDING 00037730
C      REAL ARGUMENTS IN GSAVE FOR THE GIVEN PARAMETER B.         00037740
C      NEW  =0 TO USE RELATED KERNELS (MODIFIED BY USER) CURRENTLY STORED 00037750
C      IN COMMON/SAVE/. FUN IS CALLED ONLY IF REQUIRED             00037760
C      DURING THE CONVOLUTION. ADDITIONAL FUNCTION VALUES WHEN   00037770
C      NEEDED ARE AUTOMATICALLY ADDED TO THE COMMON/SAVE/ BLOCK.  00037780
C      00037790
C      ***** NOTE THAT IT IS THE USERS RESPONSIBILITY TO MODIFY THE 00037800
C      COMMON FSAVE() VALUES FOR NEW=0 CALLS, EXTERNALLY IN      00037810
C      THE USERS CALLING PROGRAM (SEE SUBPROGRAM USAGE BELOW).    00037820
C      00037830
C=====00037840
C---SUBPROGRAM USAGE--- ZHANKS IS CALLED AS FOLLOWS      00037850
C      ... 00037860
C      COMPLEX Z1,Z2,ZHANKS,FSAVE 00037870
C      COMMON/SAVE/FSAVE(283),GSAVE(283),NSAVE 00037880
C      EXTERNAL ZF1,ZF2 00037890
C      ... 00037900
C      Z1=ZHANKS(N1,B,ZF1,TOL,NF1,1) 00037910
C      DO 1 I=1,NSAVE 00037920
C      C---MODIFY FSAVE IN COMMON/SAVE/ TO OBTAIN RELATED ZF2 FROM ZF1. 00037930
C      C---E.G. FSAVE(I)=GSAVE(I)*FSAVE(I) --- FOR RELATION ZF2(G)=G*ZF1(G) 00037940
C      1 CONTINUE 00037950
C      Z2=ZHANKS(N2,B,ZF2,TOL,NF2,0) 00037960
C      ... 00037970
C      END 00037980
C      COMPLEX FUNCTION ZF1(G) 00037990
C      ...USER SUPPLIED CODE FOR DIRECT EVALUATION OF ZF1(G), G.GT.0. 00038000
C      END 00038010
C      COMPLEX FUNCTION ZF2(G) 00038020
C      ...USER SUPPLIED CODE FOR DIRECT EVALUATION OF ZF2(G), G.GT.0. 00038030
C      END 00038040
C=====00038050
C---NOTES 00038060
C      (1). EXP--UNDERFLOW MAY OCCUR IN EXECUTING THIS SUBPROGRAM. 00038070
C      THIS IS OK PROVIDED THE MACHINE SYSTEM CONDITIONALLY SETS 00038080
  
```

```

C          EXP-UNDERFLOW TO 0.0.                                00038090
C      (2).  ANSI FORTRAN (AMERICAN STANDARD X3.9-1966) IS USED, EXCEPT 00038100
C          DATA STATEMENTS MAY NEED TO BE CHANGED FOR SOME COMPILERS. 00038110
C          TO CONVERT ZHANKS TO THE NEW AMERICAN STANDARD FORTRAN      00038120
C          (X3.9-1978), ADD THE FOLLOWING DECLARATION TO THIS ROUTINE 00038130
C          SAVE Y1,ISAVE                                             00038140
C      (3).  THE FILTER ABSCISSA CORRESPONDING TO EACH FILTER WEIGHT    00038150
C          IS GENERATED IN DOUBLE-PRECISION (TO REDUCE ROUND-OFF),    00038160
C          BUT IS USED IN SINGLE-PRECISION IN FUNCTION FUN.          00038170
C      (4).  NO CHECKS ARE MADE ON CALLING PARAMETERS (TO SAVE TIME),  00038180
C          HENCE UNPREDICTABLE RESULTS COULD OCCUR IF ZHANKS         00038190
C          IS CALLED INCORRECTLY (OR IF FUN OR COMMON IS IN ERROR).  00038200
C=====00038210
C          COMPLEX FUN,C,CMAX,FSAVE                                  00038220
C          COMMON/SAVE/FSAVE(283),GSAVE(283),NSAVE                  00038230
C          DOUBLE PRECISION E,ER,Y1,Y                               00038240
C          DIMENSION T(2),TMAX(2)                                   00038250
C          DIMENSION WTO(283),WAO(76),WBO(76),WCO(76),WDO(55),    00038260
C          * WT1(283),WA1(76),WB1(76),WC1(76),WD1(55)             00038270
C          EQUIVALENCE (WTO(1),WAO(1)),(WTO(77),WBO(1)),(WTO(153),WCO(1)), 00038280
C          * (WTO(229),WDO(1)),(WT1(1),WA1(1)),(WT1(77),WB1(1)),  00038290
C          * (WT1(153),WC1(1)),(WT1(229),WD1(1))                  00038300
C          EQUIVALENCE (C,T(1)),(CMAX,TMAX(1))                    00038310
C-----E=DEXP(.2D0), ER=1.0D0/E'                                00038320
C          DATA E/1.221402758160169834 D0/,ER/.818730753077981959 D0/ 00038330
C--JO--TRANSFORM FILTER WEIGHT ARRAYS (EQUIVALENT TO WTO ARRAY)  00038340
C          DATA WAO/                                             00038350
C          * 2.1969101E-11, 4.1201161E-09,-6.1322980E-09, 7.2479291E-09, 00038360
C          *-7.9821627E-09, 8.5778983E-09,-9.1157294E-09, 9.6615250E-09, 00038370
C          *-1.0207546E-08, 1.0796633E-08,-1.1393033E-08, 1.2049873E-08, 00038380
C          *-1.2708789E-08, 1.3446466E-08,-1.4174300E-08, 1.5005577E-08, 00038390
C          *-1.5807160E-08, 1.6747136E-08,-1.7625961E-08, 1.8693427E-08, 00038400
C          *-1.9650840E-08, 2.0869789E-08,-2.1903555E-08, 2.3305308E-08, 00038410
C          *-2.4407377E-08, 2.6033678E-08,-2.7186773E-08, 2.9094334E-08, 00038420
C          *-3.0266804E-08, 3.2534013E-08,-3.3672072E-08, 3.6408936E-08, 00038430
C          *-3.7425022E-08, 4.0787921E-08,-4.1543242E-08, 4.5756842E-08, 00038440
C          *-4.6035233E-08, 5.1425075E-08,-5.0893896E-08, 5.7934897E-08, 00038450
C          *-5.6086570E-08, 6.5475248E-08,-6.1539913E-08, 7.4301996E-08, 00038460
C          *-6.7117043E-08, 8.4767837E-08,-7.2583120E-08, 9.7366568E-08, 00038470
C          *-7.7553611E-08, 1.1279873E-07,-8.1416723E-08, 1.3206914E-07, 00038480
C          *-8.3217217E-08, 1.5663185E-07,-8.1482581E-08, 1.8860593E-07, 00038490
C          *-7.3963141E-08, 2.3109673E-07,-5.7243707E-08, 2.8867452E-07, 00038500
C          *-2.6163525E-08, 3.6808773E-07, 2.7049871E-08, 4.7932617E-07, 00038510
C          * 1.1407365E-07, 6.3720626E-07, 2.5241961E-07, 8.6373487E-07, 00038520
C          * 4.6831433E-07, 1.1916346E-06, 8.0099716E-07, 1.6696015E-06, 00038530
C          * 1.3091334E-06, 2.3701475E-06, 2.0803829E-06, 3.4012978E-06/ 00038540
C          DATA WBO/                                             00038550
C          * 3.2456774E-06, 4.9240402E-06, 5.0005198E-06, 7.1783540E-06, 00038560
C          * 7.6367633E-06, 1.0522038E-05, 1.1590021E-05, 1.5488635E-05, 00038570
C          * 1.7510398E-05, 2.2873836E-05, 2.6368006E-05, 3.3864387E-05, 00038580
C          * 3.9610390E-05, 5.0230379E-05, 5.9397373E-05, 7.4612122E-05, 00038590
  
```

```

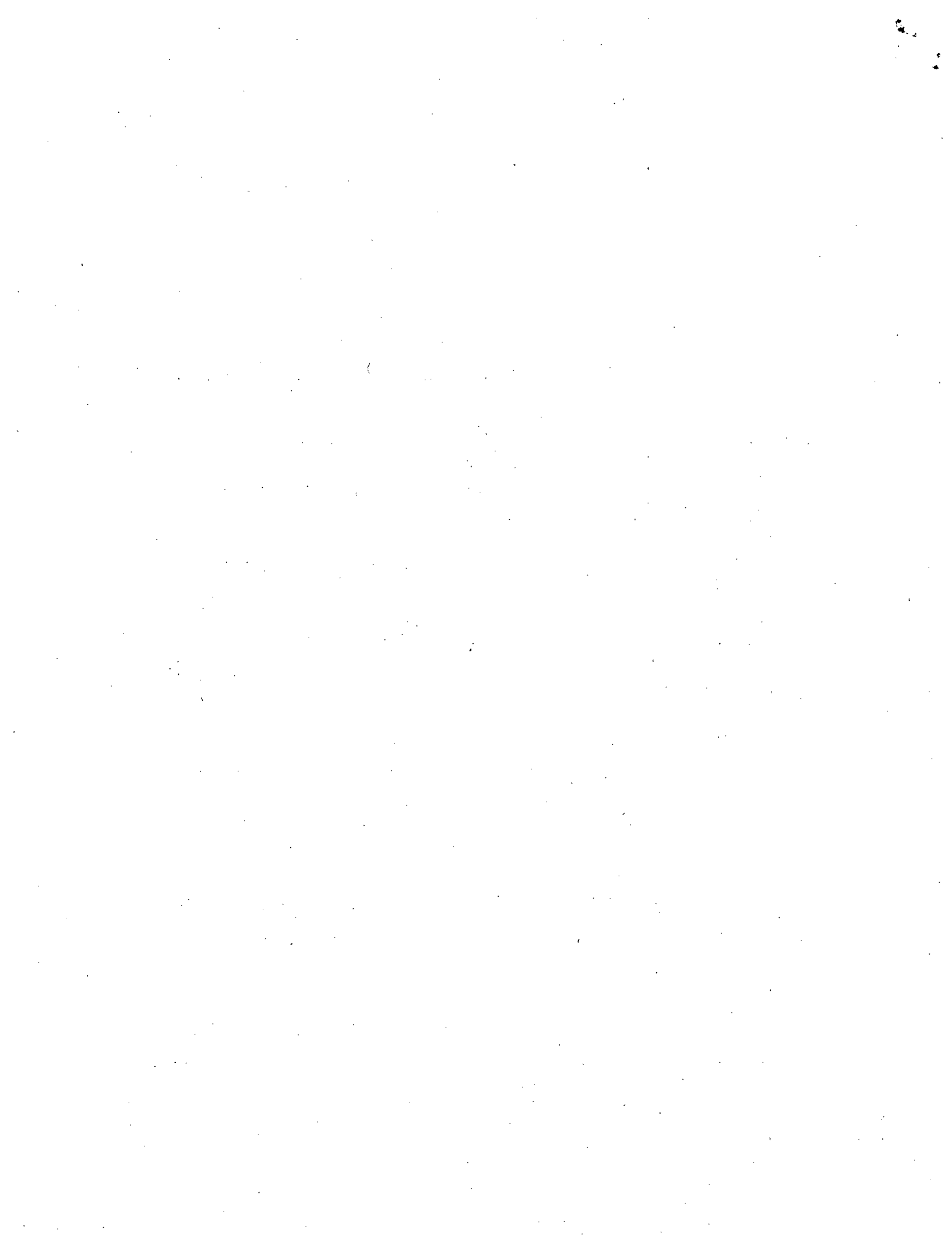
* 8.8951409E-05, 1.1094809E-04, 1.3308026E-04, 1.6511335E-04, 00038610
* 1.9895671E-04, 2.4587195E-04, 2.9728181E-04, 3.6629770E-04, 00038620
* 4.4402013E-04, 5.4589361E-04, 6.6298832E-04, 8.1375348E-04, 00038630
* 9.8971624E-04, 1.2132772E-03, 1.4772052E-03, 1.8092022E-03, 00038640
* 2.2045122E-03, 2.6980811E-03, 3.2895354E-03, 4.0238764E-03, 00038650
* 4.9080203E-03, 6.0010999E-03, 7.3216878E-03, 8.9489225E-03, 00038660
* 1.0919448E-02, 1.3340696E-02, 1.6276399E-02, 1.9873311E-02, 00038670
* 2.4233627E-02, 2.9555699E-02, 3.5990069E-02, 4.3791529E-02, 00038680
* 5.3150319E-02, 6.4341372E-02, 7.7506720E-02, 9.2749987E-02, 00038690
* 1.0980561E-01, 1.2791555E-01, 1.4525830E-01, 1.5820085E-01, 00038700
* 1.6058576E-01, 1.4196085E-01, 8.9781222E-02, -1.0238278E-02, 00038710
* -1.5083434E-01, -2.9059573E-01, -2.9105437E-01, -3.7973244E-02, 00038720
* 3.8273717E-01, 2.2014118E-01, -4.7342635E-01, 1.9331133E-01, 00038730
* 5.3839527E-02, -1.1909845E-01, 9.9317051E-02, -6.6152628E-02, 00038740
* 4.0703241E-02, -2.4358316E-02, 1.4476533E-02, -8.6198067E-03, 00038750
DATA WCO/
* 5.1597053E-03, -3.1074602E-03, 1.8822342E-03, -1.1456545E-03, 00038760
* 7.0004347E-04, -4.2904226E-04, 2.6354444E-04, -1.6215439E-04, 00038780
* 9.9891279E-05, -6.1589037E-05, 3.7996921E-05, -2.3452250E-05, 00038790
* 1.4479572E-05, -8.9417427E-06, 5.5227518E-06, -3.4114252E-06, 00038800
* 2.1074101E-06, -1.3019229E-06, 8.0433617E-07, -4.9693681E-07, 00038810
* 3.0702417E-07, -1.8969219E-07, 1.1720069E-07, -7.2412496E-08, 00038820
* 4.4740283E-08, -2.7643004E-08, 1.7079403E-08, -1.0552634E-08, 00038830
* 6.5200311E-09, -4.0284597E-09, 2.4890232E-09, -1.5378695E-09, 00038840
* 9.5019040E-10, -5.8708696E-10, 3.6273937E-10, -2.2412348E-10, 00038850
* 1.3847792E-10, -8.5560821E-11, 5.2865474E-11, -3.2664392E-11, 00038860
* 2.0182948E-11, -1.2470979E-11, 7.7057678E-12, -4.7611713E-12, 00038870
* 2.9415274E-12, -1.8170081E-12, 1.1221034E-12, -6.9271067E-13, 00038880
* 4.2739744E-13, -2.6344388E-13, 1.6197105E-13, -9.9147443E-14, 00038890
* 6.0487998E-14, -3.6973097E-14, 2.2817964E-14, -1.4315547E-14, 00038900
* 9.1574735E-15, -5.9567236E-15, 3.9209969E-15, -2.5911739E-15, 00038910
* 1.6406939E-15, -8.8248590E-16, 3.0195409E-16, 2.2622634E-17, 00038920
* -8.0942556E-17, -3.7172363E-17, 1.9299542E-16, -3.3388160E-16, 00038930
* 4.6174116E-16, -5.8627358E-16, 7.2227767E-16, -8.7972941E-16, 00038940
* 1.0211793E-15, -1.0940039E-15, 1.0789555E-15, -9.7089714E-16, 00038950
DATA WDO/
* 7.4110927E-16, -4.1700094E-16, 8.5977184E-17, 1.3396469E-16, 00038970
* -1.7838410E-16, 4.8975421E-17, 1.9398153E-16, -5.0046989E-16, 00038980
* 8.3280985E-16, -1.1544640E-15, 1.4401527E-15, -1.6637066E-15, 00038990
* 1.7777129E-15, -1.7322187E-15, 1.5247247E-15, -1.1771155E-15, 00039000
* 6.9747910E-16, -1.2088956E-16, -4.8382957E-16, 1.0408292E-15, 00039010
* -1.5220450E-15, 1.9541597E-15, -2.4107448E-15, 2.9241438E-15, 00039020
* -3.5176475E-15, 4.2276125E-15, -5.0977851E-15, 6.1428456E-15, 00039030
* -7.3949962E-15, 8.8597601E-15, -1.0515959E-14, 1.2264584E-14, 00039040
* -1.3949870E-14, 1.5332490E-14, -1.6146782E-14, 1.6084121E-14, 00039050
* -1.4962523E-14, 1.2794804E-14, -9.9286701E-15, 6.8825809E-15, 00039060
* -4.0056107E-15, 1.5965079E-15, -7.2732961E-18, -4.0433218E-16, 00039070
* -6.5679655E-16, 3.3011866E-15, -7.3545910E-15, 1.2394851E-14, 00039080
* -1.7947697E-14, 2.3774303E-14, -3.0279168E-14, 3.9252831E-14, 00039090
* -5.5510504E-14, 9.0505371E-14, -1.7064873E-13, 00039100

```

C--END OF JO FILTER WEIGHTS

C

00039110  
 00039120



```

C---J1-TRANSFORM FILTER WEIGHT ARRAYS (EQUIVALENT TO WT1 ARRAY)
DATA WA1/
*-4.2129715E-16, 5.3667031E-15, -7.1183962E-15, 8.9478500E-15,
*-1.0767891E-14, 1.2362265E-14, -1.3371129E-14, 1.3284178E-14,
*-1.1714302E-14, 8.4134738E-15, -3.7726725E-15, -1.4263879E-15,
* 6.1279163E-15, -9.1102765E-15, 9.9696405E-15, -9.3649955E-15,
* 8.6009018E-15, -8.9749846E-15, 1.1153987E-14, -1.4914821E-14,
* 1.9314024E-14, -2.3172388E-14, 2.5605477E-14, -2.6217555E-14,
* 2.5057768E-14, -2.2485539E-14, 1.9022752E-14, -1.5198084E-14,
* 1.1422464E-14, -7.9323958E-15, 4.8421406E-15, -2.1875032E-15,
*-3.2177842E-17, 1.8637565E-15, -3.3683643E-15, 4.6132219E-15,
*-5.6209538E-15, 6.4192841E-15, -6.8959928E-15, 6.9895792E-15,
*-6.5355935E-15, 5.6125163E-15, -4.1453931E-15, 2.6358827E-15,
*-9.5104370E-16, 1.4600474E-16, 5.6166519E-16, 8.2899246E-17,
* 5.0032100E-16, 4.3752205E-16, 2.1052293E-15, -9.5451973E-16,
* 6.4004437E-15, -2.1926177E-15, 1.1651003E-14, 5.8415433E-16,
* 1.8044664E-14, 1.0755745E-14, 3.0159022E-14, 3.3506138E-14,
* 5.8709354E-14, 8.1475200E-14, 1.2530006E-13, 1.8519112E-13,
* 2.7641786E-13, 4.1330823E-13, 6.1506209E-13, 9.1921659E-13,
* 1.3698462E-12, 2.0447427E-12, 3.0494477E-12, 4.5501001E-12,
* 6.7870250E-12, 1.0126237E-11, 1.5104976E-11, 2.2536053E-11/
DATA WB1/
* 3.3617368E-11, 5.0153839E-11, 7.4818173E-11, 1.1161804E-10,
* 1.6651222E-10, 2.4840923E-10, 3.7058109E-10, 5.5284353E-10,
* 8.2474468E-10, 1.2303750E-09, 1.8355034E-09, 2.7382502E-09,
* 4.0849867E-09, 6.0940898E-09, 9.0913020E-09, 1.3562651E-08,
* 2.0233058E-08, 3.0184244E-08, 4.5029477E-08, 6.7176304E-08,
* 1.0021488E-07, 1.4950371E-07, 2.2303208E-07, 3.3272689E-07,
* 4.9636623E-07, 7.4049804E-07, 1.1046805E-06, 1.6480103E-06,
* 2.4585014E-06, 3.6677163E-06, 5.4714550E-06, 8.1626422E-06,
* 1.2176782E-05, 1.8166479E-05, 2.7099223E-05, 4.0428804E-05,
* 6.0307294E-05, 8.9971508E-05, 1.3420195E-04, 2.0021123E-04,
* 2.9860417E-04, 4.4545291E-04, 6.6423156E-04, 9.9073275E-04,
* 1.4767050E-03, 2.2016806E-03, 3.2788147E-03, 4.8837292E-03,
* 7.2596811E-03, 1.0788355E-02, 1.5973323E-02, 2.3612041E-02,
* 3.4655327E-02, 5.0608141E-02, 7.2827752E-02, 1.0337889E-01,
* 1.4207357E-01, 1.8821315E-01, 2.2996815E-01, 2.5088500E-01,
* 2.0334626E-01, 6.0665451E-02, -2.0275683E-01, -3.5772336E-01,
*-1.8280529E-01, 4.7014634E-01, 7.2991233E-03, -3.0614594E-01,
* 2.4781735E-01, -1.1149185E-01, 2.5985386E-02, 1.0850279E-02,
*-2.2830217E-02, 2.4644647E-02, -2.2895284E-02, 2.0197032E-02/
DATA WC1/
*-1.7488968E-02, 1.5057670E-02, -1.2953923E-02, 1.1153254E-02,
*-9.6138436E-03, 8.2952090E-03, -7.1628361E-03, 6.1882910E-03,
*-5.3482055E-03, 4.6232056E-03, -3.9970542E-03, 3.4560118E-03,
*-2.9883670E-03, 2.5840861E-03, -2.2345428E-03, 1.9323046E-03,
*-1.6709583E-03, 1.4449655E-03, -1.2495408E-03, 1.0805480E-03,
*-9.3441130E-04, 8.0803899E-04, -6.9875784E-04, 6.0425624E-04,
*-5.2253532E-04, 4.5186652E-04, -3.9075515E-04, 3.3790861E-04,
*-2.9220916E-04, 2.5269019E-04, -2.1851585E-04, 1.8896332E-04,
*-1.6340753E-04, 1.4130796E-04, -1.2219719E-04, 1.0567099E-04,
*-9.1379828E-05, 7.9021432E-05, -6.8334412E-05, 5.9092726E-05,
  
```



```
*-5.1100905E-05, 4.4189914E-05,-3.8213580E-05, 3.3045496E-05, 00039650
*-2.8576356E-05, 2.4711631E-05,-2.1369580E-05, 1.8479514E-05, 00039660
*-1.5980307E-05, 1.3819097E-05,-1.1950174E-05, 1.0334008E-05, 00039670
*-8.9364160E-06, 7.7278366E-06,-6.6827083E-06, 5.7789251E-06, 00039680
*-4.9973715E-06, 4.3215167E-06,-3.7370660E-06, 3.2316575E-06, 00039690
*-2.7946015E-06, 2.4166539E-06,-2.0898207E-06, 1.8071890E-06, 00039700
*-1.5627811E-06, 1.3514274E-06,-1.1686576E-06, 1.0106059E-06, 00039710
*-8.7392952E-07, 7.5573750E-07,-6.5353002E-07, 5.6514528E-07, 00039720
*-4.8871388E-07, 4.2261921E-07,-3.6546333E-07, 3.1603732E-07/ 00039730
```

DATA WD1/ 00039740

```
*-2.7329579E-07, 2.3633470E-07,-2.0437231E-07, 1.7673258E-07, 00039750
*-1.5283091E-07, 1.3216174E-07,-1.1428792E-07, 9.8831386E-08, 00039760
*-8.5465227E-08, 7.3906734E-08,-6.3911437E-08, 5.5267923E-08, 00039770
*-4.7793376E-08, 4.1329702E-08,-3.5740189E-08, 3.0906612E-08, 00039780
*-2.6726739E-08, 2.3112160E-08,-1.9986424E-08, 1.7283419E-08, 00039790
*-1.4945974E-08, 1.2924650E-08,-1.1176694E-08, 9.6651347E-09, 00039800
*-8.3580023E-09, 7.2276490E-09,-6.2501673E-09, 5.4048822E-09, 00039810
*-4.6739154E-09, 4.0418061E-09,-3.4951847E-09, 3.0224895E-09, 00039820
*-2.6137226E-09, 2.2602382E-09,-1.9545596E-09, 1.6902214E-09, 00039830
*-1.4616324E-09, 1.2639577E-09,-1.0930164E-09, 9.4519327E-10, 00039840
*-8.1736202E-10, 7.0681930E-10,-6.1122713E-10, 5.2856342E-10, 00039850
*-4.5707937E-10, 3.9526267E-10,-3.4180569E-10, 2.9557785E-10, 00039860
*-2.5560176E-10, 2.2103233E-10,-1.9113891E-10, 1.6528994E-10, 00039870
*-1.4294012E-10, 1.2361991E-10,-8.2740936E-11/ 00039880
```

C---END OF J1 FILTER WEIGHTS 00039890

```
C 00039900
  NONE=0 00039910
  IF(NEW.EQ.0) GO TO 100 00039920
  NSAVE=0 00039930
```

```
C-----INITIALIZE KERNEL ABSCISSA GENERATION FOR GIVEN B 00039940
  Y1=0.735885266147979446000/DBLE(B) 00039950
  100 ZHANKS=(0.0,0.0) 00039960
  CMAX=(0.0,0.0) 00039970
  NF=0 00039980
  Y=Y1 00039990
```

```
C-----BEGIN RIGHT-SIDE CONVOLUTION AT WEIGHT 131 (EITHER NEW=1 OR 0) 00040000
  ASSIGN 110 TO M 00040010
  I=131 00040020
  Y=Y*E 00040030
  GO TO 200 00040040
  110 TMAX(1)=AMAX1(ABS(T(1)),TMAX(1)) 00040050
  TMAX(2)=AMAX1(ABS(T(2)),TMAX(2)) 00040060
  I=I+1 00040070
  Y=Y*E 00040080
  IF(I.LE.149) GO TO 200 00040090
  IF(TMAX(1).EQ.0.0.AND.TMAX(2).EQ.0.0) NONE=1 00040100
```

```
C-----ESTABLISH TRUNCATION CRITERION (CMAX=CMPLX(TMAX(1),TMAX(2))) 00040110
  CMAX=TOL*CMAX 00040120
  ASSIGN 120 TO M 00040130
  GO TO 200 00040140
```

```
C-----CHECK FOR FILTER TRUNCATION AT RIGHT END 00040150
  120 IF(ABS(T(1)).LE.TMAX(1).AND.ABS(T(2)).LE.TMAX(2)) GO TO 130 00040160
```

```

I=I+1
Y=Y*E
IF(I,LE,283) GO TO 200
130 Y=Y1
C-----CONTINUE WITH LEFT-SIDE CONVOLUTION AT WEIGHT 130
I=130
ASSIGN 140 TO M.
GO TO 200
C-----CHECK FOR FILTER TRUNCATION AT LEFT END
140 IF(ABS(T(1)),LE,TMAX(1),AND,ABS(T(2)),LE,TMAX(2),AND,
* NONE,EQ,0) GO TO 190
I=I-1
Y=Y*E
IF(I,GT,0) GO TO 200
C-----RETURN WITH ISAVE=1 PRESET FOR POSSIBLE NEW=0 USE.
190 ISAVE=1
C-----NORMALIZE BY B TO ACCOUNT FOR INTEGRATION RANGE CHANGE
ZHANKS=ZHANKS/B
RETURN
C-----SAVE/RETRIEVE PSEUDO-SUBROUTINE (CALL FUN ONLY WHEN NECESSARY)
200 G=SNGL(Y)
IF(NEW) 300,210,300
210 IF(ISAVE,GT,NSAVE) GO TO 300
ISAVE=ISAVE
220 IF(G,EQ,GSAVE(ISAVE)) GO TO 240
ISAVE=ISAVE+1
IF(ISAVE,LE,NSAVE) GO TO 220
ISAVE=ISAVE0
C-----G NOT IN COMMON/SAVE/----- EVALUATE FUN.
GO TO 300
C-----G FOUND IN COMMON/SAVE/----- USE FSAVE AS GIVEN.
240 C=FSAVE(ISAVE)
ISAVE=ISAVE+1
C-----SWITCH ON ORDER N
250 IF(N) 270,260,270
260 C=C*WTO(I)
GO TO 280
270 C=C*WT1(I)
280 ZHANKS=ZHANKS+C
GO TO M,(110,120,140)
C-----DIRECT FUN EVALUATION (AND ADD TO END OF COMMON/SAVE/)
300 NSAVE=NSAVE+1
C=FUN(G)
NF=NF+1
FSAVE(NSAVE)=C
GSAVE(NSAVE)=G
GO TO 250
END
SUBROUTINE ERRMSG(M50,M5,16,19)
C---ERROR MESSAGE WRITE ROUTINE AND STOP, WHERE----
C

```

```

00040170
00040180
00040190
00040200
00040210
00040220
00040230
00040240
00040250
00040260
00040270
00040280
00040290
00040300
00040310
00040320
00040330
00040340
00040350
00040360
00040370
00040380
00040390
00040400
00040410
00040420
00040430
00040440
00040450
00040460
00040470
00040480
00040490
00040500
00040510
00040520
00040530
00040540
00040550
00040560
00040570
00040580
00040590
00040600
00040610
00040620
00040630
00040640
00040650
00040660
00040670

```

```

C      MSG=      ANY MULTIPLE OF 5 CHARACTERS--MAX. OF 120      00040680
C              (USE NH----- FORM FOR ANSI COMPATABILITY)      00040690
C      M5=      NO.CHARS IN MSG/5 (REMAINDER MUST BE 0) 1.LE.M5.LE.24 00040700
C      I6=      1ST UNIT FOR WRITE(I6, ) MSG -- USUALLY I6=6 FOR LPT. 00040710
C              IF I6.LE.0 UNIT I6 IGNORED.      00040720
C      I9=      2ND UNIT FIR WRITE(I9, ) MSG --      00040730
C              IF I9.LE.0, UNIT I9 IGNORED.      00040740
C--MESSAGE WRITTEN IN FORM--      00040750
C      /ERROR--MSG HERE      00040760
C      00040770
C      DIMENSION MSG(30)      00040780
C      J=5*M5      00040790
C      K=J/4+MOD(J,4)      00040800
C      IF(I6.GT.0) WRITE(I6,10) (MSG(I),I=1,K)      00040810
10  FORMAT(/8H ERROR--,30A4)      00040820
C      IF(I9.GT.0) WRITE(I9,10) (MSG(I),I=1,K)      00040830
C      CALL CLOSE_FILE('-ALL')      00040840
C      00040850
C      STOP      00040860
C      END      00040870

      SUBROUTINE WARN(MSG,M5,I6,I9,*)      00040880
C--WARNING MESSAGE WRITE ROUTINE WHERE:      00040890
C      00040900
C      MSG=      'ANY MULTIPLE OF 5 CHARACTERS--MAX. OF 120      00040910
C      M5=      NO.CHAR'S IN MSG/5 (REMAINDER MUST BE 0) 1<=M5<=24 00040920
C      I6=      1ST UNIT NO. FOR WRITE(I6, )MSG--USUALLY I6=6 FOR LPT. 00040930
C              IF I6<=0 UNIT I6 IGNORED.      00040940
C      I9=      2ND UNIT NO. FOR WRITE(I9, )MSG--IF I9<=0, I9 IGNORED 00040950
C      *=      $LABEL NO. TO RETURN AFTER ERROR CALLED ($NO. MUST BE 00040960
C              GIVEN      00040970
C      00040980
C      DIMENSION MSG(30)      00040990
C      J=5*M5      00041000
C      K=J/4+MOD(J,4)      00041010
C      IF(I6.GT.0) WRITE(I6,6) (MSG(I),I=1,K)      00041020
6  FORMAT('OWARNING--',30A4)      00041030
C      IF(I9.GT.0) WRITE(I9,6) (MSG(I),I=1,K)      00041040
C      RETURN (1)      00041050
C      END      00041060

```

Note: Subprogram DUMMY\_FCODE is referenced as the second external parameter in the CALL MARKRT, but DUMMY\_FCODE will never be called since \$parms ider=1 should be used (because analytic derivatives are not used in IMSLMQ, the code subprogram was not needed). For systems requiring all

```

C---NON-IMSL MAIN PROGRAM USING MARKRT (ANDERSON, 1979)
EXTERNAL FCODE,DUMMY_FCODE,ITHXYZ,ETHXYZ
CALL MARKRT(FCODE,DUMMY_FCODE,ITHXYZ,ETHXYZ)
STOP
END

```

6. To replace the IMSL interface routines (IMSLMQ, FFXSSQ, LNXXSQ, and all calls to the IMSL library) with the nonlinear least squares subprogram MARKRT (available in Anderson, 1979c), replace the main program (lines 0000010-0000140) with the following code:

5. Multics names greater than 6-characters (e.g. MQLVTHXYZ\_SUBZ, MQLVTHXYZ\_SUBEND, etc) should be renamed to 6 or less characters for most other systems.

4. Subprograms ERKMSG and WARN should be changed according to the number of characters per word of the target machine (note that 4 char/word uses format A4 on the Honeywell Multics system; however, 5 char/word is assumed in the input parameter array MSG). Similar changes should be made, if necessary, to other character arrays and format statements (e.g., see subroutine IMSLMQ, array TITLE and FMT).

3. All Multics exponent underflow messages are suppressed and the result set to 0.0. An equivalent method should be used for other systems.

```

character*n (replace with logical*n or delete)
call open_ (delete)
call close_ (delete)
exp_ (replace with exp)
dex_ (replace with dex)
cexp_ (replace with cexp)
call initref (delete)

```

2. Any of the following Multics statements and/or calls should be deleted or replaced if converting to another system:

1. All lower-case letters used for parameters and Fortran names in this report should be changed to upper-case letters for most other systems.

Appendix 2. --- Conversion to other systems

external references to be available (even if not called),  
 the following subroutine could be used:

```

SUBROUTINE DUMMY_FCODE(A,B,C,D,E,I,J,K)
  PRINT /USE IDEF=1, STOP
END
  
```

If converting to subroutine MARKRT, all parameters  
 prefixed by an "\*" apply; however, parameters  
 IORT,PARM(),INSID, EPS,DELTA, and MAXFN cannot be used in  
 this case. In addition to subroutine MARKRT, the following  
 subprograms from Anderson (1979c) are also called: GJR,  
 UNSCAL, ASINH, and ERMSSG.

Appendix 3. -- Test problem input/output listings

The following input file (file05) was used to run a test problem on a Honeywell Multics system. Note that file10 is not required because \$parms ialt=5. The output listing (file16) follows beginning on the next page.

file05

```
mlvthxyz test problem2, model parms b0=1,1,1,1,1,150
$parms n=15,lf=1,lb=1,sp=1,ss=1,m=2,wt=1,prt=-2,ialt=5,k=6,
b=1,1,1,1,1,1,250$
(2e14.5,f13.0,e14.5).
```

```
0.23010e-07 0.50000e-02 2.
0.46094e-07 0.19905e-01 2.
0.86019e-07 0.79245e-01 2.
0.81049e-07 0.79245e+00 2.
0.80052e-07 0.19905e+01 2.
0.30237e-08 0.79245e-02 3.
0.15069e-07 0.31548e-01 3.
0.71822e-07 0.19905e+00 3.
0.79080e-07 0.79245e+00 3.
0.79539e-07 0.31548e+01 3.
0.10003e+01 0.158 e+02 4.
0.10601e+01 0.1 e+03 4.
0.155 e+01 0.252 e+03 4.
0.311 e+01 0.63 e+03 4.
0.426 e+01 0.1 e+04 4.
$init nf=5,mm=2,x=0,v=1000,istep=1,lob=4$
0.23 0.469 0.86 0.81 0.8 0.3 0.15 0.72 0.79 0.8 e-07
0.155 0.31 e+01 0.43 e+01
```

IT H X Y Z-- malvthxyz test problem2, model parms b0=1,1,1,1,1,250

data set parms are

istep= 1 x= 0.0000E+00 y= 0.1000E+04 r= 0.1000E+04 current= 0.1000E+01  
source length= 0.1000E+01 l= 1  
ifltx= 0 iflty= 0 ifltz= 0

control parms are

tol= 0.1000E-03 nf = 5 iob = 4  
method= 2

ii m s l m a -- malvthxyz test problem2, model parms b0=1,1,1,1,1,250

n= 15 k= 6 ip= 1 m= 2 e= 0.000E+00  
ialt= 5 istop= 1 iwt= 1 niter= 10 scaler= 1  
iprt= -2 iopt= 1 nsid= 3 maxfn= 120 eps= 0.000E+00  
delta= 0.000E+00  
parm= 0.100E-01 0.200E+01 0.120E+03 0.100E+00

ib= 1

fmt=(2e14.5,f13.0,e14.5)

initial parameters

0.10000000E+01 0.10000000E+01 0.10000000E+01 0.10000000E+01 0.10000000E+00  
0.25000000E+03

parameter index: 1 2 3 4 5 6  
reordered as...: 2 3 4 5 6

reordered parameters

0.10000000E+01 0.10000000E+01 0.10000000E+01 0.10000000E+00 0.25000000E+03  
Warning--marquardt parameter exceeded parm(3)

\*\*\*: imslm convergence information:

norm of gradient 0.16253650E-02  
 function evaluations 0.46000000E+02  
 est. sign. digits 0.45923176E+01  
 marquardt parameter 0.26165364E+03  
 no. iterations 0.70000000E+01  
 type convergence (infer) 0  
 error code (ier) 131  
 residual sum-of-squares (ssq)= 0.41113755E-03

\*\*\*\* final unscaled parameters

0.16042551E+01 0.99890090E+00 0.99777942E+00 0.98586691E-01 0.15074997E+03

scaled gradient

-0.26512254E-04 -0.25798774E-03 0.16032409E-02 0.27969514E-04 0.58197892E-04

i	obs.y(i)	cal	res	x(i,1)	x(i,2)	x(i,3)	x(i,4)	wt(i)
1	0.230100E-07	0.226074E-07	0.403E-09	0.500000E-02	0.200000E+01	0.230000E-07	0.000000E+00	0.189036E+16
2	0.460940E-07	0.463114E-07	-0.217E-09	0.199050E-01	0.200000E+01	0.469000E-07	0.000000E+00	0.454626E+15
3	0.860190E-07	0.863804E-07	-0.361E-09	0.792450E-01	0.200000E+01	0.860000E-07	0.000000E+00	0.135208E+15
4	0.810490E-07	0.813849E-07	-0.336E-09	0.792450E+00	0.200000E+01	0.810000E-07	0.000000E+00	0.152416E+15
5	0.800520E-07	0.803887E-07	-0.337E-09	0.199050E+01	0.200000E+01	0.800000E-07	0.000000E+00	0.156250E+15
6	0.302370E-08	0.303244E-08	-0.874E-11	0.792450E-02	0.300000E+01	0.300000E-08	0.000000E+00	0.111111E+18
7	0.150690E-07	0.150673E-07	0.165E-11	0.315480E-01	0.300000E+01	0.150000E-07	0.000000E+00	0.444444E+16
8	0.718220E-07	0.717602E-07	0.618E-10	0.199050E+00	0.300000E+01	0.720000E-07	0.000000E+00	0.192901E+15
9	0.790800E-07	0.789986E-07	0.814E-10	0.792450E+00	0.300000E+01	0.790000E-07	0.000000E+00	0.160231E+15
10	0.795390E-07	0.794522E-07	0.868E-10	0.315480E+01	0.300000E+01	0.800000E-07	0.000000E+00	0.156250E+15
11	0.100030E+01	0.100249E+01	-0.219E-02	0.158000E+02	0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01
12	0.106010E+01	0.106171E+01	-0.161E-02	0.100000E+03	0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01
13	0.155000E+01	0.154799E+01	0.201E-02	0.252000E+03	0.400000E+01	0.155000E+01	0.000000E+00	0.416233E+00
14	0.311000E+01	0.310224E+01	0.776E-02	0.630000E+03	0.400000E+01	0.310000E+01	0.000000E+00	0.104058E+00
15	0.426000E+01	0.426851E+01	-0.851E-02	0.100000E+04	0.400000E+01	0.430000E+01	0.000000E+00	0.540833E-01

\*\*\*\* rmserr= 0.64120007E-02

final scaled partials (Jacobian)

1	-0.98300802E+00	0.00000000E+00	0.45169752E+00	-0.21029400E-03	0.32285083E-02
2	-0.98752421E+00	0.00000000E+00	0.54289651E+00	0.18732516E-01	0.20704610E+00
3	-0.10040869E+01	0.00000000E+00	0.15933372E+00	0.84025984E-02	0.10631049E+00
4	-0.10045505E+01	0.00000000E+00	-0.14146162E-01	-0.98771935E-02	-0.11320418E-01
5	-0.10046485E+01	0.00000000E+00	-0.43594473E-02	-0.41669784E-02	-0.25581703E-02
6	0.00000000E+00	-0.10110864E+01	0.56004203E+00	0.11198296E-03	0.26295337E-01
7	0.00000000E+00	-0.10040268E+01	0.12456908E+01	0.10079574E+00	0.68332703E+00
8	0.00000000E+00	-0.99655251E+00	0.16540021E+00	0.38473478E-01	0.13312211E+00
9	0.00000000E+00	-0.10003372E+01	0.81988631E-02	0.50908907E-02	0.62373467E-02
10	0.00000000E+00	-0.99281596E+00	0.24915792E-02	0.48387353E-03	0.22351111E-03
11	0.00000000E+00	0.00000000E+00	0.10014908E+01	0.10824785E-03	0.81669775E-03
12	0.00000000E+00	0.00000000E+00	0.98560736E+00	0.14433047E-01	0.16270619E+00
13	0.00000000E+00	0.00000000E+00	0.55119987E+00	0.93325926E-01	0.68588632E+00
14	0.00000000E+00	0.00000000E+00	0.80360779E-01	0.24391849E+00	0.73421075E+00
15	0.00000000E+00	0.00000000E+00	-0.10809307E+00	0.34162687E+00	0.63704752E+00



