GL00416

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.

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.

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.

By James P. Kauahikaua

INTRODUCTION.

Program MQLVTHXYZ is a general purpose program for inversion of time-domain electromagnetic (TDEM) sounding data when a grounded electric wire source is employed. Details of the actual computations are given in other papers (Anderson, 1974, 1975, 1977, 1979a; Kauahikaua and Anderson, 1977; Kauahikaua, 1978). An IMSL (International Mathematical and Statistical Library, 1977) derivative-free Levenburg-Marquardt (Brown-Dennis, 1972) nonlinear least squares subprogram (ZXSSQ) is used for inversion of the TDEM sounding data. See appendix 2 notes for conversion to other computer systems where the IMSL library may not available.

The following program options are currently available:

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

(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 TDEM 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 accomodate other current source waveforms can be done using the user-defined function FLT which is controlled by $init parameters ifltx, iflty, and ifltz.

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,...,n, where n<=200.

k= Total number of parameters (1<=k<=20, k<=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).

ip= 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 ip=0 with the restrictions that ip<k and n>=k-ip.
(cref: $parms k,n,ib(), and b).

m= Number of independent variables (m<=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<=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,...,n will be weighted unity (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<=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 (pcode) 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 pcode is a dummy routine].
(cref: $parms del).

iprt= 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 scalep.
= 1 for detail printout format for each iteration, which includes the parameter changes from the Marquardt algorithm. [note iprt=1 behaves like iprt=-2, unless converting to subprogram MARQRT].
= -1 (recommended if scalep>0 used) for abbreviated printout format for each iteration with printed unscaled values of b(j) but scaled values of phi.
= -2 same as iprt=-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 iprt=-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 fcode 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).

* ff= 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 eps (see below)].
(cref: Marquardt, 1963).

* tau= Convergence criterion test parameter (default 1e-3).
(cref: Marquardt, 1963).

* xl= 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 xl (e.g., 1.0) may

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

* modlam= 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.

* gamcr= Marquardt's critical angle between the gradient and adjustment vectors (default 45.0 degrees). The value of gamcr 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 file16 control.
= 1 (default) for printable ouput to both file06 and file16.
= 0 for print output only on file06.
Note: file16 output may be useful for deferred output when running the job from a time-sharing terminal; also, file16 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 file16.
(cref: $parms iprt).

sp= scalep (equivalent names) is a parameter scaling option.
= 0 (default) to ignore parameter scaling (i.e., to use unscaled 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 scalep=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 arcsinh(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<=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= scaley (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,...,n.
= 2 to scale observations y(i) using 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<=3, the parameter order must be given as:

b(1) is the amplitude shift parameter,

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

b(mm+2),b(mm+3),...,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),...,b(mm+3) are the mm layer conductivities (in mhos/meter), and

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

For iob=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 ip and ib()),

b(4),b(5),b(6) are the orientation parameters, angles theta, phi, and omega, where
znew = z*cos(theta) + sin(theta)*(x*cos(phi) + y*sin(phi))
xnew = xprime*cos(omega) + yprime*sin(omega)
ynew = yprime*cos(omega) - xprime*sin(omega)
where xprime and yprime are defined by
xprime = x*cos(theta) + sin(theta)*(y*sin(phi) - z*cos(phi))
yprime = y*cos(theta) - sin(theta)*(x*sin(phi) + z*cos(phi))

b(7),b(8),...,b(mm+6) are the mm layer conductivities (in mhos/meter), and

b(mm+7),b(mm+8),...,b(2*mm+5) are the mm-1 layer thicknesses in meters.
(cref: $parms k,ip,ib and $init mm,iob).

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

[the following $parms are parameters used only by IMSL subprogram ZXSSQ, and cannot be used if converting to subprogram MARQRT as described in appendix 2].

iopt= 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 iopt=2. The default is parm()=.01,2.,120.,.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, xjtj, by the factor (1.0+parm(1)). A small value gives a Newton step, while a large value gives a steepest descent step. (default parm(1)=.01).

i=2, the scaling factor used to modify the Marquardt parameter, which is decreased by parm(2) after an immediately successful descent direction, and increased by the square of parm(2) if not. (default parm(2)=2 where 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 parm(3) is exceeded. parm(3)>100 is recommended. (default 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 parm(4). Central differencing is good in the vicinity of the solution, so parm(4) should be small. (default parm(4)=.1).
(cref: $parms iopt).

nsig= The first convergence criterion. Convergence is satisfied if on 2 successive iterations, the parameter estimates agree, component by component, to nsig digits. (default nsig=3; using nsig>5 may not converge since single precision is used).

eps= The second convergence criterion. Convergence is statisfied if on 2 successive iterations the residual sum of squares estimates have relative differences <= eps. (default eps=0.0).
(cref: $parms e, which is equivalent to eps).

delta= The third convergence criterion. Convergence is statisfied if the Euclidean norm of the approximate gradient is <= delta. (default delta=0.0).

Note: The Marquardt iteration is terminated, and convergence is considered achieved, if any one of the three convergence conditions (nsig,eps, or 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 maxfn

```
          slightly.   Note:  unless  maxfn>0  is  given,
          maxfn=2*k*niter is used as the default value.
          (cref: $parms k,niter, where default niter=10).


$end      [end of $parms namelist]


$init parameters (with defaults and cross-references):

iob=      observation-type defined for y(i):
        = 1 defines y(i) as x-component  magnetic  field  TDEM
          data, in amps/m (parallel to wire source).
        = 2 defines y(i) as y-component  magnetic  field  TDEM
          data,  in  amps/m  (perpendicular  to  the  wire
          source).
        = 3 defines y(i) as z-component  magnetic  field  TDEM
          data, in amps/m (vertical).
          (note:  for  iob<=3,  m=1  must  also be given in
          $parms).
        = 4 defines mixed observation-type sounding where  the
          i-th  observation  type is given by x(i,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 iob=4  except  that  the  orientation
          parameters  may  be  used.  It is advisable to use
          the iob=5 option only after the best solution  has
          been  obtained  with the appropriate iob<=4.  Then
          when the iob=5 run is made, $parms  sp  should  be
          set  to zero to allow the angles to be positive or
          negative.
          (note: for iob=4,5, m=2  must  also  be  given  in
          $parms).
          (cref:  $parms  m,b(),  $init  mm, and DATA MATRIX
          NOTES).

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

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

method=   selects the numerical technique for computing  the
          step response (only used if $init istep=1).
        = 1 computed as the  integral  from  0  to  t  of  the
          impulse  response  (see  Kauahikaua  and Anderson,
```

1977).

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

ifltx= filter option parameter for the x-component computations (default ifltx=0). If ifltx is different from zero, the magnetic field TDEM response will be computed as the transform of the x-component frequency response multiplied by a user-defined complex function FLT(IFLTX,F) where F is the frequency in Hertz. For multicomponent inversion ($init iob=4,5), ifltx may be used to differentiate between a call to FLT(IFLTX,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.

iflty= similar to ifltx except for the y-component (default iflty=0).

ifltz= similar to ifltx except for the z-component (default ifltz=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 $init parameter icase. (cref: $init icase).

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

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

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

nf= number of frequencies per decade at which to compute the magnetic field frequency response between $init blwr and bupr (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, bupr).

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, intype).

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, intype).

intype= type of integration algorithm (subprogram canc4) and convergence test desired for finite-length wire integration.
=1,2,3 uses Newton-Coates no. 4 quadrature:
= 1 for absolute error test (not generally recommended),
= 2 for "l-one" type error test,
= 3 for "l-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 intype convergence test must be satisfied by both the real and the imaginary parts of the complex integral simultaneously.
(cref: $init cnctol).

nrun= number of spline nodes to be calculated per digital filter interval for the finite-length wire integration (default nrun=1). Increasing nrun 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: (y(i),x(i,j),j=1,m*), where i=row number 1,2,...,n, and m*=m+1 if iwt=1, otherwise m*=m<=2. The data matrix is read on logical unit ialt (default 10) using an object-time format statement (see any Fortran manual). The number of items read depends on $parms m,iwt and $init iob as previously defined. The various data matrix options are summarized as follows:

(a) Specific observation component, TDEM sounding (iob<=3, m=1, and max. 3 items per record):

1. y(i)= i-th observation, where $init iob<=3 defines the particular component.
2. x(i,1)= i-th time (x(i,1)>0.0 seconds).
3. x(i,2)= standard deviation of observation i (include only if iwt=1).

(b) Mixed observation types: x-, y-, or z-component TDEM soundings and/or Schlumberger sounding data (iob=4 or 5, m=2, and max. 4 items per record):

1. y(i)= i-th observation (where actual type is defined by x(i,2)).
2. x(i,1)= i-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 y(i); use x(i,2)= 1.0 for x-component TDEM data, =2.0 for y-component TDEM data, =3.0 for z-component TDEM data, or =4.0 for Schlumberger apparent resistivity data.
4. x(i,3)= standard deviation of observation i (include only if iwt=1). Note: for joint inversion of Schlumberger apparent resistivity and TDEM data, a weighted least squares should be used (iwt=1 option) to produce residuals of near-equal magnitude.

EXAMPLES OF INPUT PARAMETERS AND DATA ORDERING.

1. Vertical magnetic field TDEM sounding (default iob=3, istep=0):

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

2. Mixed observation types (iob=4), y-component TDEM (step response) and Schlumberger sounding data, and weighted observations (iwt=1):

```
example 2
$parms n=100,k=8,m=2,iprt=-2,sp=1,iwt=1,ialt=5,ip=2,ib=1,3,
 b=1e0,1e9,1e0,.01,.2,.01,100,200$
(4f10.0)
20.1      .05       2.        .05
29.87     .06       2.        .05
--(etc. for rest of TDEM sounding data)--
108.      10.       4.        5.
120.      13.       4.        5.
--(etc. for rest of Schlumberger sounding data)--
$init mm=3,iob=4,icase=1,istep=1,y=1000$
```

3. Vertical magnetic field TDEM sounding (default istep=0), orientation option desired:

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

SPECIAL OBJECT FORMAT PHRASES.

If an existing data matrix file does not have the properly defined column ordering in the form (y(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 y(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:
   >udd>Emodl_inv>JKauahikaua>exec_segs,
   >udd>Emodl_inv>WAnderson>lib_em,
   >udd>Emodl_inv>WAnderson>lib_1, 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:
   io attach file05 vfile_ parameter_file_name

3. Attach "file10" to an input data-matrix ascii file if ialt=10 (default) is used. If ialt=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:
   io attach file10 vfile_ data_file_name

4. Set the underflow condition handler off by typing:
   set_ufl -off

5. Execute program MQLVTHXYZ by typing: mqlvthxyz

If file05 was not attached, then the user must anticipate the required title, $parms, object format, and $init 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 dprinted 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_ufl -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 (mm) chosen. First check that all $parms, $init, 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.

__________, 1975, Improved digital filters for evaluating 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, pg. 1287-1305.

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

__________, 1979c, Program MARQLOOPS: 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 analogues 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.

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

Kauahikaua, J. and Anderson, W.L., 1977, Calculation of standard transient and frequency sounding 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. NTIS, Springfield, Va. 22161 as Rept. PB-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.

Patterson, 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 listing

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

Source Availability

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 bpi density, 80-character records (unblocked card images), and contained on one file.

Note: The source code for all IMSL Library routines used (ZXSSQ, LEQT1P, LUDECP, LUELMP, LINV1P, and UERTST) are only available from International Mathematical and Statistical Libraries (1977), whose address is given in the references.

Copyright Notices

(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 CQUAD, CQSUB, and CQSUBA 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.

```
C*****      PROGRAM MQLVTHXYZ      *****
C  NONLINEAR LEAST SQUARES INVERSION OF THREE COMPONENT
C  MAGNETIC FIELD (TDEM) DATA USING A MODIFIED
C  MARQUARDT-LEVENBURG ALGORITHM (FROM IMSL LIBRARY)
C
C  WRITTEN BY JIM KAUAHIKAUA, USGS, DENVER, COLORADO
C  LAST MODIFIED: NOVEMBER, 1978
C
      EXTERNAL ITHXYZ,ETHXYZ
      CALL INITREF('>UDD>EMODL_INV>JKAUAHIKAUA>ARCHIVES','HXYZINV_OLIB',
     1 'FCODE')
      CALL IMSLMQ(ITHXYZ,ETHXYZ)
      STOP
      END

      SUBROUTINE ETHXYZ(Y,X,B,K,NN,TITLE,IOUT)
C
C  TERMINATION ROUTINE FOR COMPUTER INVERSION OF
C  THREE-COMPONENT MAGNETIC FIELD TDEM DATA
C
      CHARACTER*5 TITLE
      DIMENSION Y(1),X(200,5),B(1),TITLE(16)
      COMMON/CXYZ/A1,A2,J1,J2,JN(3),A6,A7,A8,A9,A10,A11,SCALE,U1,U2,J3,
     $ IOB,NEXTRA,NPARMS,B1,B2,B3,B4,B5,J4,J5,J6
      DATA DCON/57.295779/
      WRITE(6,10) TITLE
      IF(IOUT.EQ.1) WRITE(16,10) TITLE
   10 FORMAT(12H1T H X Y Z--,5X,16A5//)
      IF(NEXTRA.LT.6) GO TO 100
      THETA=B(4)*DCON
      PHI=B(5)*DCON
      OMEGA=B(6)*DCON
      WRITE(6,20) THETA,PHI,OMEGA
      IF(IOUT.EQ.1) WRITE(16,20) THETA,PHI,OMEGA
   20 FORMAT(//" ORIENTATION OF MEASUREMENT AXES WITH RESPECT TO WIRE",
     $ " AXES"/
     $ " SPHERICAL COORDS OF Z MSMT AXIS: THETA=",F10.6," DEGREES"/
     $ 36X,"PHI=",F10.6," DEGREES"/
     $ " ROTATION AROUND Z AXIS:          OMEGA=",F10.6," DEGREES"//)
  100 N=1
      IF(IOB.GT.3) N=3
      WRITE(6,110) (B(I),I=1,N)
      IF(IOUT.EQ.1) WRITE(16,110) (B(I),I=1,N)
  110 FORMAT(" PRIMARY FIELD FACTORS:", 3E16.8//)
      N=(NPARMS+1)/2
      WRITE(6,130)
      IF(IOUT.EQ.1) WRITE(16,130)
  130 FORMAT(28H FINAL UNSCALED PARAMETERS--/
     $ 32X,11HRESISTIVITY,2X,3HLYR,10X,5HDEPTH/)
      DO 150 I=1,N
           RHO=1./B(NEXTRA+I)
           WRITE(6,140) I,B(NEXTRA+I),RHO
```

```
      IF(IOUT.EQ.1) WRITE(16,140) I,B(NEXTRA+I),RHO
140   FORMAT(5X,I3,4X,2E16.8)
150   CONTINUE
  IF(NPARMS.EQ.1) GO TO 300
  M2=N+1
  D=0.0
  DO 170 I=M2,NPARMS
      D=D+B(NEXTRA+I)
      L=I-N
      WRITE(6,160) I,B(NEXTRA+I),L,D
      IF(IOUT.EQ.1) WRITE(16,160) I,B(NEXTRA+I),L,D
160   FORMAT(5X,I3,4X,E16.8,16X,I3,4X,E16.8)
170   CONTINUE
300 RETURN
  END

  SUBROUTINE ITHXYZ(DV,XM,B,PRNT,NPRNT,N,TITLE,IOUT)

 INITIALIZATION ROUTINE FOR PROGRAM MQLVTHXYZ

  CHARACTER*5 TITLE
  DIMENSION DV(1),XM(200,5),B(1),PRNT(5),TITLE(16),ALPH(3),
 $ PRED(2),DT(50),AV(50)
  COMPLEX ESUM
  COMMON/MODEL/RK(10),D(9),NLAYER
  COMMON/PARM/ISTEP,XX,YY,RR,S,BNYQ,MM,TOL
  COMMON/FIN/R1,R2,R,SL,SIG1,X,Y
  COMMON/FINERR/HTOL,CNCTOL,INTYPE,NRUN,NEV,MEV,ESUM,LW
  COMMON/DCSAV/ARHO(100),BRHO(100),DRHO(100),AB2(100),RHOA(100),NRHO
 $ ,ABMAX,ABMIN
  COMMON/CXYZ/FNYQ,SLL,METHOD,NF,NCOMP(3),TOX,TNX,TOY,TNY,TOZ,TNZ,
 1SCALE,BL,BU,ND,IOB,NEXTRA,NPARMS,GX,GY,GZ,GT,XO,IFLTX,IFLTY,IFLTZ
 INPUT PARAMETERS ARE:

  TOL      - RELATIVE TOLERANCE USED IN HANKEL
             TRANSFORM CALCULATION
  METHOD   - CALCULATE STEP RESPONSE BY
             TIME INTEGRAL OF IMPULSE REPONSE IF METHOD=1
             TRANSFORM OF F(W)/W IF METHOD=2
  NF       - CALCULATE TRANSIENT MODEL AND DERIVATIVES
             USING FREQUENCY FUNCTION EVALUATED AT NF
             POINTS PER DECADE FREQUENCY.  IF NF=0 OR
             NF>12 THE COSINE OR SINE ROUTINE WILL EVALUATE
             THE FREQQENCY FUNCTION DIRECTLY
  CNCTOL   - RELATIVE TOLERANCE USED IN FINITE WIRE
             INTEGRATION
  INTYPE   -
  NRUN     - DENSITY OF NODES FOR FREQUENCY FUNCTION
             SPLINE INTERPOLATION
  MEV      - MAXIMUM NUMBER OF FUNCTION EVALUATIONS
             BY INTEGRATIN ROUTINE
```

```
      EQUIVALENCE (M,MM),(L,ICASE),(BL,BLWR),(BU,BUPR)
      NAMELIST/INIT/TOL,METHOD,CNCTOL,INTYPE,NRUN,MEV
     1,NF,ISTEP,X,Y,CI,SL,MM,M,IFLTX,IFLTY,IFLTZ,IOB,L,ICASE,BLWR,BUPR
C
C   CALL DELETE_EXTERNAL_VARIABLES('SAVEC')
C
      WRITE(6,100) TITLE
  100 FORMAT(12H1T H X Y Z--,5X,16A5/)
      IF(IOUT.EQ.1) WRITE(16,100) TITLE
C--SET DEFAULTS
      M=1
      ND=N
      L=1
      BLWR=-1E0
      BUPR=-1E0
      CI=1.0
      SL=1.0
      TOL=1.E-4
      METHOD=2
      MEV=300
      NRUN=1
      INTYPE=4
      CNCTOL=1.E-5
      ISTEP=0
      FNYQ=1200
      BNYQ=1E29
C
C   FNYQ IS SET TO HAVE NO EFFECT AND IS NOT A PARAMETER.  FOR INFO
C   ON ITS ORIGINAL PURPOSE, SEE KAUAHIKAUA AND ANDERSON, 1977
C
      NF=0
      IFLTX=0
      IFLTY=0
      IFLTZ=0
      IOB=3
C
      READ(5,INIT)
C
      IF(IOB.GT.0.AND.IOB.LE.6.AND.(L.EQ.1.OR.L.EQ.2).AND.SL.GE.0.0.AND.
     $ CI.GT.0.0.AND.(ISTEP.EQ.1.OR.ISTEP.EQ.0).AND.X.GE.0.0.AND.Y.GE.
     $ 0.0.AND.FNYQ.GT.0.0.AND.M.GT.0.AND.M.LT.10.AND.TOL.GE.0.0.AND.
     $ (METHOD.EQ.1.OR.METHOD.EQ.2).AND.CNCTOL.GE.0.0.AND.INTYPE.GT.0.
     $ AND.INTYPE.LE.4.AND.NRUN.GT.0.AND.MEV.GT.0) GO TO 4
      WRITE(6,2)
    2 FORMAT(" SOME $INIT PARMS OUT OF RANGE...")
      STOP
    4 R=SQRT(X*X+Y*Y)
      SL=SL/2.
C
      IF(BLWR.LT.0E0) BLWR=.005
      IF(BUPR.LT.0E0) BUPR=1E4
C
```

```
C--CALCULATE ESTIMATE OF PRIMARY FIELD STRENGTH
C  FROM DATA
C
      R12=(X+SL)**2+Y*Y
      R22=(X-SL)**2+Y*Y
      R1=SQRT(R12)
      R2=SQRT(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
    1       NCOMP(I)=0
      TOX=0E0
      TNX=0E0
      TOY=0E0
      TNY=0E0
      TOZ=0E0
      TNZ=0E0
      NPRNT=2
      IF(IOB.GT.3) GO TO 5
      IS=IOB
      IM=IOB
      GO TO 6
    5 IS=1
      IM=4
      NPRNT=3
    6 DO 30 I=IS,IM
            KOUNT=0
            DO 16 J=1,N
                  IF(IOB.LE.3) GO TO 15
C
C  CHECK XM(J,2) VALUES FOR PROPER RANGE
C
                  IF(XM(J,2).GE.1..OR.XM(J,2).LE.4..OR.I.GT.IS) GO TO 14
                  WRITE(6,13) J
   13             FORMAT(" DATA MATRIX ENTRY (",I3,",2) OUT OF RANGE")
                  STOP
   14             IF(IFIX(XM(J,2)).NE.I) GO TO 16
   15             KOUNT=KOUNT+1
                  DT(KOUNT)=XM(J,1)
                  AV(KOUNT)=DV(J)
   16             CONTINUE
            IF(KOUNT.EQ.0) GO TO 30
            GO TO (17,18,19,20),I
   17       TOX=DT(1)
            TNX=DT(KOUNT)
            NCOMP(1)=KOUNT
            GO TO 30
```

```
 18       TOY=DT(1)
          TNY=DT(KOUNT)
          NCOMP(2)=KOUNT
          GO TO 30
 19       TOZ=DT(1)
          TNZ=DT(KOUNT)
          NCOMP(3)=KOUNT
          GO TO 30
 20       NRHO=KOUNT
          ABMIN=DT(1)
          ABMAX=DT(KOUNT)
 30       CONTINUE
      IF(IOB.LT.5) GO TO 40
      DO 35 I=1,3
 35       NCOMP(I)=1
      TMAX=AMAX1(TNX,TNY,TNZ)
      TNX=TMAX
      TNY=TMAX
      TNZ=TMAX
      TMIN=TMAX
      IF(TOX.LT.TMIN.AND.TOX.GT.0E0) TMIN=TOX
      IF(TOY.LT.TMIN.AND.TOY.GT.0E0) TMIN=TOY
      IF(TOZ.LT.TMIN.AND.TOZ.GT.0E0) TMIN=TOZ
      TOX=TMIN
      TOY=TMIN
      TOZ=TMIN
 40   SLL=2.*SL
C
C   SET UNUSED IFLTX, IFLTY, IFLTZ TO ZERO
C
 45   GO TO (46,47,48,47,49,49),IOB
 46   IFLTY=0
 47   IFLTZ=0
      IF(IOB.EQ.4.OR.IOB.EQ.1) GO TO 49
 48   IFLTX=0
      IF(IOB.EQ.2) GO TO 49
      IFLTY=0
 49   CONTINUE
      WRITE(6,55) ISTEP,X,Y,R,CI,SLL,L,IFLTX,IFLTY,IFLTZ
      IF(IOUT.EQ.1) WRITE(16,55) ISTEP,X,Y,R,CI,SLL,L,IFLTX,IFLTY,IFLTZ
 55   FORMAT(/19H DATA SET PARMS ARE/7H ISTEP=,I2,2X,2HX=,E12.4,
     12X,2HY=,E12.4,2X,2HR=,E12.4,2X,8HCURRENT=,E12.4/
     215H SOURCE LENGTH=,E12.4,2X,3H L=,I2/
     4 8H  IFLTX=,I2,8H  IFLTY=,I2,8H  IFLTZ=,I2)
      SCALE=CI
      IF(L.EQ.2) GO TO 56
      SL=0.0
      SCALE=SCALE*SLL
      GX=GX0*CI*SLL
      GY=GY0*CI*SLL
      GZ=GZ0*CI*SLL
C--SET EQUIVALENT VALUES FOR INITIALIZING COMMON AREAS
```

```
   56 XX=X
      YY=Y
      S=SIG1
      RR=R
      HTOL=TOL
      SLL=SL
      NLAYER=M
      NPARMS=2*M-1
      NEXTRA=1
      IF(IOB.EQ.4) NEXTRA=3
      IF(IOB.EQ.5) NEXTRA=6
C--GX,GY,GZ ARE PRIMARY FIELD VALUES PASSED TO MODEL ROUTINES
C  GT IS SCALING FACTOR FOR TIME AND FREQUENCY
      GT=2./(R*R*12.56637062E-7)
      WRITE(6,60) TOL,NF,IOB
      IF(IOUT.EQ.1) WRITE(16,60) TOL,NF,IOB
   60 FORMAT(/18H CONTROL PARMS ARE/5H TOL=,E12.4,2X,4HNF =,I4,
     1 2X,5HIOB =,I4)
      IF(L.EQ.1.OR.IOB.EQ.1) GO TO 66
      WRITE(6,65) CNCTOL,INTYPE,NRUN,MEV
      IF(IOUT.EQ.1) WRITE(16,65) CNCTOL,INTYPE,NRUN,MEV
   65 FORMAT(8H CNCTOL=,E12.4,2X,8HINTYPE =,I5,6X,6HNRUN =,I5,8X,
     15HMEV =,I5)
   66 IF(ISTEP.EQ.1) WRITE(6,70) METHOD
      IF(IOUT.EQ.1.AND.ISTEP.EQ.1) WRITE(16,70) METHOD
   70 FORMAT(8H METHOD=,I5//)
C
C  IF IOB.EQ.5, THE ORIENTATION OPTION HAS BEEN CHOSEN AND NEXTRA IS
C  SET TO 6 TO ACCOMODATE 3 SCALING FACTORS AND 3 ORIENTATION PARMS.
C  FINALLY IOB IS SET TO 4 TO CAUSE ALL 3 FIELD COMPONENTS TO
C  BE COMPUTED
C
      IF(IOB.EQ.5) IOB=4
C
C  CHECK B(I) FOR VALID VALUES
C
      K=NEXTRA+NPARMS
      DO 110 I=1,K
           IF(B(I).GT.0E0) GO TO 110
           IF(NEXTRA.EQ.6.AND.I.GE.3.AND.I.LE.6) GO TO 110
           WRITE(6,99) I
   99      FORMAT(" B(",I2,") .LE. 0E0")
      STOP
  110      CONTINUE
      RETURN
      END

      SUBROUTINE FCODE(DV,XM,BPARMS,PRNT,F,IF,IDER)
C
C  FCODE CALLED BY MQLVTHXYZ
C  FCODE CALCULATES HX, HY, AND HZ (STEP OR IMPULSE)
C  TRANSIENT SOUNDING MODELS FOR AN INFINITESIMAL AND
```

```
C   FINITE LENGTH WIRE SOURCE.
C
C   T,VX,VY,VZ    WORK ARRAYS INTERNAL TO FCODE
C   DV            VOLTAGE DATA ARRAY
C   XM(1,I)       TIMES ARRAY
C   XM(2,I)       WEIGHTS ARRAY IF IOB.LE.3
C                 INDICES ARRAY IF IOB.GT.3
C                 (1=HX, 2=HY, 3=HZ, 4=RHOA)
C   XM(3,I)       WEIGHTS ARRAY IF IOB.GT.3
C   ND            NUMBER OF DATA POINTS
C   P,S,PS        WORK ARRAYS REQUIRED BY SPLIN1
C
C   THE PARAMETER IOB CONTROLS THE OPERATION OF FCODE:
C
C      IOB=1 CALCULATE X-COMPONENT ONLY
C      IOB=2 CALCULATE Y-COMPONENT ONLY
C      IOB=3 CALCULATE Z-COMPONENT ONLY
C      IOB=4 CALCULATE MIXED COMPONENTS, INCLUDING SCHLUMBERGER RHOA
C      IOB=5 SAME AS IOB=4 WITH ORIENTATION PARAMETERS.
C
C   WRITTEN BY JIM KAUAHIKAUA, USGS, DENVER, COLORADO
C   LAST MODIFIED NOVEMBER, 1978
C
      DIMENSION BPARMS(1),P(50),S(50),DV(1),XM(200,5),PS(2)
      COMMON/MODEL/K,DD,NLAYER
      COMMON/THICK/D
      COMMON/PASS/CHX
      COMMON/PARM/ISTEP,XX,YY,R,SIG1,BNYQ,M,TOL
      COMMON/FIN/A1,A2,A3,A4,SIGG1,A6,A7
      COMMON/SPLXYZ/AX(100),BX(100),DX(100),AY(100),BY(100),DY(100),
     $ AZ(100),BZ(100),DZ(100),NB,B(100),SPX(100),SPY(100),SPZ(100),IW
      COMMON/CXYZ/FN,SL,MTD,NF,NCOMP(3),TOX,TNX,TOY,TNY,TOZ,TNZ,SCALE,
     $ BLWR,BUPR,ND,IOB,NEXTRA,NPARMS,GX,GY,GZ,GT,XO,IFLTX,IFLTY,IFLTZ
      COMMON/DCSAV/ARHO(100),BRHO(100),DRHO(100),AB2(100),RHOA(100),NRHO
     $ ,ABMAX,ABMIN
      COMMON/SAVEC/TX(50),VX(50),NTX,TY(50),VY(50),NTY,TZ(50),VZ(50),
     $ NTZ,CPX,CPY,CPZ,PARMS(20)
C
C  GX,GY, AND GZ ARE THE PREDICTED PRIMARY FIELDS (FROM ITHXYZ)
C  SL IS HALF SOURCE LENGTH IF FINITE WIRE MODELS ARE TO BE USED,
C     IS ZERO OTHERWISE.
C
      REAL K(10),D(9),DD(9),PRNT(5)
      EXTERNAL CSPLN
      COMPLEX COMPL,FINHX,FINHY,FINHZ,CHX,CHY,CHZ,FLT
      DATA PS/0.0,0.0/
C
      IF(IF.GT.1) GO TO 150
C
C  MODELS ARE CALCULATED ONLY IF IF=1, OTHERWISE THEY ARE RETRIEVED
C  FROM STORAGE IN COMMON BLOCK /SAVEC/
C
```

```
      ISKIP=1
C
C   CHECK BPARMS TO SEE IF SAME AS LAST PARMS
C
      DO 5 I=1,NPARMS
          IF(PARMS(I).NE.BPARMS(NEXTRA+I)) ISKIP=0
    5     PARMS(I)=BPARMS(NEXTRA+I)
      IF(IOB.EQ.1.OR.IOB.GT.3) CPX=BPARMS(1)
      IF(IOB.EQ.2) CPY=BPARMS(1)
      IF(IOB.GT.3) CPY=BPARMS(2)
      IF(IOB.EQ.3) CPZ=BPARMS(1)
      IF(IOB.GT.3) CPZ=BPARMS(3)
C
C   IF BPARMS SAME AS LAST PARMS, SKIP MODEL COMPUTATION
C
      IF(ISKIP.EQ.1) GO TO 150
      SIG1=PARMS(1)
      SIGG1=SIG1
      M1=M-1
      BMIN=1.0
      BMAX=1.0
      DO 10 I=1,M1
         K(I)=PARMS(I)/SIG1
C---MAXIMUM NORMALIZED FREQUENCY, BMAX, IS DETERMINED BY SMALLEST
C   CONDUCTIVITY CONTRAST IN MODEL OR BY THE NYQUIST FREQUENCY, FN.
         BMAX=AMIN1(BMAX,K(I))
C---MINIMUM NORMALIZED FREQUENCY, BMIN, IS DETERMINED BY LARGEST
C   CONDUCTIVITY CONTRAST IN MODEL
         BMIN=AMAX1(BMIN,K(I))
   10    D(I)=PARMS(M+I)
      K(M)=PARMS(M)/SIG1
      BMIN=AMAX1(BMIN,K(M))
      BMAX=AMIN1(BMAX,K(M))
C
      CONT=GT/SIG1
      XTMAX=CONT*TNX*1.15
      XTMIN=CONT*TOX
      YTMAX=CONT*TNY*1.15
      YTMIN=CONT*TOY
      ZTMAX=CONT*TNZ*1.15
      ZTMIN=CONT*TOZ
C
      VX(1)=1.0
      VY(1)=1.0
      VZ(1)=3.*CONT*GZ
C
C
      IF(NF.GT.12.OR.NF.LE.0) NF=12
C
C---MODEL CALCULATED ON A SPLINE INTERPOLATED FREQUENCY FUNCTION
C   COMPUTED AT NF FREQUENCIES PER DECADE
C   RANGE AUTOMATICALLY DETERMINED FROM THE MAXIMUM AND MINIMUM
```

```
C   CONDUCTIVITIES IN MODEL
C
C
      BMAX=BUPR/BMAX
      BMIN=BLWR/BMIN
C
   15 NB=AINT(NF*ALOG10(BMAX/BMIN))+2
      NB1=NB+1
      X0=ALOG10(BMAX)+0.25
      ISAV=ISTEP
      IF(MTD.EQ.1) ISTEP=0
      IF(NB.LE.100) GO TO 18
C
C   NB MAY NOT EXCEED THE SIZE OF THE ARRAYS IN COMMON BLOCK /SPLXYZ/
C
      WRITE(6,95) NB
   95 FORMAT(" FCODE:  NB.GT.100, NB=",I5)
      STOP
   18 DO 1 J=1,NB
          I=NB1-J
          X=X0-J/FLOAT(NF)
          B(I)=X
          X=10E0**X
          F=X*CONT/6.2831853
          IF(SL.EQ.0E0) CALL HXYZ(X,CHX,CHY,CHZ,IOB)
          GO TO (17,20,40,20,20),IOB
   17     IF(SL.GT.0E0) CHX=FINHX(X)
   20     IF(SL.GT.0E0.AND.IOB.GT.1) CHY=FINHY(X)
          IF(IFLTX.NE.0) CHX=CHX*FLT(IFLTX,F)
          SPX(I)=REAL(CHX)
          IF(IOB.EQ.1) GO TO 1
          IF(IFLTY.NE.0) CHY=CHY*FLT(IFLTY,F)
          SPY(I)=REAL(CHY)
          IF(IOB.LT.3) GO TO 1
   40     IF(SL.GT.0E0) CHZ=FINHZ(X)
          IF(IFLTZ.NE.0) CHZ=CHZ*FLT(IFLTZ,F)
          SPZ(I)=REAL(CHZ)
    1     CONTINUE
      ISTEP=ISAV
C
C   NOTE:  B() ARRAY IS ACTUALLY ALOG10(B)
C
      IF(NCOMP(1).NE.0) CALL SPLIN1(NB,0,B,SPX,AX,BX,DX,0,PS,P,S)
      IF(NCOMP(2).NE.0) CALL SPLIN1(NB,0,B,SPY,AY,BY,DY,0,PS,P,S)
      IF(NCOMP(3).NE.0) CALL SPLIN1(NB,0,B,SPZ,AZ,BZ,DZ,0,PS,P,S)
C
C   THE SWITCH IW WORKS WITH COMMON BLOCK /SPLXYZ/ IN ROUTINE CSPLN
C   TO INTERPOLATE EITHER THE X, Y, OR Z COMPONENT
C
      IW=1
      IF(NCOMP(1).NE.0) CALL TRNSI(CSPLN,XTMAX,XTMIN,NTX,TX,VX,MTD)
      IW=2
```

```
      IF(NCOMP(2).NE.0) CALL TRNSI(CSPLN,YTMAX,YTMIN,NTY,TY,VY,MTD)
      IW=3
      IF(NCOMP(3).NE.0) CALL TRNSI(CSPLN,ZTMAX,ZTMIN,NTZ,TZ,VZ,MTD)
C
      IF(IOB.LE.3.OR.NRHO.EQ.0) GO TO 3
      CALL SCHSDG(SIG1,ABMAX,ABMIN,AB2,RHOA,NRHO,TOL)
      DO 2 I=1,NRHO
    2      AB2(I)=ALOG(AB2(I))
      CALL SPLIN1(NRHO,0,AB2,RHOA,ARHO,BRHO,DRHO,0,PS,P,S)
C
    3 NT=MAXO(NTX,NTY,NTZ)
      IF(ISTEP.EQ.0.OR.MTD.EQ.1) GO TO 35
      DO 30 I=1,NT
          IF(NCOMP(1).NE.0) VX(I)=VX(I)*SCALE+GX
          IF(NCOMP(2).NE.0) VY(I)=VY(I)*SCALE+GY
   30     IF(NCOMP(3).NE.0) VZ(I)=VZ(I)*SCALE+GZ
   35 DO 50 I=1,NT
           IF(I.LE.NTX) TX(I)=ALOG(TX(I))
           IF(I.LE.NTY) TY(I)=ALOG(TY(I))
           IF(I.LE.NTZ) TZ(I)=ALOG(TZ(I))
   50      CONTINUE
      IF(NCOMP(1).NE.0) CALL SPLIN1(NTX,0,TX,VX,AX,BX,DX,0,PS,P,S)
      IF(NCOMP(2).NE.0) CALL SPLIN1(NTY,0,TY,VY,AY,BY,DY,0,PS,P,S)
      IF(NCOMP(3).NE.0) CALL SPLIN1(NTZ,0,TZ,VZ,AZ,BZ,DZ,0,PS,P,S)
C
  150 TAU=CONT*XM(IF,1)
      TAU=ALOG(TAU)
      INDEX=IOB
      IF(IOB.GT.3) INDEX=IFIX(XM(IF,2))
      IF(INDEX.EQ.4) GO TO 188
      IF(NEXTRA.EQ.6) GO TO 180
      IF(INDEX.EQ.1) F=CPX*SPLINT(TAU,AX,BX,DX,NTX,TX,VX)
      IF(INDEX.EQ.2) F=CPY*SPLINT(TAU,AY,BY,DY,NTY,TY,VY)
      IF(INDEX.EQ.3) F=CPZ*SPLINT(TAU,AZ,BZ,DZ,NTZ,TZ,VZ)
      GO TO 190
C
C  ORIENTATION PARAMETERS OPTION HAS BEEN CHOSEN
C
  180 FX=CPX*SPLINT(TAU,AX,BX,DX,NTX,TX,VX)
      FY=CPY*SPLINT(TAU,AY,BY,DY,NTY,TY,VY)
      FZ=CPZ*SPLINT(TAU,AZ,BZ,DZ,NTZ,TZ,VZ)
      CS1=COS(BPARMS(4))
      SN1=SIN(BPARMS(4))
      CS2=COS(BPARMS(5))
      SN2=SIN(BPARMS(5))
      CS3=COS(BPARMS(6))
      SN3=SIN(BPARMS(6))
      IF(INDEX.EQ.3) GO TO 185
      FXP=FX*CS1+SN1*(-FZ*CS2+FY*SN2)
      FYP=FY*CS1-SN1*(FZ*CS2+FX*SN2)
      IF(INDEX.EQ.1) F=FXP*CS3+FYP*SN3
      IF(INDEX.EQ.2) F=FYP*CS3-FXP*SN3
```

```
      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,NRHO,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

      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/PARM/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
```

```
      NEW=0
      YY=ZHANKS(1,B,F3,TOL,LW,1)
      CALL SAVER(1,1)
      XX=ZHANKS(0,B,F4,TOL,LW,0)
      HX=2.*(2.*YY/B-XX)*X*Y/(DEL2*R2)
      HY=(XR2-1.)*2.*XX/DEL2+2.*(1.-2.*XR2)*YY/(R*DEL)
      IF(IOB.LT.3) GO TO 10
    5 HZ=ZHANKS(1,B,F4,TOL,LW,NEW)*CMPLX(2.0*B2*C,0.0)
   10 B8=.70710678118654D0*B
C
C  CALCULATE THE MODIFIED BESSEL FUNCTIONS OF THE FIRST AND
C  SECOND KINDS FOR THE HX AND HY HALFSPACE RESPONSES.
C
      IF(IOB.NE.3) CALL IKS(B8,I1K1,IKDIF)
      ZZ=ZERO
      IF(IOB.LT.3) GO TO 12
      ZZ=ONE-(ONE+CB+CMPLX(0E0,2E0*B2/3E0))*CEXP(-CB)
      ZZ=ZZ*CMPLX(0E0,-3E0*C/B2)
C
C  SUBTRACT THE PRIMARY FIELD VALUES IF ISTEP = 1
C
   12 IF(ISTEP.EQ.0) GO TO 20
      IF(IOB.EQ.3) GO TO 15
      IKDIF=IKDIF-CMPLX(1.0,0.0)
      I1K1=I1K1-CMPLX(0.5,0.0)
      IF(IOB.LT.3) GO TO 20
   15 ZZ=ZZ-CMPLX(C,0.0)
C
   20 HX=HX+2.*X*Y*IKDIF/(R2*R2)
      HY=HY+2.*(Y*Y*IKDIF/R2-I1K1)/R2
      HZ=HZ+ZZ
      DENOM=12.56637062*B2**ISTEP
      HX=HX/DENOM
      HY=HY/DENOM
      HZ=HZ/DENOM
      RETURN
      END

      COMPLEX FUNCTION FLT(IFLAG,F)
C
C  USER-DEFINED FUNCTION
C     IFLAG IS A PARAMETER PASSED FROM MQLVTHXYZ VIA $INIT
C     PARMS IFLTX, IFLTY, OR IFLTZ
C     F IS THE FREQUENCY IN HERTZ
C
      COMPLEX ONE
      DATA ONE/(1E0,0E0)/
      IF(IFLAG.EQ.0) RETURN
C
C  EXAMPLE:  SINGLE-POLE LOW PASS FILTER WHERE
C  IFLAG IS USED AS THE CORNER FREQUENCY
C
```

```
      CF=IFLAG*6.283185
      OMEGA=F*6.283185
      FLT=ONE/CMPLX(-CF,OMEGA)
      RETURN
      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
      DIMENSION T(100),V(100)
      EXTERNAL FUNC
      COMPLEX FUNC,ZLAGFO,ZLAGF1,CON
      COMMON/PARM/ISTEP,XX,YY,R,SIG1,BNYQ,M,TOL
      NT=AINT(0.5+5.*ALOG(TMAX/TMIN))+1
      NT1=NT+1
      X0=ALOG(TMAX)+0.2
      C=4.0E-7*3.1415927
      XI=C*SIG1*R*R/2.
      C=2./(XI*3.1415927)
      V0=V(1)
      NEW=1
      ISAV=ISTEP
      IF(MTD.EQ.1) ISTEP=0
      DO 2 J=1,NT
      I=NT1-J
      X=X0-0.2*J
      V(I)=0.0
      T(I)=EXP(X)
      IF(ISTEP.EQ.1) GO TO 3
      CON=C*ZLAGFO(X,FUNC,TOL,LW,NEW)/T(I)
      GO TO 4
    3 CON=C*XI*ZLAGF1(X,FUNC,TOL,LW,NEW)/T(I)
    4 V(I)=REAL(CON)
    2 NEW=0
      ISTEP=ISAV
      IF(MTD.EQ.2.OR.ISTEP.EQ.0) GO TO 10
      CALL INTEG1(NT,T,V,V0)
      DO 9 I=1,NT
    9 V(I)=V(I)*XI
   10 RETURN
      END
```

```
      SUBROUTINE SCHSDG(SIG1,XMAX,XMIN,AB2,RHOA,NRHO,TOL)
C
C  COMPUTE SCHLUMBERGER APPARENT RESISTIVITY SOUNDING CURVE USING
C  LAGGED CONVOLUTION
C
      DIMENSION AB2(1),RHOA(1)
      COMPLEX ZLAGH1,CMP
      COMMON/MODEL/RK(10),DNORM(9),M
      COMMON/THICK/D(9)
C
C  MUST TRANSFER MODEL FROM EM COMMON BLOCK, MODEL,
C  TO DC COMMON BLOCK DCMDL.
C
      COMMON/DCMDL/SIG(10),DDC(9),NLYR,INDEX
      EXTERNAL DCKERN
C
      NRHO=AINT(5E0*ALOG(XMAX/XMIN))+2
      NRHO1=NRHO+1
      X0=ALOG(XMAX)+0.25
      NLYR=M
      RHO1=1E0/SIG1
      DO 10 I=1,M
           DDC(I)=D(I)
   10      SIG(I)=SIG1*RK(I)
C
      INDEX=0
      NEW=1
      CMP=CMPLX(0E0,0E0)
      DO 20 J=1,NRHO
           I=NRHO1-J
           XX=X0-0.2*J
           AB2(I)=EXP(XX)
           A2=EXP(2E0*XX)
           IF(M.EQ.1) GO TO 19
           CMP=ZLAGH1(XX,DCKERN,TOL,LW,NEW)/AB2(I)
   19      RHOA(I)=RHO1+A2*REAL(CMP)
   20      NEW=0
      RETURN
      END

      REAL FUNCTION SPLINT(T,A,B,C,N,X,Y)
C---CUBIC SPLINE INTERPOLATION ROUTINE
C  ASSUMES PRIOR CALL TO SPLIN1
      DIMENSION A(1),B(1),C(1),X(1),Y(1)
      IF(T.LT.X(1).OR.T.GT.X(N)) GO TO 9
      N1=N-1
      DO 1 I=1,N1
      J=I
      IF(T.LT.X(I+1)) GO TO 2
    1 CONTINUE
    9 SPLINT=0.0
      IF(T.LT.X(1)) SPLINT=Y(1)
```

```
      RETURN
    2 Z=T-X(J)
      SPLINT=Y(J)+((C(J)*Z+B(J))*Z+A(J))*Z
      RETURN
      END

      COMPLEX FUNCTION DCKERN(AMBDA)
C
C  RESISTIVITY KERNEL FUNCTION CALLED BY SCHSDG
C
C     FOR INDEX = 0 RETURNS (KERNEL,PDIF W/R RHO(M))
C        WHERE PDIF = PARTIAL DERIVATIVE
C     FOR INDEX = I RETURNS (PDIF KERNEL W/R D(I),
C                           PDIF KERNEL W/R RHO(I))
C     M > I > 0
C
C  ALGORITHM FOR PARTIALS FROM:
C  JOHANSEN, H.K., 1975, AN INTERACTIVE COMPUTER GRAPHICS-
C     DISPLAY-TERMINAL SYSTEM FOR INTERPRETATION OF RESISTIVITY SOUNDING
C     GEOPHYSICAL PROSPECTING, 23: 449-458.
C
C  PROGRAMMED BY JKAUAHIKAUA, USGS, DENVER, COLO, 1977.
C
      COMMON/DCMDL/SIG(10),D(9),M,INDEX
      REAL LM,LN,L2
      PSIG=0E0
      PD=0E0
      L=M
      LM=1E0
   10 L1=L-1
      E=EXP_(-2E0*AMBDA*D(L1))
      EN=(1E0-E)/(1E0+E)
      DENOM=SIG(L)+SIG(L1)*LM*EN
      IF(L.EQ.2) L2=LM
      IF(L.EQ.2) E1=EN
      LN=(SIG(L1)*LM+SIG(L)*EN)/DENOM
      IF(INDEX+1.LT.M.AND.L-2.EQ.INDEX) PSIG=(1E0-EN*LN)*LM/DENOM
      IF(INDEX-L1) 15,13,14
   13 PSIG=((EN-LN)+(1E0-EN*LN)*SIG(L1)*PSIG)/DENOM
      PD=(1E0+EN)*(SIG(L)-LN*LM*SIG(L1))*2E0*AMBDA*E/(DENOM*(1E0+E))
      GO TO 15
   14 X=(1E0-LN*EN)*SIG(L1)/DENOM
      PSIG=PSIG*X
      PD=PD*X
   15 L=L-1
      LM=LN
      IF(L.GT.1) GO TO 10
      IF(INDEX.GT.0) GO TO 20
      PL1=L2*(1E0-E1*LM)*AMBDA/DENOM
      LM=AMBDA*(LM-1E0)
      DCKERN=CMPLX(LM,PL1)
      RETURN
```

```
   20 DCKERN=CMPLX(AMBDA*PD,AMBDA*PSIG)
      RETURN
      END

      COMPLEX FUNCTION CSPLN(RT)
C---CUBIC SPLINE INTERPOLATION ROUTINE
C   RETURNS A COMPLEX RESULT (IMAG PART = 0)
C   CALLED WITH TRNSI BY FTHXYZ
C   ASSUMES PREVIOUS CALL TO SPLIN1
C
C   IW USED AS FLAG (SET BY CALLING ROUTINE):
C      IW=1 - USE AX, BX, AND CX TO INTERPOLATE SPX
C      IW=2 - USE AY, BY, AND CY TO INTERPOLATE SPY
C      IW=3 - USE AZ, BZ, AND CZ TO INTERPOLATE SPZ
C
      COMMON/SPLXYZ/AX(100),BX(100),CX(100),AY(100),BY(100),CY(100),
     1 AZ(100),BZ(100),CZ(100),NB,B(100),SPX(100),SPY(100),SPZ(100),IW
C
C   NOTE THAT THE FUNCTION BEING SPLINED IS A FUNCTION
C   OF B() = ALOG(B)
C
      T=ALOG10(RT)
      IF(T.LT.B(1).OR.T.GT.B(NB)) GO TO 20
      N1=NB-1
      DO 10 I=1,N1
           J=I
           IF(T.LT.B(I+1)) GO TO 30
   10      CONTINUE
   20 CS=0.0
      IF(T.GE.B(1)) GO TO 25
      IF(IW.EQ.1) CS=SPX(1)
      IF(IW.EQ.2) CS=SPY(1)
      IF(IW.EQ.3) CS=SPZ(1)
   25 CSPLN=CMPLX(CS,0.0)
      RETURN
   30 Z=T-B(J)
      IF(IW.EQ.1) CS=SPX(J)+((CX(J)*Z+BX(J))*Z+AX(J))*Z
      IF(IW.EQ.2) CS=SPY(J)+((CY(J)*Z+BY(J))*Z+AY(J))*Z
      IF(IW.EQ.3) CS=SPZ(J)+((CZ(J)*Z+BZ(J))*Z+AZ(J))*Z
      CSPLN=CMPLX(CS,0.0)
      RETURN
      END

      COMPLEX FUNCTION FINHX(B)
C--HX FIELD OF FINITE ELECTRIC WIRE
      COMMON/PARM/ISTEP,A1,A2,A3,A4,BNYQ,A5,A6
      COMMON/FIN/R1,R2,R,L,SIG1,X,Y
      COMMON/THICK/D(9)
      COMMON/MODEL/K(10),DD(9),M
      REAL L,K
      DOUBLE PRECISION B8
      COMPLEX F31,F32,I1K1
```

```
      SB=SQRT(B)
      FINHX=CMPLX(0.0,0.0)
      IF(SB.GT.BNYQ) RETURN
      DEL=R/SB
      IF(M.EQ.1) GO TO 12
      B1=R1/DEL
      B2=R2/DEL
      M1=M-1
      DO 1 I=1,M1
    1 DD(I)=2.*D(I)/DEL
      CALL FINF3(B1,B2,F31,F32)
      FINHX=CMPLX(1./DEL,0.0)*(F31/R1-F32/R2)
   12 B8=0.7071067811865475D0/DEL
      FINHX=FINHX+(I1K1(B8*R1)/R1**2-I1K1(B8*R2)/R2**2)
      FINHX=-FINHX*CMPLX(Y/6.2838153,0.0)
      RETURN
      END

      COMPLEX FUNCTION FINHY(B)
C--HY FIELD OF FINITE ELECTRIC WIRE
C  PASSES HX IN COMMON BLOCK 'PASS'
      EXTERNAL ZHY
      COMMON/FIN/R1,R2,R,L,SIG1,X,Y
      COMMON/CONST/DEL,DEL2,Z
      COMMON/PASS/FINHX
      COMMON/PARM/ISTEP,A1,A2,A3,A4,BNYQ,M,A6
      COMPLEX FINITE,F31,F32,FINHX,Z,I1K1,IK1,IK2
      DOUBLE PRECISION B8
      REAL L
      SB=SQRT(B)
      FINHX=CMPLX(0.0,0.0)
      FINHY=CMPLX(0.0,0.0)
      IF(SB.GT.BNYQ) RETURN
      DEL=R/SB
      DEL2=DEL*DEL
      FINHY=FINITE(ZHY,B)
      IF(M.EQ.1) GO TO 10
      B1=R1/DEL
      B2=R2/DEL
      CALL FINF3(B1,B2,F31,F32)
      FINHY=FINHY-((X+L)*F31/R1-(X-L)*F32/R2)*CMPLX(1./DEL,0.0)
   10 B8=0.7071067811865475D0/DEL
      R12=R1*R1
      R22=R2*R2
      IK1=I1K1(B8*R1)/R12
      IK2=I1K1(B8*R2)/R22
C         REMOVE PRIMARY FIELD IF ISTEP=1
      IF(ISTEP.EQ.1) IK1=IK1-CMPLX(0.5/R12,0.0)
      IF(ISTEP.EQ.1) IK2=IK2-CMPLX(0.5/R22,0.0)
      FINHY=-(FINHY-((X+L)*IK1-(X-L)*IK2))*CMPLX(1./6.2831853,0.0)
      IF(X*Y.EQ.0.0) GO TO 20
      IF(M.EQ.1) GO TO 15
```

```
      FINHX=(F31/R1-F32/R2)*CMPLX(1./DEL,0.0)
   15 FINHX=-(FINHX+(IK1-IK2))*CMPLX(Y/6.2831853,0.0)
   20 IF(ISTEP.EQ.0) GO TO 30
      FINHY=FINHY/B
      FINHX=FINHX/B
   30 RETURN
      END

      COMPLEX FUNCTION FINHZ(B)
C--HZ FIELD OF FINITE ELECTRIC WIRE
      COMPLEX FINITE
      EXTERNAL ZHZ
      COMMON/PARM/ISTEP,A1,A2,A3,A4,BNYQ,M,A6
      COMMON/FIN/R1,R2,R,RL,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.EQ.1) FINHZ=FINHZ/B
      RETURN
      END

      COMPLEX FUNCTION F3(G)
      COMPLEX V1,F1,C,ONE
      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)
      COMPLEX V1,F1,C,ONE
      DATA ONE/(1.0,0.0)/
      CALL RECURS(G,V1,F1)
      C=G
      F4=(V1*C*C*(ONE-F1))/((C+V1*F1)*(C+V1))
      RETURN
      END

      SUBROUTINE RECURS(G,V1,F1)
C
C           RECURS ALGORITHM FROM ANDERSON(1974)
C
      COMPLEX C,VM,V1,F1,EVD,ONE,T
      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.EQ.1) GO TO 30
```

```
      J=M-1
   10 V1=CSQRT(CMPLX(G2,2.0*K(J)))
      EVD=-V1*D(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 FINF3(B1,B2,F31,F32)
      COMMON/FINERR/TOL,T,IT,N,NEV,MEV,ES,LW
      COMPLEX F31,F32,ZHANKS,ES
      EXTERNAL F3
      F31=ZHANKS(1,B1,F3,TOL,LW,1)
      IF(B1.EQ.B2) GO TO 10
      F32=ZHANKS(1,B2,F3,TOL,LW,1)
      RETURN
   10 F32=F31
      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 'QPOINT').
C
C   PARAMETERS:
C
C   FUNC =  EXTERNAL DECLARED COMPLEX FUNCTION DEFINING THE DIPOLE FIELD
C           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
C---COMMON PARMAMETERS (INPUT) REQUIRED:
C
C   HAKTOL = REQUESTED HANKEL TRANSFORM (ZLAG) TOLERANCE.
C            USE HAKTOL.LE.1.E-6*EPS, EPS=ACTUAL HANKEL REL. ERROR.
C   FINTOL = REQUESTED FINITE INTEGRAL (CANC4) TOLERANCE.
C            USE FINTOL.LE.1.E-3*EP, EP=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.
C            NOTE: NFIN.GT.1 TAKES 'NFIN TIMES' AS LONG TO RUN, BUT
```

```
C            WILL GIVE ADDITIONAL ACCURACY, IF NEEDED.
C  MEV     = MAX. FUNCT EVAL"S FOR CANC4 (NORMALLY MEV>300)
C  R1      = MAX SPACING FROM WIRE END TO RECEIVER POINT (XX,YY)
C          = SQRT((XX+L)**2+YY*YY)
C  R2      = MIN SPACING FROM WIRE END TO RECEIVER POINT (XX,YY)
C          = SQRT((XX-L)**2+YY*YY)
C  RO      = SPACING FROM WIRE CENTER TO RECEIVER POINT (XX,YY)
C          = SQRT(XX*XX+YY*YY)
C  D(9)    = THICKNESS OF M-LAYERS IN MODEL. (METERS)
C  K(10)   = CONDUCTIVITY RATIO SIG(I)/SIG(1)
C          FOR I=1,M
C  M       = NO. LAYERS IN MODEL (M.GE.1.AND.M.LT.10)
C
      COMMON/FINERR/HAKTOL,FINTOL,INTYPE,NFIN,NEV,MEV,ESUM,LW
      COMMON/SPLN80/FDR(80),AR(80),BR(80),CR(80),DR(80),ER(80),
     & FDI(80),AI(80),BI(80),CI(80),DI(80),EI(80),RLM1,DELRLM,NB
      COMMON/FIN/R1,R2,RO,L,SIG1,X,Y
      COMMON/THICK/D(9)
      COMMON/MODEL/K(10),DD(9),M
      COMMON/CONST/DEL,DEL2,Z2DEL3
      REAL L,K
      COMPLEX FUNC,ESUM,FD,FUNINT,CANC4,Z2DEL3
      EXTERNAL FUNINT
C   ISIZE IS THE MAXIMUM POSSIBLE NUMBER OF NODES IN QUINTIC SPLINE.
      DATA ISIZE/80/
      DEL=RO/SQRT(BFIN)
      DEL2=DEL*DEL
      Z2DEL3=CMPLX(0.0,2./(DEL2*DEL))
      M1=M-1
      DO 1 I=1,M1
    1 DD(I)=2.*D(I)/DEL
      BMAX=R1/DEL
      BMIN=R2/DEL
      IF(X.LE.L) BMIN=Y/DEL
      NB=AINT(5.*ALOG(BMAX/BMIN))+2
      NB=MAX(NB,3)
      XO=ALOG(BMIN)+NB*0.2
      NB=NB+3
      NRMAX=ISIZE/NB
      IF(NFIN.LE.NRMAX) GO TO 3
      IF(NRMAX.GT.0.0) GO TO 2
      PRINT, 'ERROR IN FINITE: INSUFFICIENT SPLINE NODES'
      STOP
    2 NFIN=NRMAX
      PRINT, 'ERROR IN FINITE: NFIN TOO LARGE, RESET TO ',NFIN
3     DELRLM=.2/FLOAT(NFIN)
      XO=XO-DELRLM
      DO 5 ITIME=1,NFIN
      NEW=1
      XO=XO+DELRLM
      DO 5 J=1,NB
      I=(NB+1)-J
```

```
      I=NFIN*(I-1)+ITIME
      XX=X0-0.2*J
      BM=EXP(XX)
      RM=BM*DEL
      IF(I.EQ.1) RLM1=ALOG(RM)
      FD=FUNC(BM,NEW,RM)
      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)
   10 RETURN
      END

      COMPLEX FUNCTION ZHY(B,NEW,R)
      COMPLEX ZLAGHO,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=ZLAGHO(ALOG(B),F4,TOL,LW,NEW)/B
    2 B8=0.70710678118654D0*B
      CALL IKS(B8,I1K1,IKDIF)
      IF(ISTEP.EQ.0) GO TO 5
C        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.))*CEXP(CMPLX(-B,-B))
      ZHZ=ZHZ-HAF*CMPLX(0.0,3./(R**3*B*B))
```

```
C          REMOVE THE PRIMARY FIELD IF ISTEP=1
      IF(ISTEP.EQ.1) ZHZ=ZHZ-CMPLX(1./R**3,0.0)
      RETURN
      END

      COMPLEX FUNCTION FUNINT(X)
C--COMPLEX FUNCTION INTERPOLATION BY QUINTIC SPLINE VIA
C   CALL TO 'QPOINT', WHERE THE QUINTIC SPLINE
C   COEFFICIENTS AR,BR,CR,DR,ER, AI,BI,CI,DI,EI WERE
C   PREVIOUSLY OBTAINED BY SUBR 'QUINT'.
C
      DIMENSION SR(80),AR(80),BR(80),CR(80),DR(80),ER(80),
     & SI(80),AI(80),BI(80),CI(80),DI(80),EI(80)
      COMMON/SPLN80/SR,AR,BR,CR,DR,ER,SI,AI,BI,CI,DI,EI,RLM1,DELRLM,NL
      COMMON/FIN/R1,R2,R0,XL,SIG1,XX,Y
      R=ALOG(SQRT(X*X+Y*Y))
      CALL QPOINT(NL,SR,AR,BR,CR,DR,ER,RLM1,DELRLM,R,YR)
      CALL QPOINT(NL,SI,AI,BI,CI,DI,EI,RLM1,DELRLM,R,YI)
      FUNINT=CMPLX(YR,YI)
      RETURN
      END

      COMPLEX FUNCTION I1K1(B8)
C--COMPUTE MODIFIED BESSEL FUNCTION PRODUCT I1*K1 FOR
C   PARAMETERS
C   B8 =    DOUBLE PRECISION ARGUMENT (=B/DSQRT(2.0D0) HERE)
C   I1K1 = I1*K1 COMPLEX RESULT
C
C--SUBR KELVIN CALLED AND D.P. USED BEFORE CMPLX"ING
C
      COMPLEX Z,Z2,TERM1
      DOUBLE PRECISION B8,BB(8),Q1,Q2
      IF(B8.GT.20.0D0) GO TO 1
      CALL KELVIN(B8,8,BB)
      Q1=-BB(6)*BB(8)+BB(5)*BB(7)
      Q2= BB(5)*BB(8)+BB(6)*BB(7)
      I1K1=CMPLX(SNGL(Q1),SNGL(Q2))
      RETURN
    1 B=0.70710678*SNGL(B8)
      Z=CMPLX(B,B)
      K=-1
      Z2=4.*Z*Z
      TERM1=CMPLX(1.0,0.0)
      I1K1=TERM1
    2 K=K+2
      COM=-K/(Z2*(K+1))
      TERM1=TERM1*COM*(4.-K*K)
      I1K1=I1K1+TERM1
      IF(CABS(TERM1)/CABS(I1K1).GT.1.E-6) GO TO 2
      I1K1=I1K1/(2.*Z)
      RETURN
      END
```

```
      SUBROUTINE SPLIN1(M,H,X,Y,A,B,C,IT,D,P,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= INDEP.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            P,S= WORK ARRAYS (EACH DIM=M).
C---ERROR RETURN WITH M=-(ABS(M)) IF ANY PARM OUT OF RANGE.
C  THE RESULTING CUBIC SPLINE IS OF THE FORM:
C     Y=Y(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),P(1),S(1),MUL
      IF(IT.LT.0.OR.IT.GT.1.OR.H.LT.0..OR.M.LT.3) GO TO 999
      N=M-1
      IF(IT.EQ.0) GO TO 20
C---1ST DERIVATIVE BOUNDARIES GIVEN
      NE=N-1
      IF(H) 999,11,1
C---EQUAL SPACING H .GT. 0. AND IT=1
    1 HH=3.0/H
      DO 2 I=1,NE
      B(I)=4.0
      C(I)=1.0
      A(I)=1.0
    2 P(I)=HH*(Y(I+2)-Y(I))
      P(1)=P(1)-D(1)
      P(NE)=P(NE)-D(2)
C---SOLUTION OF TRIDIAGONAL MATRIX EQ. OF ORDER NE
    3 C(1)=C(1)/B(1)
      P(1)=P(1)/B(1)
      DO 4 I=2,NE
      MUL=1.0/(B(I)-A(I)*C(I-1))
      C(I)=MUL*C(I)
    4 P(I)=MUL*(P(I)-A(I)*P(I-1))
C---OBTAIN SPLINE COEFFICIENTS
      A(NE+IT)=P(NE)
      I=NE-1
    5 A(I+IT)=P(I)-C(I)*A(I+IT+1)
```

```
      I=I-1
       IF(I.GE.1) GO TO 5
      IF(IT.EQ.0) GO TO 6
      A(1)=D(1)
      A(M)=D(2)
    6 IF(H.EQ.0.) GO TO 14
      HH=1.0/H
      DO 7 I=1,N
      MUL=HH*(Y(I+1)-Y(I))
      B(I)=HH*(3.0*MUL-(A(I+1)+2.0*A(I)))
    7 C(I)=HH*HH*(-2.0*MUL+A(I+1)+A(I))
      RETURN
C--UNEQUAL SPACING H=0.. AND IT=1
   11 DO 12 I=1,N
   12 S(I+1)=X(I+1)-X(I)
      DO 13 I=1,NE
      B(I)=2.0*(S(I+1)+S(I+2))
      C(I)=S(I+1)
      A(I)=S(I+2)
   13 P(I)=3.0*(S(I+1)**2*(Y(I+2)-Y(I+1))+S(I+2)**2*(Y(I+1)-Y(I)))/
     $ (S(I+1)*S(I+2))
      P(1)=P(1)-S(3)*D(1)
      P(NE)=P(NE)-S(N)*D(2)
      GO TO 3
   14 DO 15 I=1,N
      HH=1.0/S(I+1)
      MUL=(Y(I+1)-Y(I))*HH**2
      B(I)=3.0*MUL-(A(I+1)+2.0*A(I))*HH
   15 C(I)=-2.0*MUL*HH+(A(I+1)+A(I))*HH**2
      RETURN
C--2ND DERIVATIVE BOUNDARIES GIVEN
   20 NE=N+1
      IF(H) 999,31,21
C--EQUAL SPACING H .GT. 0 AND IT=0
   21 HH=3.0/H
      DO 22 I=2,N
      B(I)=4.0
      C(I)=1.0
      A(I)=1.0
   22 P(I)=HH*(Y(I+1)-Y(I-1))
      B(1)=2.0
      B(NE)=2.0
      C(1)=1.0
      C(NE)=1.0
      A(NE)=1.0
      P(1)=HH*(Y(2)-Y(1))-0.5*H*D(1)
      P(NE)=HH*(Y(M)-Y(N))+0.5*H*D(2)
      GO TO 3
C--UNEQUAL SPACING H=0 AND IT=0
   31 DO 32 I=1,N
   32 S(I+1)=X(I+1)-X(I)
      N1=N-1
```

```
      DO 33 I=1,N1
      B(I+1)=2.0*(S(I+1)+S(I+2))
      C(I+1)=S(I+1)
      A(I+1)=S(I+2)
   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)))/
     *      (S(I+1)*S(I+2))
      B(1)=2.0
      B(NE)=2.0
      C(1)=1.0
      C(NE)=1.0
      A(NE)=1.0
      P(1)=3.0*(Y(2)-Y(1))/S(2)-0.5*S(2)*D(1)
      P(NE)=3.0*(Y(M)-Y(N))/S(M)+0.5*S(M)*D(2)
      GO TO 3
  999 M=-IABS(M)
      RETURN
      END

         COMPLEX FUNCTION ZLAGHO(X,FUN,TOL,L,NEW)
C--*** A SPECIAL LAGGED* CONVOLUTION METHOD TO COMPUTE THE
C  INTEGRAL FROM 0 TO INFINITY OF 'FUN(G)*J0(G*B)*DG' DEFINED AS THE
C  COMPLEX HANKEL TRANSFORM OF ORDER 0 AND ARGUMENT X(=ALOG(B))
C  BY CONVOLUTION FILTERING WITH COMPLEX FUNCTION 'FUN'--AND
C  USING A VARIABLE CUT-OFF METHOD WITH EXTENDED FILTER TAILS....
C
C--REF: ANDERSON, W.L., 1975, NTIS REPT. PB-242-800.
C
C--PARAMETERS:
C
C      * X     = REAL ARGUMENT(=ALOG(B) AT CALL) OF THE HANKEL TRANSFORM
C                'ZLAGHO' IS USEFUL ONLY WHEN X=(LAST X)-.20 *** I.E.,
C                SPACED SAME AS FILTER USED--IF THIS IS NOT CONVENIENT,
C                THEN SUBPROGRAM 'ZHANKO' IS ADVISED FOR GENERAL USE.
C                (ALSO SEE PARM 'NEW' & NOTES (2)-(4) BELOW).
C      FUN(G)= EXTERNAL DECLARED COMPLEX FUNCTION NAME (USER SUPPLIED)
C                OF A REAL ARGUMENT G.
C              NOTE: IF PARMS OTHER THAN G ARE REQUIRED, USE COMMON IN
C                  CALLING PROGRAM AND IN SUBPROGRAM FUN.
C                  THE COMPLEX FUNCTION FUN SHOULD BE A MONOTONE
C                  DECREASING FUNCTION AS THE ARGUMENT G BECOMES LARGE...
C                FOR REAL-ONLY FUNCTIONS, SUBPROGRAM 'RLAGHO' IS ADVISED;
C                HOWEVER, TWO REAL-FUNCTIONS F1(G),F2(G) MAY BE
C                INTEGRATED IN PARALLEL BY WRITING FUN=CMPLX(F1(G),F2(G))
C        TOL=      REAL TOLERANCE EXCEPTED AT CONVOLVED TAILS--I.E.,
C                  IF FILTER*FUN<TOL*MAX, THEN REST OF TAIL IS TRUNCATED.
C                  THIS IS DONE AT BOTH ENDS OF FILTER.  TYPICALLY,
C                  TOL <= .0001 IS USUALLY OK--BUT THIS DEPENDS ON
C                  THE FUNCTION FUN AND PARAMETER X...IN GENERAL,
C                A 'SMALLER TOL' WILL USUALLY RESULT IN 'MORE ACCURACY'
C                BUT WITH 'MORE WEIGHTS' BEING USED.  TOL IS NOT DIRECTLY
C                RELATED TO TRUNCATION ERROR, BUT GENERALLY SERVES AS AN
C                APPROXIMATION INDICATOR... FOR VERY LARGE OR SMALL B,
```

```
C              ONE SHOULD USE A SMALLER TOL THAN RECOMMENDED ABOVE...
C       L=       RESULTING NO. FILTER WTS. USED IN THE VARIABLE
C                CONVOLUTION (L DEPENDS ON TOL AND FUN).
C              MIN.L=20 AND MAX.L=193---WHICH COULD
C                OCCUR IF TOL IS VERY SMALL AND/OR FUN NOT DECREASING
C                VERY FAST...
C     * NEW=   1 IS NECESSARY 1ST TIME OR BRAND NEW X.
C              0 FOR ALL SUBSEQUENT CALLS WHERE X=(LAST X)-0.20
C                IS ASSUMED INTERNALLY BY THIS ROUTINE.
C                NOTE:  IF THIS IS NOT TRUE, ROUTINE WILL
C                STILL ASSUME X=(LAST X)-0.20 ANYWAY...
C                IT IS THE USERS RESPONSIBILITY TO NORMALIZE
C                BY CORRECT B=EXP(X) OUTSIDE OF CALL (SEE USAGE BELOW).
C              THE LAGGED CONVOLUTION METHOD PICKS UP SIGNIFICANT
C                TIME IMPROVEMENTS WHEN THE KERNEL IS NOT A
C                SIMPLE ELEMENTARY FUNCTION...DUE TO INTERNALLY SAVING
C                ALL KERNEL FUNCTION EVALUATIONS WHEN NEW=1...
C                THEN WHEN NEW=0, ALL PREVIOUSLY CALCULATED
C                KERNELS WILL BE USED IN THE LAGGED CONVOLUTION
C                WHERE POSSIBLE, ONLY ADDING NEW KERNEL EVALUATIONS
C                WHEN NEEDED (DEPENDS ON PARMS TOL AND FUN)
C
C---THE RESULTING COMPLEX CONVOLUTION SUM IS GIVEN IN ZLAGHO; THE HANKEL
C  TRANSFORM IS THEN ZLAGHO/B WHICH IS TO BE COMPUTED AFTER EXIT FROM
C  THIS ROUTINE.... WHERE B=EXP(X), X=ARGUMENT USED IN CALL...
C
C---USAGE--- 'ZLAGHO' IS CALLED AS FOLLOWS:
C     ...
C     COMPLEX Z,ZLAGHO,ZF
C     EXTERNAL ZF
C     ...
C     Z=ZLAGHO(ALOG(B),ZF,TOL,L,NEW)/B
C     ...
C     END
C     COMPLEX FUNCTION ZF(G)
C     ...USER SUPPLIED CODE...
C     END
C
C---NOTES:
C       (1).  EXP-UNDERFLOW'S MAY OCCUR IN EXECUTING THE SUBPROGRAM
C     BELOW; HOWEVER, THIS IS OK PROVIDED THE MACHINE SYSTEM SETS
C     ANY & ALL EXP-UNDERFLOW'S TO 0.0....
C     (2).  AS AN AID TO UNDERSTANDING & USING THE LAGGED CONVOLUTION
C     METHOD, LET BMAX>=BMIN>0 BE GIVEN. THEN IT CAN BE SHOWN
C     THAT THE ACTUAL NUMBER OF B'S IS NB=AINT(5.*ALOG(BMAX/BMIN))+1,
C     PROVIDED BMAX/BMIN>=1. THE USER MAY THEN ASSUME AN 'ADJUSTED'
C     BMINA=BMAX*EXP(-.2*(NB-1)).  THE METHOD GENERATES THE DECREASING
C     ARGUMENTS SPACED AS X=ALOG(BMAX),X-.2,X-.2*2,...,ALOG(BMINA).
C     FOR EXAMPLE, ONE MAY CONTROL THIS WITH THE CODE:
C              ...
C              NB=AINT(5.*ALOG(BMAX/BMIN))+1
C              NB1=NB+1
```

```
C               X0=ALOG(BMAX)+.2
C               NEW=1
C               DO 1 J=1,NB
C               I=NB1-J
C               X=X0-.2*J
C               ARG(I)=EXP(X)
C               Z(I)=ZLAGHO(X,ZF,TOL,L,NEW)/ARG(I)
C     1         NEW=0
C               ...
C     (3).  IF RESULTS ARE STORED IN ARRAYS ARG(I),Z(I),I=1,NB FOR
C     ARG IN (BMINA,BMAX), THEN THESE ARRAYS MAY BE USED, FOR EXAMPLE,
C     TO SPLINE-INTERPOLATE AT A DIFFERENT (LARGER OR SMALLER)
C     SPACING THAN USED IN THE LAGGED CONVOLUTION METHOD.
C     (4).  IF A DIFFERENT RANGE OF B IS DESIRED, THEN ONE MAY
C     ALWAYS RESTART THE ABOVE PROCEDURE IN (2) WITH A NEW
C     BMAX,BMIN AND BY SETTING NEW=1....
C     (5). ABSCISSA CORRESPONDING TO WEIGHT IS GENERATED TO SAVE STORAGE
C
C
      COMPLEX FUN,C,CMAX,SAVE
      DIMENSION KEY(193),SAVE(193),T(2),TMAX(2)
      DIMENSION YT(193),Y1(76),Y2(76),Y3(41)
      EQUIVALENCE (C,T(1)),(CMAX,TMAX(1))
      EQUIVALENCE (YT(1),Y1(1)),(YT(77),Y2(1)),(YT(153),Y3(1))
C--JO-EXTENDED FILTER WEIGHT ARRAYS:
      DATA Y1/
     1 5.8565723E-08, 7.1143477E-11,-7.8395565E-11, 8.7489547E-11,
     2-8.9007811E-11, 9.8790055E-11,-9.8675347E-11, 1.1118797E-10,
     3-1.0893474E-10, 1.2543400E-10,-1.1979399E-10, 1.4200767E-10,
     4-1.3106341E-10, 1.6153229E-10,-1.4238602E-10, 1.8486236E-10,
     5-1.5315381E-10, 2.1319755E-10,-1.6238115E-10, 2.4824144E-10,
     6-1.6850378E-10, 2.9243813E-10,-1.6909302E-10, 3.4934366E-10,
     7-1.6043759E-10, 4.2417082E-10,-1.3690001E-10, 5.2458440E-10,
     8-8.9946096E-11, 6.6188220E-10,-6.6964033E-12, 8.5276151E-10,
     9 1.3222770E-10, 1.1219600E-09, 3.5591442E-10, 1.5061956E-09,
     1 7.0795382E-10, 2.0600379E-09, 1.2535947E-09, 2.8646623E-09,
     2 2.0904225E-09, 4.0409101E-09, 3.3642886E-09, 5.7687700E-09,
     3 5.2930786E-09, 8.3164338E-09, 8.2021809E-09, 1.2083635E-08,
     4 1.2577400E-08, 1.7666303E-08, 1.9143895E-08, 2.5953011E-08,
     5 2.8983953E-08, 3.8268851E-08, 4.3712685E-08, 5.6590075E-08,
     6 6.5740136E-08, 8.3864288E-08, 9.8662323E-08, 1.2448811E-07,
     7 1.4784461E-07, 1.8501974E-07, 2.2129198E-07, 2.7524203E-07,
     8 3.3094739E-07, 4.0974828E-07, 4.9462868E-07, 6.1030809E-07,
     9 7.3891802E-07, 9.0939667E-07, 1.1034727E-06, 1.3554600E-06,
     1 1.6474556E-06, 2.0207696E-06, 2.4591294E-06, 3.0131400E-06/
      DATA Y2/
     1 3.6701680E-06, 4.4934101E-06, 5.4770076E-06, 6.7015208E-06,
     2 8.1726989E-06, 9.9954201E-06, 1.2194425E-05, 1.4909101E-05,
     3 1.8194388E-05, 2.2239184E-05, 2.7145562E-05, 3.3174088E-05,
     4 4.0499452E-05, 4.9486730E-05, 6.0421440E-05, 7.3822001E-05,
     5 9.0141902E-05, 1.1012552E-04, 1.3448017E-04, 1.6428337E-04,
     6 2.0062570E-04, 2.4507680E-04, 2.9930366E-04, 3.6560582E-04,
```

```
     7 4.4651421E-04, 5.4541300E-04, 6.6612648E-04, 8.1365181E-04,
     8 9.9374786E-04, 1.2138120E-03, 1.4824945E-03, 1.8107657E-03,
     9 2.2115938E-03, 2.7012675E-03, 3.2991969E-03, 4.0295817E-03,
     1 4.9214244E-03, 6.0106700E-03, 7.3405529E-03, 8.9643708E-03,
     2 1.0946310E-02, 1.3365017E-02, 1.6314985E-02, 1.9910907E-02,
     3 2.4289325E-02, 2.9612896E-02, 3.6070402E-02, 4.3876936E-02,
     4 5.3264829E-02, 6.4465091E-02, 7.7664144E-02, 9.2918324E-02,
     5 1.1000121E-01, 1.2811102E-01, 1.4543025E-01, 1.5832248E-01,
     6 1.6049224E-01, 1.4170064E-01, 8.8788108E-02,-1.1330934E-02,
     7-1.5331864E-01,-2.9094670E-01,-2.9084655E-01,-2.9708834E-02,
     8 3.9009601E-01, 1.7999785E-01,-4.1858139E-01, 1.5317216E-01,
     9 6.5184953E-02,-1.0751806E-01, 7.8429567E-02,-4.6019124E-02,
     1 2.5309571E-02,-1.3904823E-02, 7.8187120E-03,-4.5190369E-03/
      DATA Y3/
     1 2.6729062E-03,-1.6073718E-03, 9.7715622E-04,-5.9804407E-04,
     2 3.6749320E-04,-2.2635296E-04, 1.3960805E-04,-8.6172618E-05,
     3 5.3212947E-05,-3.2867888E-05, 2.0304203E-05,-1.2543926E-05,
     4 7.7499633E-06,-4.7882430E-06, 2.9584108E-06,-1.8278645E-06,
     5 1.1293571E-06,-6.9778174E-07, 4.3113019E-07,-2.6637753E-07,
     6 1.6458373E-07,-1.0168954E-07, 6.2829807E-08,-3.8819969E-08,
     7 2.3985272E-08,-1.4819520E-08, 9.1563774E-09,-5.6573541E-09,
     8 3.4954514E-09,-2.1597005E-09, 1.3343946E-09,-8.2447148E-10,
     9 5.0941033E-10,-3.1474631E-10, 1.9447072E-10,-1.2015685E-10,
     1 7.4241055E-11,-4.5871468E-11, 2.8343095E-11,-1.7513137E-11,
     2 6.9049613E-12/
C--$$ENDATA
C
      IF(NEW) 10,30,10
10    LAG=-1
      X0=-X-26.30455704
      DO 20 IR=1,193
20    KEY(IR)=0
30    LAG=LAG+1
      ZLAGH0=(0.0,0.0)
      CMAX=(0.0,0.0)
      L=0
      ASSIGN 110 TO M
      I=129
      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.146) 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=128
      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
130   ASSIGN 140 TO M
```

```
      I=147
      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.193) GO TO 200
      GO TO 190
150   ASSIGN 160 TO M
      I=1
      GO TO 200
160   IF(T(1).EQ.0.0.AND.T(2).EQ.0.0) GO TO 170
      I=I+1
      IF(I.LE.128) GO TO 200
170   ASSIGN 180 TO M
      I=193
      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.147) GO TO 200
190   RETURN
C--STORE/RETRIEVE ROUTINE (DONE INTERNALLY TO SAVE CALL'S)
200   LOOK=I+LAG
      IQ=LOOK/194
      IR=MOD(LOOK,194)
      IF(IR.EQ.0) IR=1
      IROLL=IQ*193
      IF(KEY(IR).LE.IROLL) GO TO 220
210   C=SAVE(IR)*YT(I)
      ZLAGH0=ZLAGH0+C
      L=L+1
      GO TO M,(110,120,140,160,180)
220   KEY(IR)=IROLL+IR
      SAVE(IR)=FUN(EXP(X0+FLOAT(LOOK)*.20))
      GO TO 210
      END

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

```
C                  OF A REAL ARGUMENT G.
C                NOTE: IF PARMS OTHER THAN G ARE REQUIRED, USE COMMON IN
C                    CALLING PROGRAM AND IN SUBPROGRAM FUN.
C                    THE COMPLEX FUNCTION FUN SHOULD BE A MONOTONE
C                    DECREASING FUNCTION AS THE ARGUMENT G BECOMES LARGE...
C                  FOR REAL-ONLY FUNCTIONS, SUBPROGRAM 'RLAGH1' IS ADVISED;
C                  HOWEVER, TWO REAL-FUNCTIONS F1(G),F2(G) MAY BE
C                  INTEGRATED IN PARALLEL BY WRITING FUN=CMPLX(F1(G),F2(G))
C         TOL=       REAL TOLERANCE EXCEPTED AT CONVOLVED TAILS--I.E.,
C                    IF FILTER*FUN<TOL*MAX, THEN REST OF TAIL IS TRUNCATED.
C                    THIS IS DONE AT BOTH ENDS OF FILTER.  TYPICALLY,
C                    TOL <= .0001 IS USUALLY OK--BUT THIS DEPENDS ON
C                    THE FUNCTION FUN AND PARAMETER X...IN GENERAL,
C                  A 'SMALLER TOL' WILL USUALLY RESULT IN 'MORE ACCURACY'
C                  BUT WITH 'MORE WEIGHTS' BEING USED.  TOL IS NOT DIRECTLY
C                  RELATED TO TRUNCATION ERROR, BUT GENERALLY SERVES AS AN
C                  APPROXIMATION INDICATOR... FOR VERY LARGE OR SMALL B,
C                  ONE SHOULD USE A SMALLER TOL THAN RECOMMENDED ABOVE...
C         L=         RESULTING NO. FILTER WTS. USED IN THE VARIABLE
C                    CONVOLUTION (L DEPENDS ON TOL AND FUN).
C                  MIN.L=15 AND MAX.L=236--WHICH COULD
C                    OCCUR IF TOL IS VERY SMALL AND/OR FUN NOT DECREASING
C                    VERY FAST...
C       * NEW=     1 IS NECESSARY 1ST TIME OR BRAND NEW X.
C                  0 FOR ALL SUBSEQUENT CALLS WHERE X=(LAST X)-0.20
C                    IS ASSUMED INTERNALLY BY THIS ROUTINE.
C                    NOTE:  IF THIS IS NOT TRUE, ROUTINE WILL
C                    STILL ASSUME X=(LAST X)-0.20 ANYWAY...
C                    IT IS THE USERS RESPONSIBILITY TO NORMALIZE
C                    BY CORRECT B=EXP(X) OUTSIDE OF CALL (SEE USAGE BELOW).
C                  THE LAGGED CONVOLUTION METHOD PICKS UP SIGNIFICANT
C                    TIME IMPROVEMENTS WHEN THE KERNEL IS NOT A
C                    SIMPLE ELEMENTARY FUNCTION...DUE TO INTERNALLY SAVING
C                    ALL KERNEL FUNCTION EVALUATIONS WHEN NEW=1...
C                    THEN WHEN NEW=0, ALL PREVIOUSLY CALCULATED
C                    KERNELS WILL BE USED IN THE LAGGED CONVOLUTION
C                    WHERE POSSIBLE, ONLY ADDING NEW KERNEL EVALUATIONS
C                    WHEN NEEDED (DEPENDS ON PARMS TOL AND FUN)
C
C--THE RESULTING COMPLEX CONVOLUTION SUM IS GIVEN IN ZLAGH1; THE HANKEL
C  TRANSFORM IS THEN ZLAGH1/B WHICH IS TO BE COMPUTED AFTER EXIT FROM
C  THIS ROUTINE.... WHERE B=EXP(X), X=ARGUMENT USED IN CALL...
C
C--USAGE-- 'ZLAGH1' IS CALLED AS FOLLOWS:
C     ...
C     COMPLEX Z,ZLAGH1,ZF
C     EXTERNAL ZF
C     ...
C     Z=ZLAGH1(ALOG(B),ZF,TOL,L,NEW)/B
C     ...
C     END
C     COMPLEX FUNCTION ZF(G)
```

```
C      ...USER SUPPLIED CODE...
C      END
C
C---NOTES:
C         (1).  EXP-UNDERFLOW'S MAY OCCUR IN EXECUTING THE SUBPROGRAM
C      BELOW; HOWEVER, THIS IS OK PROVIDED THE MACHINE SYSTEM SETS
C      ANY & ALL EXP-UNDERFLOW'S TO 0.0....
C      (2).  AS AN AID TO UNDERSTANDING & USING THE LAGGED CONVOLUTION
C      METHOD, LET BMAX>=BMIN>0 BE GIVEN. THEN IT CAN BE SHOWN
C      THAT THE ACTUAL NUMBER OF B'S IS NB=AINT(5.*ALOG(BMAX/BMIN))+1,
C      PROVIDED BMAX/BMIN>=1. THE USER MAY THEN ASSUME AN 'ADJUSTED'
C      BMINA=BMAX*EXP(-.2*(NB-1)).  THE METHOD GENERATES THE DECREASING
C      ARGUMENTS SPACED AS X=ALOG(BMAX),X-.2,X-.2*2,...,ALOG(BMINA).
C      FOR EXAMPLE, ONE MAY CONTROL THIS WITH THE CODE:
C               ...
C               NB=AINT(5.*ALOG(BMAX/BMIN))+1
C               NB1=NB+1
C               X0=ALOG(BMAX)+.2
C               NEW=1
C               DO 1 J=1,NB
C               I=NB1-J
C               X=X0-.2*J
C               ARG(I)=EXP(X)
C               Z(I)=ZLAGH1(X,ZF,TOL,L,NEW)/ARG(I)
C      1        NEW=0
C               ...
C      (3).  IF RESULTS ARE STORED IN ARRAYS ARG(I),Z(I),I=1,NB FOR
C      ARG IN (BMINA,BMAX), THEN THESE ARRAYS MAY BE USED, FOR EXAMPLE,
C      TO SPLINE-INTERPOLATE AT A DIFFERENT (LARGER OR SMALLER)
C      SPACING THAN USED IN THE LAGGED CONVOLUTION METHOD.
C      (4).  IF A DIFFERENT RANGE OF B IS DESIRED, THEN ONE MAY
C      ALWAYS RESTART THE ABOVE PROCEDURE IN (2) WITH A NEW
C      BMAX,BMIN AND BY SETTING NEW=1....
C      (5). ABSCISSA CORRESPONDING TO WEIGHT IS GENERATED TO SAVE STORAGE
C
C
        COMPLEX FUN,C,CMAX,SAVE
        DIMENSION KEY(236),SAVE(236),T(2),TMAX(2)
        DIMENSION WT(236),W1(76),W2(76),W3(76),W4(8)
        EQUIVALENCE (C,T(1)),(CMAX,TMAX(1))
        EQUIVALENCE (WT(1),W1(1)),(WT(77),W2(1)),(WT(153),W3(1)),
     1 (WT(229),W4(1))
C---J1-EXTENDED FILTER WEIGHT ARRAYS:
      DATA W1/
     1-8.8863805E-10, 1.1293811E-09,-1.2050872E-09, 1.2696232E-09,
     2-1.3223909E-09, 1.3642393E-09,-1.3969439E-09, 1.4225941E-09,
     3-1.4427475E-09, 1.4580582E-09,-1.4682563E-09, 1.4732179E-09,
     4-1.4735606E-09, 1.4719870E-09,-1.4727091E-09, 1.4828225E-09,
     5-1.5102619E-09, 1.5667752E-09,-1.6634522E-09, 1.8172900E-09,
     6-2.0412753E-09, 2.3595230E-09,-2.7861077E-09, 3.3592871E-09,
     7-4.0940172E-09, 5.0571015E-09,-6.2604109E-09, 7.8269461E-09,
     8-9.7514701E-09, 1.2267639E-08,-1.5312389E-08, 1.9339924E-08,
```

```
     9-2.4126297E-08, 3.0576829E-08,-3.8060204E-08, 4.8423732E-08,
     1-6.0051116E-08, 7.6787475E-08,-9.4700993E-08, 1.2192844E-07,
     2-1.4918997E-07, 1.9392737E-07,-2.3464786E-07, 3.0911127E-07,
     3-3.6815394E-07, 4.9413800E-07,-5.7554168E-07, 7.9301529E-07,
     4-8.9502818E-07, 1.2794292E-06,-1.3811469E-06, 2.0789668E-06,
     5-2.1069398E-06, 3.4103188E-06,-3.1584463E-06, 5.6639045E-06,
     6-4.6059955E-06, 9.5561672E-06,-6.4142855E-06, 1.6440205E-05,
     7-8.2010619E-06, 2.8945217E-05,-8.6348466E-06, 5.2317398E-05,
     8-3.9915035E-06, 9.7273612E-05, 1.5220520E-05, 1.8614373E-04,
     9 7.2023760E-05, 3.6620099E-04, 2.2062958E-04, 7.3874539E-04,
     1 5.8623480E-04, 1.5226779E-03, 1.4538718E-03, 3.1930365E-03/
      DATA W2/
     1 3.4640868E-03, 6.7790882E-03, 8.0328420E-03, 1.4484339E-02,
     2 1.8201316E-02, 3.0866143E-02, 4.0106549E-02, 6.4527872E-02,
     3 8.4285526E-02, 1.2773175E-01, 1.6020907E-01, 2.1948043E-01,
     4 2.3636305E-01, 2.4895051E-01, 1.2586300E-01,-5.1060445E-02,
     5-3.4376222E-01,-2.9042175E-01, 1.1564736E-01, 4.9253231E-01,
     6-4.6748595E-01, 1.5280945E-01, 3.3348541E-02,-8.2485252E-02,
     7 7.9740630E-02,-6.6934498E-02, 5.5150465E-02,-4.5868721E-02,
     8 3.8651958E-02,-3.2935834E-02, 2.8303994E-02,-2.4475127E-02,
     9 2.1259541E-02,-1.8526278E-02, 1.6182037E-02,-1.4158101E-02,
     1 1.2402225E-02,-1.0873526E-02, 9.5392016E-03,-8.3723743E-03,
     2 7.3506490E-03,-6.4551136E-03, 5.6696335E-03,-4.9803353E-03,
     3 4.3752213E-03,-3.8438703E-03, 3.3772023E-03,-2.9672872E-03,
     4 2.6071877E-03,-2.2908274E-03, 2.0128794E-03,-1.7686706E-03,
     5 1.5540998E-03,-1.3655666E-03, 1.1999089E-03,-1.0543497E-03,
     6 9.2644973E-04,-8.1406593E-04, 7.1531559E-04,-6.2854459E-04,
     7 5.5229955E-04,-4.8530352E-04, 4.2643446E-04,-3.7470650E-04,
     8 3.2925334E-04,-2.8931382E-04, 2.5421910E-04,-2.2338147E-04,
     9 1.9628455E-04,-1.7247455E-04, 1.5155278E-04,-1.3316889E-04,
     1 1.1701502E-04,-1.0282066E-04, 9.0348135E-05,-7.9388568E-05/
      DATA W3/
     1 6.9758436E-05,-6.1296474E-05, 5.3860978E-05,-4.7327436E-05,
     2 4.1586435E-05,-3.6541840E-05, 3.2109174E-05,-2.8214208E-05,
     3 2.4791718E-05,-2.1784390E-05, 1.9141864E-05,-1.6819888E-05,
     4 1.4779578E-05,-1.2986765E-05, 1.1411426E-05,-1.0027182E-05,
     5 8.8108499E-06,-7.7420630E-06, 6.8029235E-06,-5.9777053E-06,
     6 5.2525892E-06,-4.6154325E-06, 4.0555653E-06,-3.5636118E-06,
     7 3.1313335E-06,-2.7514911E-06, 2.4177236E-06,-2.1244417E-06,
     8 1.8667342E-06,-1.6402859E-06, 1.4413051E-06,-1.2664597E-06,
     9 1.1128220E-06,-9.7781908E-07, 8.5919028E-07,-7.5494920E-07,
     1 6.6335060E-07,-5.8286113E-07, 5.1213358E-07,-4.4998431E-07,
     2 3.9537334E-07,-3.4738689E-07, 3.0522189E-07,-2.6817250E-07,
     3 2.3561831E-07,-2.0701397E-07, 1.8188012E-07,-1.5979545E-07,
     4 1.4038968E-07,-1.2333746E-07, 1.0835294E-07,-9.5185048E-08,
     5 8.3613184E-08,-7.3443411E-08, 6.4505118E-08,-5.6648167E-08,
     6 4.9740428E-08,-4.3665572E-08, 3.8321109E-08,-3.3616717E-08,
     7 2.9472836E-08,-2.5819439E-08, 2.2594957E-08,-1.9745353E-08,
     8 1.7223359E-08,-1.4987869E-08, 1.3003472E-08,-1.1240058E-08,
     9 9.6723739E-09,-8.2794392E-09, 7.0438407E-09,-5.9509676E-09,
     1 4.9882405E-09,-4.1443813E-09, 3.4088114E-09,-2.7712762E-09/
      DATA W4/
```

```
     1 2.2217311E-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
130   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
150   ASSIGN 160 TO M
      I=1
      GO TO 200
160   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
170   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
190   RETURN
C--STORE/RETRIEVE ROUTINE (DONE INTERNALLY TO SAVE CALL'S)
200   LOOK=I+LAG
      IQ=LOOK/237
      IR=MOD(LOOK,237)
```

```
      IF(IR.EQ.0) IR=1
      IROLL=IQ*236
      IF(KEY(IR).LE.IROLL) GO TO 220
210   C=SAVE(IR)*WT(I)
      ZLAGH1=ZLAGH1+C
      L=L+1
      GO TO M,(110,120,140,160,180)
220   KEY(IR)=IROLL+IR
      SAVE(IR)=FUN(EXP(X0+FLOAT(LOOK)*.20))
      GO TO 210
      END

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

```
C      * NEW=     1 IS NECESSARY 1ST TIME OR BRAND NEW X.
C                 0 FOR ALL SUBSEQUENT CALLS WHERE X=(LAST X)-0.20
C                   IS ASSUMED INTERNALLY BY THIS ROUTINE.
C                   NOTE:  IF THIS IS NOT TRUE, ROUTINE WILL
C                   STILL ASSUME X=(LAST X)-0.20 ANYWAY...
C                   IT IS THE USERS RESPONSIBILITY TO NORMALIZE
C                   BY CORRECT B=EXP(X) OUTSIDE OF CALL (SEE USAGE BELOW).
C                 THE LAGGED CONVOLUTION METHOD PICKS UP SIGNIFICANT
C                   TIME IMPROVEMENTS WHEN THE KERNEL IS NOT A
C                   SIMPLE ELEMENTARY FUNCTION...DUE TO INTERNALLY SAVING
C                   ALL KERNEL FUNCTION EVALUATIONS WHEN NEW=1...
C                   THEN WHEN NEW=0, ALL PREVIOUSLY CALCULATED
C                   KERNELS WILL BE USED IN THE LAGGED CONVOLUTION
C                   WHERE POSSIBLE, ONLY ADDING NEW KERNEL EVALUATIONS
C                   WHEN NEEDED (DEPENDS ON PARMS TOL AND FUN)
C
C--THE RESULTING COMPLEX CONVOLUTION SUM IS GIVEN IN ZLAGFO; THE FOURIER
C  TRANSFORM IS THEN ZLAGFO/B WHICH IS TO BE COMPUTED AFTER EXIT FROM
C  THIS ROUTINE.... WHERE B=EXP(X), X=ARGUMENT USED IN CALL...
C
C--USAGE-- 'ZLAGFO' IS CALLED AS FOLLOWS:
C      ...
C      COMPLEX Z ZLAGFO,ZF
C      EXTERNAL ZF
C      ...
C      Z=ZLAGFO(ALOG(B),ZF,TOL,L,NEW)/B
C      ...
C      END
C      COMPLEX FUNCTION ZF(G)
C      ...USER SUPPLIED CODE...
C      END
C
C--NOTES:
C        (1).  EXP-UNDERFLOW'S MAY OCCUR IN EXECUTING THE SUBPROGRAM
C      BELOW; HOWEVER, THIS IS OK PROVIDED THE MACHINE SYSTEM SETS
C      ANY & ALL EXP-UNDERFLOW'S TO 0.0....
C      (2).  AS AN AID TO UNDERSTANDING & USING THE LAGGED CONVOLUTION
C      METHOD, LET BMAX>=BMIN>0 BE GIVEN. THEN IT CAN BE SHOWN
C      THAT THE ACTUAL NUMBER OF B'S IS NB=AINT(5.*ALOG(BMAX/BMIN))+1,
C      PROVIDED BMAX/BMIN>=1. THE USER MAY THEN ASSUME AN 'ADJUSTED'
C      BMINA=BMAX*EXP(-.2*(NB-1)).  THE METHOD GENERATES THE DECREASING
C      ARGUMENTS SPACED AS X=ALOG(BMAX),X-.2,X-.2*2,...,ALOG(BMINA).
C      FOR EXAMPLE, ONE MAY CONTROL THIS WITH THE CODE:
C               ...
C               NB=AINT(5.*ALOG(BMAX/BMIN))+1
C               NB1=NB+1
C               X0=ALOG(BMAX)+.2
C               NEW=1
C               DO 1 J=1,NB
C               I=NB1-J
C               X=X0-.2*J
C               ARG(I)=EXP(X)
```

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

```
     3 4.5083748E-04, 5.4213338E-04, 6.7315347E-04, 8.0800951E-04,
     4 1.0051938E-03, 1.2041401E-03, 1.5011708E-03, 1.7942344E-03,
     5 2.2421056E-03, 2.6730676E-03, 3.3490681E-03, 3.9815050E-03,
     6 5.0028666E-03, 5.9285668E-03, 7.4730905E-03, 8.8233510E-03,
     7 1.1160132E-02, 1.3119627E-02, 1.6653199E-02, 1.9472767E-02,
     8 2.4800811E-02, 2.8793704E-02, 3.6762063E-02, 4.2228780E-02,
     9 5.3905163E-02, 6.0804660E-02, 7.7081738E-02, 8.3874501E-02,
     1 1.0377190E-01, 1.0377718E-01, 1.1892208E-01, 9.0437429E-02/
      DATA Y3/
     1 7.1685138E-02,-3.9473064E-02,-1.5078720E-01,-4.0489859E-01,
     2-5.6018995E-01,-6.8050388E-01,-1.5094224E-01, 6.6304064E-01,
     3 1.3766748E+00,-8.0373222E-01,-1.0869629E+00, 1.2812892E+00,
     4-5.0341082E-01,-4.4274455E-02, 2.0913102E-01,-1.9999661E-01,
     5 1.5207664E-01,-1.0920260E-01, 7.8169956E-02,-5.6651561E-02,
     6 4.1611799E-02,-3.0880012E-02, 2.3072559E-02,-1.7311631E-02,
     7 1.3021442E-02,-9.8085025E-03, 7.3943529E-03,-5.5769518E-03,
     8 4.2073164E-03,-3.1745026E-03, 2.3954154E-03,-1.8076122E-03,
     9 1.3640816E-03,-1.0293934E-03, 7.7682952E-04,-5.8623518E-04,
     1 4.4240399E-04,-3.3386183E-04, 2.5195025E-04,-1.9013541E-04,
     2 1.4348659E-04,-1.0828284E-04, 8.1716174E-05,-6.1667509E-05,
     3 4.6537684E-05,-3.5119887E-05, 2.6503388E-05,-2.0000904E-05,
     4 1.5093768E-05,-1.1390572E-05, 8.5959318E-06,-6.4869407E-06,
     5 4.8953713E-06,-3.6942830E-06, 2.7878625E-06,-2.1038241E-06,
     6 1.5875917E-06,-1.1980090E-06, 9.0398030E-07,-6.8208296E-07,
     7 5.1458650E-07,-3.8817581E-07, 2.9272267E-07,-2.2067921E-07,
     8 1.6623514E-07,-1.2514102E-07, 9.4034535E-08,-7.0556837E-08,
     9 5.2741581E-08,-3.9298610E-08, 2.9107255E-08,-2.1413893E-08,
     1 1.5742032E-08,-1.1498608E-08, 8.7561571E-09,-7.2959446E-09/
      DATA Y4/
     1 6.8816619E-09,-8.9679825E-09, 1.4258275E-08,-1.9564299E-08,
     2 2.0235313E-08,-1.4725545E-08, 5.4632820E-09, 3.5995580E-09,
     3-9.5287133E-09, 1.1460041E-08,-1.0250532E-08, 7.4641748E-09,
     4-4.4703465E-09, 2.0499053E-09,-4.4806353E-10,-4.0374336E-10,
     5 7.0321001E-10,-6.7067960E-10, 4.9130404E-10,-2.8840747E-10,
     6 1.2373144E-10,-1.5260443E-11,-4.2027559E-11, 6.1885474E-11,
     7-5.9273937E-11, 4.6588766E-11,-3.2054182E-11, 1.9831637E-11,
     8-1.1210098E-11, 5.9567021E-12,-3.2427812E-12, 2.1353868E-12,
     9-1.8476851E-12, 1.8438474E-12,-1.8362842E-12, 1.7241847E-12,
     1-1.5161479E-12, 1.2627657E-12,-1.0129176E-12, 7.9578625E-13,
     2-6.2131435E-13, 4.8745900E-13,-3.8703630E-13, 3.1172547E-13,
     3-2.5397802E-13, 2.0824130E-13,-1.7123163E-13, 1.4113344E-13,
     4-1.1687986E-13, 9.7664016E-14,-8.2977176E-14, 7.2515267E-14,
     5-5.6047478E-14/
C--$$ENDATA
      IF(NEW) 10,30,10
10    LAG=-1
      X0=-X-30.30251236
      DO 20 IR=1,281
20    KEY(IR)=0
30    LAG=LAG+1
      ZLAGF0=(0.0,0.0)
      CMAX=(0.0,0.0)
```

```
      L=0
      ASSIGN 110 TO M
      I=149
      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.170) 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=148
      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
 130  ASSIGN 140 TO M
      I=171
      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.281) GO TO 200
      GO TO 190
 150  ASSIGN 160 TO M
      I=1
      GO TO 200
 160  IF(T(1).EQ.0.0.AND.T(2).EQ.0.0) GO TO 170
      I=I+1
      IF(I.LE.148) GO TO 200
 170  ASSIGN 180 TO M
      I=281
      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.171) GO TO 200
 190  RETURN
C--STORE/RETRIEVE ROUTINE (DONE INTERNALLY TO SAVE CALL'S)
 200  LOOK=I+LAG
      IQ=LOOK/282
      IR=MOD(LOOK,282)
      IF(IR.EQ.0) IR=1
      IROLL=IQ*281
      IF(KEY(IR).LE.IROLL) GO TO 220
 210  C=SAVE(IR)*YT(I)
      ZLAGFO=ZLAGFO+C
      L=L+1
      GO TO M,(110,120,140,160,180)
 220  KEY(IR)=IROLL+IR
      SAVE(IR)=FUN(EXP(XO+FLOAT(LOOK)*.20))
      GO TO 210
      END
```

```
      COMPLEX FUNCTION ZLAGF1(X,FUN,TOL,L,NEW)
C--*** A SPECIAL LAGGED* CONVOLUTION METHOD TO COMPUTE THE
C  INTEGRAL FROM 0 TO INFINITY OF 'FUN(G)*SIN(G*B)*DG' DEFINED AS THE
C  COMPLEX FOURIER SINE TRANSFORM WITH ARGUMENT X(=ALOG(B))
C  BY CONVOLUTION FILTERING WITH COMPLEX FUNCTION 'FUN'--AND
C  USING A VARIABLE CUT-OFF METHOD WITH EXTENDED FILTER TAILS....
C
C--REF: ANDERSON, W.L., 1975, NTIS REPT. PB-242-800.
C
C--PARAMETERS:
C
C     * X     = REAL ARGUMENT(=ALOG(B) AT CALL) OF THE FOURIER TRANSFORM
C               'ZLAGF1' IS USEFUL ONLY WHEN X=(LAST X)-.20 *** I.E.,
C               SPACED SAME AS FILTER USED--IF THIS IS NOT CONVENIENT,
C               THEN SUBPROGRAM 'ZFOUR1' IS ADVISED FOR GENERAL USE.
C               (ALSO SEE PARM 'NEW' & NOTES (2)-(4) BELOW).
C     FUN(G)= EXTERNAL DECLARED COMPLEX FUNCTION NAME (USER SUPPLIED)
C               OF A REAL ARGUMENT G.
C             NOTE: IF PARMS OTHER THAN G ARE REQUIRED, USE COMMON IN
C                 CALLING PROGRAM AND IN SUBPROGRAM FUN.
C                 THE COMPLEX FUNCTION FUN SHOULD BE A MONOTONE
C                 DECREASING FUNCTION AS THE ARGUMENT G BECOMES LARGE...
C               FOR REAL-ONLY FUNCTIONS, SUBPROGRAM 'RLAGF1' IS ADVISED;
C               HOWEVER, TWO REAL-FUNCTIONS F1(G),F2(G) MAY BE
C               INTEGRATED IN PARALLEL BY WRITING FUN=CMPLX(F1(G),F2(G))
C       TOL=      REAL TOLERANCE EXCEPTED AT CONVOLVED TAILS--I.E.,
C                 IF FILTER*FUN<TOL*MAX, THEN REST OF TAIL IS TRUNCATED.
C                 THIS IS DONE AT BOTH ENDS OF FILTER.  TYPICALLY,
C                 TOL <= .0001 IS USUALLY OK--BUT THIS DEPENDS ON
C                 THE FUNCTION FUN AND PARAMETER X...IN GENERAL,
C               A 'SMALLER TOL' WILL USUALLY RESULT IN 'MORE ACCURACY'
C               BUT WITH 'MORE WEIGHTS' BEING USED.  TOL IS NOT DIRECTLY
C               RELATED TO TRUNCATION ERROR, BUT GENERALLY SERVES AS AN
C               APPROXIMATION INDICATOR... FOR VERY LARGE OR SMALL B,
C               ONE SHOULD USE A SMALLER TOL THAN RECOMMENDED ABOVE...
C       L=        RESULTING NO. FILTER WTS. USED IN THE VARIABLE
C                 CONVOLUTION (L DEPENDS ON TOL AND FUN).
C               MIN.L=20 AND MAX.L=266--WHICH COULD
C                 OCCUR IF TOL IS VERY SMALL AND/OR FUN NOT DECREASING
C                 VERY FAST...
C     * NEW=    1 IS NECESSARY 1ST TIME OR BRAND NEW X.
C               0 FOR ALL SUBSEQUENT CALLS WHERE X=(LAST X)-0.20
C                 IS ASSUMED INTERNALLY BY THIS ROUTINE.
C                 NOTE:  IF THIS IS NOT TRUE, ROUTINE WILL
C                 STILL ASSUME X=(LAST X)-0.20 ANYWAY...
C                 IT IS THE USERS RESPONSIBILITY TO NORMALIZE
C                 BY CORRECT B=EXP(X) OUTSIDE OF CALL (SEE USAGE BELOW).
C               THE LAGGED CONVOLUTION METHOD PICKS UP SIGNIFICANT
C                 TIME IMPROVEMENTS WHEN THE KERNEL IS NOT A
C                 SIMPLE ELEMENTARY FUNCTION...DUE TO INTERNALLY SAVING
C                 ALL KERNEL FUNCTION EVALUATIONS WHEN NEW=1...
C                 THEN WHEN NEW=0, ALL PREVIOUSLY CALCULATED
```

```
C                 KERNELS WILL BE USED IN THE LAGGED CONVOLUTION
C                 WHERE POSSIBLE, ONLY ADDING NEW KERNEL EVALUATIONS
C                 WHEN NEEDED (DEPENDS ON PARMS TOL AND FUN)
C
C---THE RESULTING COMPLEX CONVOLUTION SUM IS GIVEN IN ZLAGF1; THE FOURIER
C  TRANSFORM IS THEN ZLAGF1/B WHICH IS TO BE COMPUTED AFTER EXIT FROM
C  THIS ROUTINE.... WHERE B=EXP(X), X=ARGUMENT USED IN CALL...
C
C---USAGE-- 'ZLAGF1' IS CALLED AS FOLLOWS:
C     ...
C     COMPLEX Z,ZLAGF1,ZF
C     EXTERNAL ZF
C     ...
C     Z=ZLAGF1(ALOG(B),ZF,TOL,L,NEW)/B
C     ...
C     END
C     COMPLEX FUNCTION ZF(G)
C     ...USER SUPPLIED CODE...
C     END
C
C---NOTES:
C       (1).  EXP-UNDERFLOW'S MAY OCCUR IN EXECUTING THE SUBPROGRAM
C     BELOW; HOWEVER, THIS IS OK PROVIDED THE MACHINE SYSTEM SETS
C     ANY & ALL EXP-UNDERFLOW'S TO 0.0....
C     (2).  AS AN AID TO UNDERSTANDING & USING THE LAGGED CONVOLUTION
C     METHOD, LET BMAX>=BMIN>0 BE GIVEN. THEN IT CAN BE SHOWN
C     THAT THE ACTUAL NUMBER OF B'S IS NB=AINT(5.*ALOG(BMAX/BMIN))+1,
C     PROVIDED BMAX/BMIN>=1. THE USER MAY THEN ASSUME AN 'ADJUSTED'
C     BMINA=BMAX*EXP(-.2*(NB-1)).  THE METHOD GENERATES THE DECREASING
C     ARGUMENTS SPACED AS X=ALOG(BMAX),X-.2,X-.2*2,...,ALOG(BMINA).
C     FOR EXAMPLE, ONE MAY CONTROL THIS WITH THE CODE:
C                ...
C                NB=AINT(5.*ALOG(BMAX/BMIN))+1
C                NB1=NB+1
C                X0=ALOG(BMAX)+.2
C                NEW=1
C                DO 1 J=1,NB
C                I=NB1-J
C                X=X0-.2*J
C                ARG(I)=EXP(X)
C                Z(I)=ZLAGF1(X,ZF,TOL,L,NEW)/ARG(I)
C     1          NEW=0
C                ...
C     (3).  IF RESULTS ARE STORED IN ARRAYS ARG(I),Z(I),I=1,NB FOR
C     ARG IN (BMINA,BMAX), THEN THESE ARRAYS MAY BE USED, FOR EXAMPLE,
C     TO SPLINE-INTERPOLATE AT A DIFFERENT (LARGER OR SMALLER)
C     SPACING THAN USED IN THE LAGGED CONVOLUTION METHOD.
C     (4).  IF A DIFFERENT RANGE OF B IS DESIRED, THEN ONE MAY
C     ALWAYS RESTART THE ABOVE PROCEDURE IN (2) WITH A NEW
C     BMAX,BMIN AND BY SETTING NEW=1....
C     (5). ABSCISSA CORRESPONDING TO WEIGHT IS GENERATED TO SAVE STORAGE
C
```

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

```
     4 4.9485823E-06, 7.3545473E-06, 1.1001083E-05, 1.6380539E-05,
     5 2.4469550E-05, 3.6469246E-05, 5.4441527E-05, 8.1176726E-05,
     6 1.2113828E-04, 1.8066494E-04, 2.6954609E-04, 4.0202288E-04,
     7 5.9969995E-04, 8.9437312E-04, 1.3338166E-03, 1.9886697E-03,
     8 2.9643943E-03, 4.4168923E-03, 6.5773518E-03, 9.7855105E-03,
     9 1.4539361E-02, 2.1558670E-02, 3.1871864E-02, 4.6903518E-02,
     1 6.8559512E-02, 9.9170152E-02, 1.4120770E-01, 1.9610835E-01,
     2 2.6192603E-01, 3.2743321E-01, 3.6407406E-01, 3.1257559E-01,
     3 9.0460168E-02,-3.6051039E-01,-8.6324760E-01,-8.1178720E-01,
     4 5.2205241E-01, 1.5449873E+00,-1.1817933E+00,-2.6759896E-01,
     5 8.0869203E-01,-6.2757149E-01, 3.4062630E-01,-1.5885304E-01,
     6 7.0472984E-02,-3.1624462E-02, 1.4894068E-02,-7.4821176E-03,
     7 4.0035936E-03,-2.2543784E-03, 1.3160358E-03,-7.8636604E-04,
     8 4.7658745E-04,-2.9125817E-04, 1.7885105E-04,-1.1012416E-04,
     9 6.7910334E-05,-4.1914054E-05, 2.5881544E-05,-1.5985851E-05,
     1 9.8751880E-06,-6.1008526E-06, 3.7692543E-06,-2.3287953E-06/
      DATA W4/
     1 1.4388425E-06,-8.8899353E-07, 5.4926991E-07,-3.3937048E-07,
     2 2.0968284E-07,-1.2955437E-07, 8.0046336E-08,-4.9457371E-08,
     3 3.0557711E-08,-1.8880390E-08, 1.1665454E-08,-7.2076428E-09,
     4 4.4533423E-09,-2.7515696E-09, 1.7001092E-09,-1.0504494E-09,
     5 6.4904567E-10,-4.0102999E-10, 2.4778763E-10,-1.5310321E-10,
     6 9.4600354E-11,-5.8453314E-11, 3.6119400E-11,-2.2320056E-11,
     7 1.3793460E-11,-8.5242656E-12, 5.2675102E-12,-3.2543076E-12,
     8 2.0097689E-12,-1.2405412E-12, 7.6530538E-13,-4.7191929E-13,
     9 2.9084993E-13,-1.7923661E-13, 1.1018948E-13,-6.7885902E-14,
     1 4.2025050E-14,-2.1314731E-14/
C--$$ENDATA
C
      IF(NEW) 10,30,10
10    LAG=-1
      X0=-X-38.30455704
      DO 20 IR=1,266
20    KEY(IR)=0
30    LAG=LAG+1
      ZLAGF1=(0.0,0.0)
      CMAX=(0.0,0.0)
      L=0
      ASSIGN 110 TO M
      I=191
      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.208) 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=190
      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
130   ASSIGN 140 TO M
      I=209
      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.266) GO TO 200
      GO TO 190
150   ASSIGN 160 TO M
      I=1
      GO TO 200
160   IF(T(1).EQ.0.0.AND.T(2).EQ.0.0) GO TO 170
      I=I+1
      IF(I.LE.190) GO TO 200
170   ASSIGN 180 TO M
      I=266
      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.209) GO TO 200
190   RETURN
C--STORE/RETRIEVE ROUTINE (DONE INTERNALLY TO SAVE CALL'S)
200   LOOK=I+LAG
      IQ=LOOK/267
      IR=MOD(LOOK,267)
      IF(IR.EQ.0) IR=1
      IROLL=IQ*266
      IF(KEY(IR).LE.IROLL) GO TO 220
210   C=SAVE(IR)*WT(I)
      ZLAGF1=ZLAGF1+C
      L=L+1
      GO TO M,(110,120,140,160,180)
220   KEY(IR)=IROLL+IR
      SAVE(IR)=FUN(EXP(X0+FLOAT(LOOK)*.20))
      GO TO 210
      END

      SUBROUTINE IKS(B8,I1K1,IKDIF)
C--COMPUTE MODIFIED BESSEL FUNCTION (I & K) SPECIAL COMBINATIONS FOR
C  PARAMETERS
C      B8    = DOUBLE PRECISION ARGUMENT (=B/DSQRT(2.D0) HERE)
C      I1K1  = I1*K1 COMPLEX RESULT
C      IKDIF = 4*I1*K1-(B8*DSQRT(I))*(I0*K1-I1*K0) COMPLEX RESULT DONE IN
C              DP BEFORE CMPLX"ING.
C--SUBROUTINE KELVIN CALLED
C
      DOUBLE PRECISION B8,BB(8),BETA,Q1,Q2,R1,R2
      COMPLEX I1K1,IKDIF,CAMBDA,DENOM,DENOM1,TERM0,TERM1,TERM11
      COMPLEX S11,SI0,SI1,SK0,SK1,ONE
      DATA ONE/(1.0,0.0)/
      IF(B8.GT.20.D0) GO TO 10
      CALL KELVIN(B8,8,BB)
```

```
      Q1=-BB(6)*BB(8)+BB(5)*BB(7)
      Q2= BB(5)*BB(8)+BB(6)*BB(7)
      I1K1=CMPLX(SNGL(Q1),SNGL(Q2))
      R1=-BB(1)*BB(8)-BB(2)*BB(7) -
     &     BB(6)*BB(3)-BB(5)*BB(4)
      R2=-BB(2)*BB(8)+BB(1)*BB(7) +
     &     BB(5)*BB(3)-BB(6)*BB(4)
      BETA=.70710678118654750D0*B8
      Q1=4.0D0*Q1-BETA*(R1-R2)
      Q2=4.0D0*Q2-BETA*(R1+R2)
      IKDIF=CMPLX(SNGL(Q1),SNGL(Q2))
      RETURN
   10 B=SNGL(B8/0.70710678118654750D0)
      TOL=1.E-6
C--FOR LARGE ARGUMENTS, USE ABRAMOWITZ AND STEGUN
C          ASYMPTOTIC FORMULAS FOR LARGE ARGUMENTS
C          9.7.1 THROUGH 9.7.5, P. 377-378.
      CAMBDA=B*CMPLX(1.0,1.0)/2.
      IKDIF=CMPLX(100.,0.)
      ISIGN=1
      DENOM=8.*CAMBDA
      DENOM1=(2.*CAMBDA)**2
      NODD=1
      TERM0=ONE
      TERM1=ONE
      TERM11=ONE
      S11=ONE
      SI0=ONE
      SI1=ONE
      SK0=ONE
      SK1=ONE
    1 NODD2=NODD*NODD
      OIKDIF=CABS(IKDIF)
      TERM1=TERM1*CMPLX(4.-NODD2,0.)/DENOM
      TERM0=TERM0*CMPLX(-FLOAT(NODD2),0.)/DENOM
      TERM11=TERM11*CMPLX(NODD*(4.-NODD2)/(NODD+1),0.)/DENOM1
      ISIGN=-ISIGN
      S11=S11+ISIGN*TERM11
      SI0=SI0+ISIGN*TERM0
      SI1=SI1+ISIGN*TERM1
      SK0=SK0+TERM0
      SK1=SK1+TERM1
      IKDIF=SI0*SK1-SK0*SI1
      NODD=NODD+2
      IF(ABS(OIKDIF-CABS(IKDIF)).GT.TOL) GO TO 1
      I1K1=S11/(CAMBDA*CMPLX(2.,0.))
      IKDIF=CMPLX(4.,0.)*I1K1-IKDIF/CMPLX(2.,0.)
      RETURN
      END

      SUBROUTINE KELVIN(X;M,B)
C--COMPUTES M(.LE.8) KELVIN FUNCTIONS (ORDERS 0,1) CONSECUTIVELY STORED
```

```
C   IN ARRAY B(M) WHERE:
C
C      X      = DP-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 DP-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)
      REAL*8 B(8),CN(8),SN(8)
      DATA CN        /.7071067811865475D0,0.D0,-.7071067811865475D0,
     * -1.D0,-.7071067811865475D0,0.D0,.7071067811865475D0,1.D0/,
     * SN   /.7071067811865475D0,1.D0,.7071067811865475D0,0.D0,
     * -.7071067811865475D0,-1.D0,-.7071067811865475D0,0.D0/
      DATA PI4/.7853981633974483D0/,R22/.7071067811865475D0/,
     * E/0.5D-14/,
     * PI1/.3183098861837907D0/
      IF(M.LT.1.OR.M.GT.8.OR.X.LE.0.0D0) GO TO 9
      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
      T15=0.0D0
      T26=0.0D0
      T75=0.0D0
      T86=0.0D0
      IF(M.EQ.1) GO TO 100
      T2=X2**2
      S2=T2
      IF(M.EQ.2) GO TO 100
      T5=1.5D0
      S5=T1*T5
      IF(M.EQ.3) GO TO 100
      T6=1.0D0
      S6=T2
```

```
      IF(M.EQ.4) GO TO 100
      T3=-0.5D0*X2**3
      S3=T3
      T4=X2
      S4=T4
      IF(M.LE.6) GO TO 100
      T7=-0.25D0*X2**3
      S7=2.0D0*T7*T5
      T8=X2
      S8=T8
  100 TK=2.0D0
  101 TK2=TK+TK
      TK21=TK2-1.0D0
      TK22=TK2-2.0D0
      RK2=1.0D0/TK2
      RK21=1.0D0/TK21
      RK22=1.0D0/TK22
      R1=-X4*(RK21*RK2)**2
      T1=T1*R1
      S1=S1+T1
      IF(M.EQ.1) GO TO 200
      R2=-X4*(RK22*RK21)**2
      T2=T2*R2
      S2=S2+T2
      IF(M.EQ.2) GO TO 200
      T5=T5+RK21+RK2
      T15=T1*T5
      S5=S5+T15
      IF(M.EQ.3) GO TO 200
      T6=T6+RK22+RK21
      T26=T2*T6
      S6=S6+T26
      IF(M.EQ.4) GO TO 200
      T3=T3*(-X4*(RK22*RK21**2*RK2))
      S3=S3+T3
      T4=T4*(-X4*RK22**2*RK21/(TK2-3.0D0))
      S4=S4+T4
      IF(M.LE.6) GO TO 200
      T7=T7*R1
      T75=TK2*T7*T5
      S7=S7+T75
      T8=T8*R2
      T86=TK21*T8*T6
      S8=S8+T86
  200 TK=TK+1.0D0
      IF(DABS(T1).GT.E.OR.DABS(T2).GT.E.OR.DABS(T15).GT.E.OR.
     * DABS(T26).GT.E.OR.DABS(T3).GT.E.OR.DABS(T4).GT.E.OR.
     * DABS(T75).GT.E.OR.DABS(T86).GT.E) GO TO 101
      B(1)=1.0D0+S1
      IF(M.EQ.1) GO TO 9
      B(2)=S2
      IF(M.EQ.2) GO TO 9
```

```
      C=0.11593151565841240D0-DLOG(X)
      B(3)=C*B(1)+PI4*B(2)+S5
      IF(M.EQ.3) GO TO 9
      B(4)=C*B(2)-PI4*B(1)+S6
      IF(M.EQ.4) GO TO 9
      B(5)=R22*(S3-S4)
      IF(M.EQ.5) GO TO 9
      B(6)=R22*(S3+S4)
      IF(M.EQ.6) GO TO 9
      S7=C*S3-B(1)/X+PI4*S4+S7
      S8=C*S4-B(2)/X-PI4*S3+S8
      B(7)=R22*(S7-S8)
      IF(M.EQ.7) GO TO 9
      B(8)=R22*(S7+S8)
    9 RETURN
C--GENERAL ASYMPTOTIC FORM FOR NU=0,1:
    8 NU=0
      X2=R22*X
      X8=8.0D0*X
      SX=DSQRT(X)
      EX2=DEXP (-X2)
      C1=1.2533141373155000D0*EX2/SX
      C2=1.0D0/(2.5066282746310010D0*SX*EX2+1.0D-38)
      MAXK=30
      IF(X.LT.15.0D0) MAXK=X+X
    1 XNU=NU
      XMU=4.0D0*XNU
      ALP=X2+PI4*(XNU+XNU-0.5D0)
      BETA=ALP+PI4
      CB=DCOS(BETA)
      CA=DCOS(ALP)
      SB=DSIN(BETA)
      SA=DSIN(ALP)
      N4=4*NU
      FM=0.0D0
      FP=0.0D0
      GM=0.0D0
      GP=0.0D0
      TM=1.0D0
      TP=1.0D0
      K=1
    2 TK=K
      T=(XMU-(TK+TK-1.0D0)**2)/(TK*X8)
      TPL=DABS(TP)
      TP=-TP*T
      IF(DABS(TP).GT.TPL) GO TO 21
      TM=TM*T
      N=MOD(K,8)
      IF(N.EQ.0) N=8
      T1=TP*CN(N)
      FP=FP+T1
      T2=TM*CN(N)
```

```
      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

      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
      DIMENSION Y(1),B(1),C(1),D(1),E(1),F(1)
      IF(NY.LE.2) GO TO 4
      N=NY-3
      P=0.0
      Q=0.0
      R=0.0
      S=0.0
```

```
      T=0.0
      DO 1 I=1,N
      U=P*R
      B(I)=1.0/(66.0-U*R-Q)
      R=26.0-U
      C(I)=R
      D(I)=Y(I+3)-3.0*(Y(I+2)-Y(I+1))-Y(I)-U*S-Q*T
      Q=P
      P=B(I)
      T=S
      S=D(I)
1     CONTINUE
      D(N+2)=0.0
      N1=N+1
      D(N1)=0.0
      DO 2 J=1,N
      I=N1-J
      D(I)=(D(I)-C(I)*D(I+1)-D(I+2))*B(I)
2     CONTINUE
      N=NY-1
      Q=0.0
      V=D(1)
      T=V
      R=V
      DO 3 I=2,N
      P=Q
      Q=R
      R=D(I)
      S=T
      T=P-Q-Q+R
      F(I)=T
      U=5.0*(-P+Q)
      E(I)=U
      D(I)=10.0*(P+Q)
      C(I)=0.5*(Y(I+1)+Y(I-1)+S-T)-Y(I)-U
      B(I)=0.5*(Y(I+1)-Y(I-1)-S-T)-D(I)
3     CONTINUE
      F(1)=V
      E(1)=0.0
      E(NY)=0.0
      D(1)=0.0
      D(NY)=0.0
      C(1)=C(2)-10.0*V
      C(NY)=C(NY-1)+10.0*T
      B(1)=Y(2)-Y(1)-C(1)-V
      B(NY)=Y(NY)-Y(NY-1)+C(NY)-T
4     RETURN
      END

      SUBROUTINE QPOINT(NY,Y,B,C,D,E,F,X1,DELX,XX,YY)
C   GIVEN THE QUINTIC SPLINE COEFF"S B(*),C(*),D(*),E(*),F(*) AS
C   OBTAINED FROM SUBR 'QUINT', AND GIVEN NY OBS. DATA Y(NY) EQUALLY
```

```
C  SPACED BY DELX STARTING AT X1, THEN 'QPOINT' INTERPOLATES
C  YY AT ANY XX IN (X1,X1+(NY-1)*DELX).
C
      DIMENSION Y(1),B(1),C(1),D(1),E(1),F(1)
      XMAX=X1+(NY-1)*DELX
      IF(XX.LT.X1.OR.XX.GT.XMAX) GO TO 2
      I=(XX-X1)/DELX+1
      XI=X1+(I-1)*DELX
      T=(XX-XI)/DELX
      YY=((((F(I)*T+E(I))*T+D(I))*T+C(I))*T+B(I))*T+Y(I)
1     RETURN
2     WRITE(6,3) XX,X1,XMAX
3     FORMAT('0QPOINT ERROR-- XX=',E16.8,' NOT IN CLOSED INTERVAL (',
     & E16.8,',',E16.8,')')
      GO TO 1
      END

      COMPLEX FUNCTION CANC4(A1,B1,EP,M,N,FUN,MF,ESUM)
C--COMPLEX FUNCTION DEFINITE INTEGRATION BY
C     ADAPTIVE QUADRATURE USING NEWTON-COTES NO. 4 (N=1,2,3), OR
C     AUTOMATIC GAUSSIAN QUADRATURE (N=4,5).....
C  A GENERAL  ROUTINE  IN SINGLE-PRECISION COMPLEX...
C     BY W.L.ANDERSON, U.S.GEOLOGICAL SURVEY, DENVER, COLORADO.
C--PARAMETERS--
C     A1    = LOWER LIMIT OF INTEGRATION (REAL*4)
C     B1    = UPPER LIMIT OF INTEGRATION (REAL*4)
C     EP    = DESIRED REL. ERROR (REAL*4) IN COMPLEX RESULT 'CANC4'.
C              (FOR BOTH RE & IM PARTS OF CANC4)
C     M     = RESULTING NUMBER OF COMPLEX 'FUN EVALUATIONS'
C     N     = ERROR TEST TYPE:
C           = 1 FOR ABS. ERROR TEST (NOT GENERALLY RECOMMENDED)
C           = 2 FOR 'L-ONE' ERROR TEST
C           = 3 FOR 'L-INFINITY' ERROR TEST
C           = 4 FOR REL. ERROR TEST USING ADAPTIVE GAUSSIAN QUADRATURE
C           = 5 FOR REL. ERROR TEST USING NON-ADAPTIVE GAUSSIAN QUAD...
C     NOTE:  N=1,2,OR 3 USES ADAPTIVE NEWTON-COTES QUADRATURE, AND
C              N=4 OR 5 USES ADAPTIVE OR NON-ADAPTIVE GAUSS QUADRATURES
C     FUN   = EXTERNAL COMPLEX FUNCTION NAME (COMPLEX*8)
C     MF    = MAX. FUN EVALUATIONS ALLOWED BEFORE ACCEPTING COMPLEX
C             RESULT 'CANC4' WITH M.GE.MF....
C     ESUM  = ACTUAL COMPLEX ERROR ACHIEVED AT EXIT....
C--SUBPROGRAMS CALLED:  CQSUBA,CQSUB (WHICH CALLS CQUAD)
C     (THESE ARE FOR N=4 OR 5 GAUSSIAN QUADRATURES)
C
      COMPLEX FUN,ESUM,TSUM,FA, F,X,Z, CQSUBA,CQSUB,
     1 F1,FS,F3,FM,F2,FT,F4,FB,FTP,FBP,FMAX,FTST,EST,AEST,EST1,EST2,AEST
     21,AEST2,ABSAR,DELTA,DIFF,DAFT,SUM
      DIMENSION F2(30),F4(30),FTP(30),FBP(30),FTST(5),EST2(30),NRTR(30)
      DIMENSION AEST2(30),XB(30)
      DIMENSION FMX(2)
      EXTERNAL FUN
      EQUIVALENCE (FMX(1),FMAX)
```

```
C--STATEMENT FUNCTION GOES HERE ON HONEYWELL MULTICS SYSTEM             00025690
      F(X)=CMPLX(ABS(REAL(X)),ABS(AIMAG(X)))                            00025700
C     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 (10,10,10,400,500),N                                        00025740
10       A=A1                                                           00025750
      B=B1                                                              00025760
      EPS=EP*63.0                                                       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(FA)                                                        00025880
      FTST(1)=FMAX                                                      00025890
      FTST(2)=F(FS)                                                     00025900
      FTST(3)=F(FM)                                                     00025910
      FTST(4)=F(FT)                                                     00025920
      FTST(5)=F(FB)                                                     00025930
      DO 100 I=2,5                                                      00025940
      IF(FMX(1).GE.REAL(FTST(I)))GO TO 101                              00025950
      FMX(1)=REAL(FTST(I))                                              00025960
  101 IF(FMX(2).GE.AIMAG(FTST(I)))GO TO 100                             00025970
      FMX(2)=AIMAG(FTST(I))                                             00025980
  100 CONTINUE                                                          00025990
      EST=(7.0  *(FA+FB)+32.0  *(FS+FT)+12.0  *FM)*DA/90.0              00026000
      ABSAR=(7.0  *(FTST(1)+FTST(5))+32.0  *(FTST(2)+FTST(4))+12.0  *FTS00026010
     1T(3))*DA/90.0                                                     00026020
      AEST=ABSAR                                                        00026030
C     1=RECUR                                                           00026040
    1 SX=(DA/(2.0  **LVL))/90.0                                         00026050
      F1=FUN((7.0  *A+B)/8.0  )                                         00026060
      F3=FUN((5.0  *A+3.0  *B)/8.0  )                                   00026070
      F2(LVL)=FUN((3.0  *A+5.0  *B)/8.0  )                              00026080
      F4(LVL)=FUN((A+7.0  *B)/8.0  )                                    00026090
      EST1=SX*(7.0  *(FA+FM)+32.0  *(F1+F3)+12.0  *FS)                  00026100
      FBP(LVL)=FB                                                       00026110
      FTP(LVL)=FT                                                       00026120
      XB(LVL)=B                                                         00026130
      EST2(LVL)=SX*(7.0  *(FM+FB)+32.0  *(F2(LVL)+F4(LVL))+12.0  *FT)   00026140
      SUM=EST1+EST2(LVL)                                                00026150
      FTST(1)=F(F1)                                                     00026160
      FTST(2)=F(F2(LVL))                                                00026170
      FTST(3)=F(F3)                                                     00026180
      FTST(4)=F(F4(LVL))                                                00026190
      FTST(5)=F(FM)                                                     00026200
```

```
      AEST1=SX*(7.0  *(F(FA)   +FTST(5))+32.0  *(FTST(1)+FTST(3))+12.0 00026210
     X*F(FS))                                                           00026220
      AEST2(LVL)=SX*(7.0  *(FTST(5)+F(FB)   )+32.0  *(FTST(2)+FTST(4))+100026230
     X2.0*F(FT))                                                        00026240
      ABSAR=ABSAR-AEST+AEST1+AEST2(LVL)                                 00026250
      M=M+4                                                             00026260
      IF(M.GE.MF) GO TO 5                                               00026270
      GO TO (201,200,202),N                                             00026280
  200 DELTA=ABSAR                                                       00026290
      GO TO 205                                                         00026300
  210 WRITE(6,39)                                                       00026310
   39 FORMAT(' CANC4- ERROR RETURN-N.LE.0')                             00026320
      GO TO 999                                                         00026330
  211 WRITE(6,40)                                                       00026340
   40 FORMAT(' CANC4- ERROR RETURN-N.GT.5')                             00026350
      GO TO 999                                                         00026360
  201 DELTA=(1.0,1.0)                                                   00026370
      GO TO 205                                                         00026380
  202 DO 203 I=1,4                                                      00026390
      IF(FMX(1).GE.REAL(FTST(I)))GO TO 2031                             00026400
      FMX(1)=REAL(FTST(I))                                              00026410
 2031 IF(FMX(2).GE.AIMAG(FTST(I)))GO TO 203                             00026420
      FMX(2)=AIMAG(FTST(I))                                             00026430
  203 CONTINUE                                                          00026440
      DELTA=FMAX                                                        00026450
  205 DAFT=EST-SUM                                                      00026460
      DIFF=F(DAFT)                                                      00026470
      DAFT=DAFT/63.0                                                    00026480
      Z=DIFF-EPS*DELTA                                                  00026490
      IF(REAL(Z).LE.0.0.AND.AIMAG(Z).LE.0.0) GO TO 6                    00026500
    3 IF(LVL-30)4,2,2                                                   00026510
    6 IF(LVL-1)2,4,2                                                    00026520
C     2=UP                                                              00026530
    2 A=B                                                               00026540
      ESUM=ESUM+DAFT                                                    00026550
      TSUM=TSUM+SUM                                                     00026560
    9 LVL=LVL-1                                                         00026570
      L=NRTR(LVL)                                                       00026580
      GO TO (11,12),L                                                   00026590
C     11=R1,12=R2                                                       00026600
    4 NRTR(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
      EPS=EPS/2.0                                                       00026690
    7 LVL=LVL+1                                                         00026700
      GO TO 1                                                           00026710
   11 NRTR(LVL)=2                                                       00026720
```

```
      FA=FB
      FS=F2(LVL)
      FM=FTP(LVL)
      FT=F4(LVL)
      FB=FBP(LVL)
      B=XB(LVL)
      EST=EST2(LVL)
      AEST=AEST2(LVL)
      GO TO 7
   12 EPS=2.0  *EPS
      IF(LVL-1)5,5,9
    5 CANC4=TSUM-ESUM
      GO TO 999
400   CANC4=CQSUBA(A1,B1,EP,M,ICK,ESUM,FUN,MF)
      GO TO 999
500   CANC4=CQSUB(A1,B1,EP,M,ICK,ESUM,FUN,MF)
999      RETURN
      END

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

```
C          ON EXIT.
C MEV      MAX.ALLOWABLE EVALUATIONS BEFORE ACCEPTING RESULT
C            WITH NPTS>=MEV...
C ABSCISSAE AND WEIGHTS OF QUADRATURE RULES ARE STACKED IN
C ARRAY *P* IN THE ORDER IN WHICH THEY ARE NEEDED.
      DATA
     * P( 1),P( 2),P( 3),P( 4),P( 5),P( 6),P( 7),
     * P( 8),P( 9),P(10),P(11),P(12),P(13),P(14),
     * P(15),P(16),P(17),P(18),P(19),P(20),P(21),
     * P(22),P(23),P(24),P(25),P(26),P(27),P(28)/
     *  0.7745966692414833770 4E 00,0.5555555555555555556E 00,
     *  0.8888888888888888889E 00,0.2684880898683334407 3E 00,
     *  0.9604912687080202834 2E 00,0.1046562260264672651 9E 00,
     *  0.4342437493468025580 0E 00,0.4013974147759622229 1E 00,
     *  0.4509165386584741423 5E 00,0.1344152552437842203 6E 00,
     *  0.5160328299707973969 7E-01,0.2006285293769890210 3E 00,
     *  0.9938319632127550222 1E 00,0.1700171962994026033 9E-01,
     *  0.8884592328722569988 9E 00,0.9292719531512453768 6E-01,
     *  0.6211029467372264029 4E 00,0.1715119091363913807 9E 00,
     *  0.2233866864289668816 3E 00,0.2191568584015874964 0E 00,
     *  0.2255104997982066873 9E 00,0.6720775429599070354 0E-01,
     *  0.2580759809617665356 5E-01,0.1003142786117955787 7E 00,
     *  0.8434565739321106246 3E-02,0.4646289326175798654 1E-01,
     *  0.8575592004999035115 4E-01,0.1095784210559246382 4E 00/
      DATA
     * P(29),P(30),P(31),P(32),P(33),P(34),P(35),
     * P(36),P(37),P(38),P(39),P(40),P(41),P(42),
     * P(43),P(44),P(45),P(46),P(47),P(48),P(49),
     * P(50),P(51),P(52),P(53),P(54),P(55),P(56)/
     *  0.9990981249676675976 6E 00,0.2544780791561874415 4E-02,
     *  0.9815311495537401068 7E 00,0.1644604985438781093 4E-01,
     *  0.9296548574297400566 7E 00,0.3595710330712932209 7E-01,
     *  0.8367259381688687355 0E 00,0.5697950949412335741 2E-01,
     *  0.7024962064915270786 1E 00,0.7687962049900353104 3E-01,
     *  0.5313197436443756239 7E 00,0.9362710998126447361 7E-01,
     *  0.3311353932579768330 9E 00,0.1056698935802348097 4E 00,
     *  0.1124889431331866257 5E 00,0.1119568730209534568 8E 00,
     *  0.1127552567207686916 1E 00,0.3360387714820773054 2E-01,
     *  0.1290380010035126562 6E-01,0.5015713930589953741 4E-01,
     *  0.4217630441558854839 1E-02,0.2323144663991026944 3E-01,
     *  0.4287796002500773449 3E-01,0.5478921052796286503 2E-01,
     *  0.1265156556230068011 4E-02,0.8223007957235929669 3E-02,
     *  0.1797855156812827033 3E-01,0.2848975474583354861 3E-01/
      DATA
     * P(57),P(58),P(59),P(60),P(61),P(62),P(63),
     * P(64),P(65),P(66),P(67),P(68),P(69),P(70),
     * P(71),P(72),P(73),P(74),P(75),P(76),P(77),
     * P(78),P(79),P(80),P(81),P(82),P(83),P(84)/
     *  0.3843981024945553203 9E-01,0.4681355499062801240 3E-01,
     *  0.5283494679011651986 2E-01,0.5597843651047631940 8E-01,
     *  0.9998728881203576119 4E 00,0.3632214818455306596 9E-03,
     *  0.9972062593722219590 8E 00,0.2579049794685688272 4E-02,
```

```
*  0.988684757547429479994E 00,0.611550682211724633397E-02,
```

```
*  0.98868475754742947994E 00,0.61155068221172463397E-02,
*  0.97218287474858179658E 00,0.10498246909621321898E-01,
*  0.94634285837340290515E 00,0.15406750466559497802E-01,
*  0.91037115695700429250E 00,0.20594233915912711149E-01,
*  0.86390793819369047715E 00,0.25869679327214746911E-01,
*  0.80694053195021761186E 00,0.31073551111687964880E-01,
*  0.73975604435269475868E 00,0.36064432780782572640E-01,
*  0.66290966002478059546E 00,0.40715510116944318934E-01,
*  0.57719571005204581484E 00,0.44914531653632197414E-01,
*  0.48361802694584102756E 00,0.48564330406673198716E-01/
 DATA
* P( 85),P( 86),P( 87),P( 88),P( 89),P( 90),P( 91),
* P( 92),P( 93),P( 94),P( 95),P( 96),P( 97),P( 98),
* P( 99),P(100),P(101),P(102),P(103),P(104),P(105),
* P(106),P(107),P(108),P(109),P(110),P(111),P(112)/
*  0.38335932419873034692E 00,0.51583253952048458777E-01,
*  0.27774982202182431507E 00,0.53905499335266063927E-01,
*  0.16823525155220746498E 00,0.55481404356559363988E-01,
*  0.56344313046592789972E-01,0.56277699831254301273E-01,
*  0.56377628360384717388E-01,0.16801938574103865271E-01,
*  0.64519000501757369228E-02,0.25078569652949768707E-01,
*  0.21088152457266328793E-02,0.11615723319955134727E-01,
*  0.21438980012503867246E-01,0.27394605263981432516E-01,
*  0.63260731936263354422E-03,0.41115039786546930472E-02,
*  0.89892757840641357233E-02,0.14244877372916774306E-01,
*  0.19219905124727766019E-01,0.23406777495314006201E-01,
*  0.26417473395058259931E-01,0.27989218255238159704E-01,
*  0.18073956444538835782E-03,0.12895240826104173921E-02,
*  0.30577534101755311361E-02,0.52491234548088591251E-02/
 DATA
* P(113),P(114),P(115),P(116),P(117),P(118),P(119),
* P(120),P(121),P(122),P(123),P(124),P(125),P(126),
* P(127),P(128),P(129),P(130),P(131),P(132),P(133),
* P(134),P(135),P(136),P(137),P(138),P(139),P(140)/
*  0.77033752332797418482E-02,0.10297116957956355524E-01,
*  0.12934839663607373455E-01,0.15536775555843982440E-01,
*  0.18032216390391286320E-01,0.20357755058472159467E-01,
*  0.22457265826816098707E-01,0.24282165203336599358E-01,
*  0.25791626976024229388E-01,0.26952749667633031963E-01,
*  0.27740702178279681994E-01,0.28138849915627150636E-01,
*  0.99998243035489159858E 00,0.50536095207862517625E-04,
*  0.99959879967191068325E 00,0.37774664632698466027E-03,
*  0.99831663531840739253E 00,0.93836984854238150079E-03,
*  0.99572410469840718851E 00,0.16811428654214699063E-02,
*  0.99149572117810613240E 00,0.25687649437940203731E-02,
*  0.98537149959852037111E 00,0.35728927835172996494E-02,
*  0.97714151463970571416E 00,0.46710503721143217474E-02,
*  0.96663785155841656709E 00,0.58434498758356395076E-02/
 DATA
* P(141),P(142),P(143),P(144),P(145),P(146),P(147),
* P(148),P(149),P(150),P(151),P(152),P(153),P(154),
* P(155),P(156),P(157),P(158),P(159),P(160),P(161),
```

```
     * P(162),P(163),P(164),P(165),P(166),P(167),P(168)/
     *  0.95373000642576113641E 00,0.70724899954335554680E-02,
     *  0.93832039777959288365E 00,0.83428387539681577056E-02,
     *  0.92034002547001242073E 00,0.96411777297025366953E-02,
     *  0.89974489977694003664E 00,0.10955733387837901648E-01,
     *  0.87651341448470526974E 00,0.12275830560082770087E-01,
     *  0.85064449476835027976E 00,0.13591571009765546790E-01,
     *  0.82215625436498040737E 00,0.14893641664815182035E-01,
     *  0.79108493379984836143E 00,0.16173218729577719942E-01,
     *  0.75748396638051363793E 00,0.17421930159464173747E-01,
     *  0.72142308537009891548E 00,0.18631848256138790186E-01,
     *  0.68298743109107922809E 00,0.19795495048097499488E-01,
     *  0.64227664250975951377E 00,0.20905851445812023852E-01,
     *  0.59940393024224289297E 00,0.21956366305317824939E-01,
     *  0.55449513263193254887E 00,0.22940964229387748761E-01/
      DATA
     * P(169),P(170),P(171),P(172),P(173),P(174),P(175),
     * P(176),P(177),P(178),P(179),P(180),P(181),P(182),
     * P(183),P(184),P(185),P(186),P(187),P(188),P(189),
     * P(190),P(191),P(192),P(193),P(194),P(195),P(196)/
     *  0.50768775753371660215E 00,0.23854052106038540080E-01,
     *  0.45913001198983233287E 00,0.24690524744487676909E-01,
     *  0.40897982122988867241E 00,0.25445769965464765813E-01,
     *  0.35740383783153215238E 00,0.26115673376706097680E-01,
     *  0.30457644155671404334E 00,0.26696622927450359906E-01,
     *  0.25067873030348317661E 00,0.27185513229624791819E-01,
     *  0.19589750271110015392E 00,0.27579749566481873035E-01,
     *  0.14042423315256017459E 00,0.27877251476613701609E-01,
     *  0.84454040083710883710E-01,0.28076455793817246607E-01,
     *  0.28184648949745694339E-01,0.28176319033016602131E-01,
     *  0.28188814180192358694E-01,0.84009692870519326354E-02,
     *  0.32259500250878684614E-02,0.12539284826474884353E-01,
     *  0.10544076228633167722E-02,0.58078616599775673635E-02,
     *  0.10719490006251933623E-01,0.13697302631990716258E-01/
      DATA
     * P(197),P(198),P(199),P(200),P(201),P(202),P(203),
     * P(204),P(205),P(206),P(207),P(208),P(209),P(210),
     * P(211),P(212),P(213),P(214),P(215),P(216),P(217),
     * P(218),P(219),P(220),P(221),P(222),P(223),P(224)/
     *  0.31630366082226447689E-03,0.20557519893273465236E-02,
     *  0.44946378920320678616E-02,0.71224386864583871532E-02,
     *  0.96099525623638830097E-02,0.11703388747657003101E-01,
     *  0.13208736697529129966E-01,0.13994609127619079852E-01,
     *  0.90372734658751149261E-04,0.64476204130572477933E-03,
     *  0.15288767050877655684E-02,0.26245617274044295626E-02,
     *  0.38516876166398709241E-02,0.51485584789781777618E-02,
     *  0.64674198318036867274E-02,0.77683877779219912200E-02,
     *  0.90161081951956431600E-02,0.10178877529236079733E-01,
     *  0.11228632913408049354E-01,0.12141082601668299679E-01,
     *  0.12895813488012114694E-01,0.13476374833816515982E-01,
     *  0.13870351089139840997E-01,0.14069424957813575318E-01,
     *  0.25157870384280661489E-04,0.18887326450650491366E-03,
```

```
*  0.46918492424785040975E-03,0.84057143271072246365E-03/       00028800
 DATA                                                             00028810
* P(225),P(226),P(227),P(228),P(229),P(230),P(231),               00028820
* P(232),P(233),P(234),P(235),P(236),P(237),P(238),               00028830
* P(239),P(240),P(241),P(242),P(243),P(244),P(245),               00028840
* P(246),P(247),P(248),P(249),P(250),P(251),P(252)/               00028850
*  0.12843824718970101768E-02,0.17864463917586498247E-02,         00028860
*  0.23355251860571608737E-02,0.29217249379178197538E-02,         00028870
*  0.35362449977167777340E-02,0.41714193769840788528E-02,         00028880
*  0.48205888648512683476E-02,0.54778666939189508240E-02,         00028890
*  0.61379152800413850435E-02,0.67957855048827733948E-02,         00028900
*  0.74468208324075910174E-02,0.80866093647888599710E-02,         00028910
*  0.87109650797320868736E-02,0.93159241280693950932E-02,         00028920
*  0.98977475240487497440E-02,0.10452925722906011926E-01,         00028930
*  0.10978183152658912470E-01,0.11470482114693874380E-01,         00028940
*  0.11927026053019270040E-01,0.12345262372243838455E-01,         00028950
*  0.12722884982732382906E-01,0.13057836688353048840E-01,         00028960
*  0.13348311463725179953E-01,0.13592756614812395910E-01,         00028970
*  0.13789874783240936517E-01,0.13938625738306850804E-01,         00028980
*  0.14038227896908623303E-01,0.14088159516508301065E-01/         00028990
 DATA                                                             00029000
* P(253),P(254),P(255),P(256),P(257),P(258),P(259),               00029010
* P(260),P(261),P(262),P(263),P(264),P(265),P(266),               00029020
* P(267),P(268),P(269),P(270),P(271),P(272),P(273),               00029030
* P(274),P(275),P(276),P(277),P(278),P(279),P(280)/               00029040
*  0.99999759637974846462E 00,0.69379364324108267170E-05,         00029050
*  0.99994399620705437576E 00,0.53275293669780613125E-04,         00029060
*  0.99976049092443204733E 00,0.13575491094922871973E-03,         00029070
*  0.99938033802502358193E 00,0.24921240048299729402E-03,         00029080
*  0.99874561446809511470E 00,0.38974528447328229322E-03,         00029090
*  0.99780535449595727456E 00,0.55429531493037471492E-03,         00029100
*  0.99651414591489027385E 00,0.74028280424450333046E-03,         00029110
*  0.99483150280062100052E 00,0.94536151685852538246E-03,         00029120
*  0.99272134428278861533E 00,0.11674841174299594077E-02,         00029130
*  0.99015137040077015918E 00,0.14049079956551446427E-02,         00029140
*  0.98709252795403406719E 00,0.16561127281544526052E-02,         00029150
*  0.98351865757863272876E 00,0.19197129710138724125E-02,         00029160
*  0.97940628167086268381E 00,0.21944069253638388388E-02,         00029170
*  0.97473445975240266776E 00,0.24789582266575679307E-02/         00029180
 DATA                                                             00029190
* P(281),P(282),P(283),P(284),P(285),P(286),P(287),               00029200
* P(288),P(289),P(290),P(291),P(292),P(293),P(294),               00029210
* P(295),P(296),P(297),P(298),P(299),P(300),P(301),               00029220
* P(302),P(303),P(304),P(305),P(306),P(307),P(308)/               00029230
*  0.96948465950245923177E 00,0.27721957645934509940E-02,         00029240
*  0.96364062156981213252E 00,0.30730184347025783234E-02,         00029250
*  0.95718821610986096274E 00,0.33803979910869203823E-02,         00029260
*  0.95011529752129487656E 00,0.36933779170256508183E-02,         00029270
*  0.94241156519108305981E 00,0.40110687240750233989E-02,         00029280
*  0.93406843615772578800E 00,0.43326409680929828545E-02,         00029290
*  0.92507893290707565236E 00,0.46573172997568547773E-02,         00029300
*  0.91543758715576504064E 00,0.49843645647655386012E-02,         00029310
```

```
*  0.905140358813261595l9E 00,0.531308660518705656 63E-02,
```

```
     * 0.98482396598119202090E-01,0.14017968039456608810E-01,
     * 0.70406976042855179063E-01,0.14055382072649964277E-01,
     * 0.42269164765363603212E-01,0.14080351962553661325E-01,
     * 0.14093886410782462614E-01,0.14092845069160408355E-01,
     * 0.14094407090096179347E-01/
      ICHECK = 0
C CHECK FOR TRIVIAL CASE.
      IF (A.EQ.B) GO TO 70
C SCALE FACTORS.
      SUM = (B+A)/2.0
      DIFF = (B-A)/2.0
C 1-POINT GAUSS
      FZERO = F(SUM)
      RESULT(1) = 2.0*FZERO*DIFF
      I = 0
      IOLD = 0
      INEW = 1
      K = 2
      ACUM = (0.0,0.0)
      GO TO 30
   10 IF (K.EQ.8) GO TO 50
      IF(INEW+IOLD.GE.MEV) GO TO 60
      K = K + 1
      ACUM = (0.0,0.0)
C CONTRIBUTION FROM FUNCTION VALUES ALREADY COMPUTED.
      DO 20 J=1,IOLD
        I = I + 1
        ACUM = ACUM + P(I)*FUNCT(J)
   20 CONTINUE
C CONTRIBUTION FROM NEW FUNCTION VALUES.
   30 IOLD = IOLD + INEW
      DO 40 J=INEW,IOLD
        I = I + 1
        X = P(I)*DIFF
        FUNCT(J) = F(SUM+X) + F(SUM-X)
        I = I + 1
        ACUM = ACUM + P(I)*FUNCT(J)
   40 CONTINUE
      INEW = IOLD + 1
      I = I + 1
      RESULT(K) = (ACUM+P(I)*FZERO)*DIFF
C CHECK FOR CONVERGENCE.
      IF(ABS(REAL(RESULT(K))-REAL(RESULT(K-1))).LE.EPSIL*
     $ABS(REAL(RESULT(K))).AND.
     $ ABS(AIMAG(RESULT(K))-AIMAG(RESULT(K-1))).LE.EPSIL*
     $ABS(AIMAG(RESULT(K)))) GO TO 60
      GO TO 10
C CONVERGENCE NOT ACHIEVED.
   50 ICHECK = 1
C NORMAL TERMINATION.
   60 NPTS = INEW + IOLD
      RETURN
```

```
C TRIVIAL CASE
   70 K = 2
      RESULT(1) = (0.0,0.0)
      RESULT(2) = (0.0,0.0)
      NPTS = 0
      RETURN
      END

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

```
C THE TOTAL INTEGRATION WOULD BE ALMOST ENTIRELY DUE TO THE
C DELINQUENT PANEL.  WITHOUT THIS CONDITION THE COMPUTATION
C MIGHT CONTINUE DESPITE THE REQUESTED RELATIVE ERROR BEING
C ACHIEVED.
C    THE OUTCOME OF THE INTEGRATION IS INDICATED BY ICHECK.
C    ICHECK=0  -  CONVERGENCE OBTAINED WITHOUT INVOKING
C                 SUBDIVISION.  THIS CORRESPONDS TO THE
C                 DIRECT USE OF QUAD.
C    ICHECK=1  -  RESULT OBTAINED AFTER INVOKING SUBDIVISION.
C    ICHECK=2  -  AS FOR ICHECK=1 BUT AT SOME POINT THE
C                 RELAXED CONVERGENCE CRITERION WAS USED.
C                 THE RISK OF UNDERESTIMATING THE RELATIVE
C                 ERROR WILL BE INCREASED.  IF NECESSARY,
C                 CONFIDENCE MAY BE RESTORED BY CHECKING
C                 EPSIL AND RELERR FOR A SERIOUS DISCREPANCY.
C    ICHECK NEGATIVE
C                 IF DURING THE SUBDIVISION PROCESS THE
C                 ALLOWED UPPER LIMIT ON THE NUMBER OF PANELS
C                 THAT MAY BE GENERATED (PRESENTLY 4096) IS
C                 REACHED A RESULT IS OBTAINED WHICH MAY BE
C                 UNRELIABLE BY CONTINUING THE INTEGRATION
C                 WITHOUT FURTHER SUBDIVISION IGNORING
C                 CONVERGENCE FAILURES. THIS OCCURRENCE IS
C                 FLAGGED BY RETURNING ICHECK WITH NEGATIVE
C                 SIGN.
C   THE RELIABILITY OF THE ALGORITHM WILL DECREASE FOR LARGE
C   VALUES OF EPSIL.  IT IS RECOMMENDED THAT EPSIL SHOULD
C   GENERALLY BE LESS THAN ABOUT 0.001.
      DIMENSION RESULT(8)
      INTEGER BAD, OUT
      LOGICAL RHS
      EXTERNAL F
      DATA NMAX/4096/
      CALL CQUAD(A, B, RESULT, K, EPSIL, NPTS, ICHECK, F,MEV)
      CQSUB = RESULT(K)
      RELERR = (0.0,0.0)
      IF(REAL(CQSUB).NE.0.0.AND.AIMAG(CQSUB).NE.0.0) RELERR=
     $ CMPLX(ABS(REAL(RESULT(K)-RESULT(K-1)))/REAL(CQSUB),
     $ ABS(AIMAG(RESULT(K)-RESULT(K-1)))/AIMAG(CQSUB))
C CHECK IF SUBDIVISION IS NEEDED.
      IF (ICHECK.EQ.0) RETURN
C SUBDIVIDE
      ESTIM=CQSUB*EPSIL
      ESTIM=CMPLX(ABS(REAL(ESTIM)),ABS(AIMAG(ESTIM)))
      IC = 1
      RHS = .FALSE.
      N = 1
      H = B - A
      BAD = 1
   10 CQSUB = (0.0,0.0)
      RELERR = (0.0,0.0)
      H = H*0.5
```

```
      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(REAL(CQSUB)),
     $ AIMAG(RELERR)/ABS(AIMAG(CQSUB)))
      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 TROUBLESOME 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 = -IABS(IC)
   70   CQSUB = CQSUB + RESULT(M)
   80 CONTINUE
      RELERR = RELERR + COMP
```

```
   90 GO TO (20,30,40), OUT
C RELAXED CONVERGENCE
  100 IC = ISIGN(2,IC)
      GO TO 70
      END

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

```
C ACHIEVED.
C   THE OUTCOME OF THE INTEGRATION IS INDICATED BY ICHECK.
C   ICHECK=0  -  CONVERGENCE OBTAINED WITHOUT INVOKING SUB-
C                DIVISION.  THIS WOULD CORRESPOND TO THE
C                DIRECT USE OF QUAD.
C   ICHECK=1  -  RESULT OBTAINED AFTER INVOKING SUBDIVISION.
C   ICHECK=2  -  AS FOR ICHECK=1 BUT AT SOME POINT THE
C                RELAXED CONVERGENCE CRITERION WAS USED.
C                THE RISK OF UNDERESTIMATING THE RELATIVE
C                ERROR WILL BE INCREASED.  IF NECESSARY,
C                CONFIDENCE MAY BE RESTORED BY CHECKING
C                EPSIL AND RELERR FOR A SERIOUS DISCREPANCY.
C   ICHECK NEGATIVE
C                IF DURING THE SUBDIVISION PROCESS THE STACK
C                OF DELINQUENT INTERVALS BECOMES FULL (IT IS
C                PRESENTLY SET TO HOLD AT MOST 100 NUMBERS)
C                A RESULT IS OBTAINED BY CONTINUING THE
C                INTEGRATION IGNORING CONVERGENCE FAILURES
C                WHICH CANNOT BE ACCOMMODATED ON THE STACK.
C                THIS OCCURRENCE IS FLAGGED BY RETURNING
C                ICHECK WITH NEGATIVE SIGN.
C  THE RELIABILITY OF THE ALGORITHM WILL DECREASE FOR LARGE
C  VALUES OF EPSIL.  IT IS RECOMMENDED THAT EPSIL SHOULD
C  GENERALLY BE LESS THAN ABOUT 0.001.
      DIMENSION RESULT(8), STACK(100)
      EXTERNAL F
      DATA ISMAX/100/
      CALL CQUAD(A, B, RESULT, K, EPSIL, NPTS, ICHECK, F,MEV)
      CQSUBA = RESULT(K)
      RELERR = (0.0,0.0)
      IF(REAL(CQSUBA).NE.0.0.AND.AIMAG(CQSUBA).NE.0.0) RELERR=
     $ CMPLX(ABS(REAL(RESULT(K)-RESULT(K-1)))/REAL(CQSUBA),
     $ ABS(AIMAG(RESULT(K)-RESULT(K-1)))/AIMAG(CQSUBA))
C CHECK IF SUBDIVISION IS NEEDED
      IF (ICHECK.EQ.0) RETURN
C SUBDIVIDE
      ESTIM=CQSUBA*EPSIL
      ESTIM=CMPLX(ABS(REAL(ESTIM)),ABS(AIMAG(ESTIM)))
      RELERR = (0.0,0.0)
      CQSUBA = (0.0,0.0)
      IS = 1
      IC = 1
      SUB1 = A
      SUB3 = B
   10 SUB2 = (SUB1+SUB3)*0.5
      CALL  CQUAD(SUB1, SUB2, RESULT, K, EPSIL, NF, ICHECK, F,MEV)
      NPTS = NPTS + NF
      IF(NPTS.GE.MEV) GO TO 50
      COMP = (RESULT(K)-RESULT(K-1))
      COMP=CMPLX(ABS(REAL(COMP)),ABS(AIMAG(COMP)))
      IF (ICHECK.EQ.0) GO TO 30
      IF(REAL(COMP).LE.REAL(ESTIM).AND.
```

```
     $ AIMAG(COMP).LE.AIMAG(ESTIM)) GO TO 70
      IF (IS.GE.ISMAX) GO TO 20
C STACK SUBINTERVAL (SUB1,SUB2) FOR FUTURE EXAMINATION
      STACK(IS) = SUB1
      IS = IS + 1
      STACK(IS) = SUB2
      IS = IS + 1
      GO TO 40
   20 IC = -IABS(IC)
   30 CQSUBA = CQSUBA + RESULT(K)
      RELERR = RELERR + COMP
   40 CALL CQUAD(SUB2, SUB3, RESULT, K, EPSIL, NF, ICHECK, F,MEV)
      NPTS = NPTS + NF
      IF(NPTS.GE.MEV) GO TO 50
      COMP = (RESULT(K)-RESULT(K-1))
      COMP=CMPLX(ABS(REAL(COMP)),ABS(AIMAG(COMP)))
      IF (ICHECK.EQ.0) GO TO 50
      IF(REAL(COMP).LE.REAL(ESTIM).AND.
     $ AIMAG(COMP).LE.AIMAG(ESTIM)) GO TO 80
C SUBDIVIDE INTERVAL (SUB2,SUB3)
      SUB1 = SUB2
      GO TO 10
   50 CQSUBA = CQSUBA + RESULT(K)
      RELERR = RELERR + COMP
      IF(NPTS.GE.MEV) RETURN
      IF (IS.EQ.1) GO TO 60
C SUBDIVIDE THE DELINQUENT INTERVAL LAST STACKED
      IS = IS - 1
      SUB3 = STACK(IS)
      IS = IS - 1
      SUB1 = STACK(IS)
      GO TO 10
C SUBDIVISION RESULT
   60 ICHECK = IC
      RELERR=CMPLX(REAL(RELERR)/ABS(REAL(CQSUBA)),
     $ AIMAG(RELERR)/ABS(AIMAG(CQSUBA)))
      RETURN
C RELAXED CONVERGENCE
   70 IC = ISIGN(2,IC)
      GO TO 30
   80 IC = ISIGN(2,IC)
      GO TO 50
      END

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

```
C  'BLACK-BOX' NATURE OF IMSL ROUTINE 'ZXSSQ' (SEE IMSL DOC.).
C  THUS, ONE SHOULD RENAME ACTUAL NAME TO 'FCODE' FOR USE HERE.
C  (I.E., SEE CALL FCODE IN 'FPXSSQ'---EXTERNAL FUNCTION FOR ZXSSQ).
C
C--THE USER MUST DECLARE THE CALLING PARAMETERS
C  SUBZ,SUBEND (ANY DESIRED NAMES MAY BE USED) AS EXTERNAL IN
C  MAIN CALLING PROGRAM; E.G.,
C
C       EXTERNAL SUBZ,SUBEND
C       CALL IMSLMQ(SUBZ,SUBEND)
C       STOP
C       END
C
C--THIS INTERFACE BETWEEN 'MARQRT' AND 'IMSLMQ' WAS WRITTEN BY
C  W.L.ANDERSON, U.S. GEOLOGICAL SURVEY, DENVER, COLORADO.
C
C--SEE DOCUMENTATION OF 'MARQLOOPS', USGS OPEN-FILE REPT 79-240 (1979),
C  FOR DETAILS ON CODING THE REQUIRED SUBROUTINES FCODE,SUBZ, AND
C  SUBEND. ALSO SEE IMSL DOCUMENTATION FOR 'ZXSSQ'.
C
C--THE INPUT ORDER ON FILE05 (PARAMETER FILE) IS:
C
C  1.  TITLE  (MAX. 80-CHARACTER TITLE---ALWAYS REQUIRED).
C  2.  $PARMS (SAME PARMS DEFINITIONS FOR PGM MARQRT FOR PARAMETERS:
C          N,K,IP,M,IALT,ISTOP,IWT,NITER,E,SCALEP,B(),IPRT, AND IB()).
C          PLUS, ADDITIONAL PARMS FOR 'ZXSSQ' (SEE IMSL DOC):
C          IOPT,NSIG,MAXFN,EPS,DELTA,PARM(4).
C  3.  (OBJECT-TIME FORMAT STATEMENT) FOR READING THE DATA MATRIX
C        (Y(I),X(I,J),J=1,M*) ON FILE IALT (DEFAULT 10), WHERE M*=M+IWT.
C  (3A.  INSERT DATA MATRIX HERE ONLY IF IALT=5)
C  4.  $INIT OPTIONAL NAMELIST FOR READING ADDITIONAL PARMS IN
C               SUBROUTINE SUBZ  (WHICH MAY BE A DUMMY SUBROUTINE).
C  5.  OPTIONALLY, REPEAT STEPS 1-4, IF ISTOP=0 WAS USED IN STEP 2.
C
C--OUTPUT IS GIVEN ON FILE06 (ON-LINE USUALLY), AND
C  ON FILE16 (CONTAINS ALL PRINTABLE OUTPUT); FILE06 CONTAINS ONLY
C  OUTPUT VIA PARM IPRT (0--ABBREVIATED, 1 OR -2 --DETAIL).
C
C-----------------------------------------------------------------------
C--THE USER SHOULD ADD >IML>IMSL TO THE SEARCH RULES ON MULTICS.
C  (IMSL ROUTINES USED: ZXSSQ,LEQT1P,LUDECP,LUELMP,UERTST,LINV1P)
C-----------------------------------------------------------------------
C
      DIMENSION B(20),GRAD(20),TITLE(16),H(270),COV(20,20),SE(20)
      DIMENSION SQWT(200),IB(20),PRNT(5),C(20),INDEX(20),
     & XJAC(200,20),XJTJ(210),WORK(710),PARM(4),F(200)
      INTEGER SP,SCALEP,SY,SCALEY
C--THE FOLLOWING CHARACTER STATEMENTS ONLY FOR HONEYWELL MULTICS:
      CHARACTER*4 FMT(18)
      CHARACTER*5 TITLE
      COMMON/FIXDAT/Y(200),X(200,5),BFIX(20),YMAX,IIB(20),IIP,NOBS,K
      COMMON/PRT/IPRT
```

```
      EXTERNAL FPXSSQ,LNXSSQ
      EQUIVALENCE (SQWT(1),X(1,5)),(SP,SCALEP),(N,NOBS),
    & (M,NVARS),(NITER,LIMIT),(SY,SCALEY),(E,EPS),(SS,SSQ),
    & (IB(1),IIB(1)),(IP,IIP),(B(1),BFIX(1))
      NAMELIST/PARMS/N,K,IP,M,IALT,NITER,IB,E,B,IWT,ISTOP,SP,SY,
    & SCALEP,SCALEY, IDER,IPRT,INON,FF,T,TAU,XL,MODLAM,GAMCR,
    & DEL,ZETA,IOUT,IOPT,NSIG,MAXFN,EPS,DELTA,PARM
C-- NOTE NAMELIST PARMS INCLUDED (BUT IGNORED) FOR COMPATIBILITY
C   ARE: IDER,INON,FF,T,TAU,XL,MODLAM,GAMCR,DEL,ZETA,IOUT.
C   ALSO, SP=SCALEP WILL BE CONSIDERED MODE=1 (ALOG OPTION) ONLY
C   WHEN SP=2 OR 1 (AND MODE=0 LINEAR WHEN SP=0).  SY=SCALEY IS
C   IGNORED FOR THIS VERSION OF 'IMSLMQ'.
C
C--READ IMSLMQ TITLE LINE
      READ(5,4) TITLE
4     FORMAT(16A5)
C--PRESET DEFAULTS
      N=0
      K=0
      M=0
      IP=0
      IPRT=0
      ISTOP=1
      IWT=0
      IALT=10
      NITER=10
      SP=0
      MODE=0
      FMIN=0.0
      IOPT=1
      NSIG=3
      MAXFN=0
      EPS=0.0
      DELTA=0.0
      PARM(1)=.01
      PARM(2)=2.
      PARM(3)=120.
      PARM(4)=.1
      DO 5 I=1,20
      IB(I)=0
      B(I)=0.0
5     GRAD(I)=0.0
C--READ $PARMS
6     READ(5,PARMS)
C--TEST $PARMS BEFORE PROCEEDING
      IF(N.GT.200.OR.K.GT.20.OR.M.GT.4.OR.IWT.GT.1.OR.IP.GT.19.OR.
    & N.LT.1.OR.K.LT.1.OR.M.LT.1.OR.IWT.LT.0.OR.IP.LT.0.OR.
    & N.LT.K-IP.OR.IALT.EQ.6.OR.IALT.EQ.16)
    &CALL ERRMSG('SOME $PARMS OUT OF RANGE.',5,6,16)
      DO 7 I=1,K
      IF(B(I).EQ.0.0)CALL ERRMSG('SOME B(I)=0.0   ',3,6,16)
7     CONTINUE
```

```
      IF(MAXFN.EQ.0) MAXFN=2*K*NITER
      IF(IP.LE.0) GO TO 9
      DO 8 I=1,IP
      IF(IB(I).GT.0) GO TO 8
      CALL ERRMSG('IP.GT.1 BUT SOME IB(I).LE.0    ',6,6,16)
8     CONTINUE
9     IF(SP.NE.0) MODE=1
      IF(IPRT.EQ.1) IPRT=-2
C--READ OBJECT FORMAT FOR DATA MATRIX ON FILE IALT.
      READ(5,10) (FMT(I),I=1,18)
10    FORMAT(18A4)
      M1=M+IWT
      YMAX=0.0
      DO 11 I=1,N
      READ(IALT,FMT) Y(I),(X(I,J),J=1,M1)
      SQWT(I)=1.0
      IF(IWT.EQ.1.AND.X(I,M1).NE.0.0) SQWT(I)=1.0/X(I,M1)
      IF(ABS(Y(I)).GT.YMAX) YMAX=ABS(Y(I))
11    CONTINUE
      IF(IALT.NE.5) REWIND IALT
C--INITIALIZATION VIA CALL SUBZ (READ $INIT, TEST B,X,Y, ETC)
      CALL SUBZ(Y,X,B,PRNT,NDUM,N,TITLE,1)
C--WRITE $PARMS ON UNIT 6 AND 16
      WRITE(6,60) TITLE,N,K,IP,M,E,IALT,ISTOP,IWT,NITER,SP,IPRT,
     & IOPT,NSIG,MAXFN,EPS,DELTA,PARM
60    FORMAT('1I M S L M Q  --        ',16A5//' N=',I5,9X,'K=',I4,10X,'IP='
     & ,I4,9X,'M=',I3,11X,'E=',E10.3/' IALT=',I3,8X,'ISTOP=',I2,8X,
     & 'IWT=',I2,10X,'NITER=',I6,4X,'SCALEP=',I2//' IPRT=',I3,
     & 8X,'IOPT=',I2,9X,'NSIG=',I3,8X,'MAXFN=',I6,4X,'EPS=',E10.3/
     & ' DELTA=',E10.3/' PARM=',4E10.3)
      WRITE(16,60) TITLE,N,K,IP,M,E,IALT,ISTOP,IWT,NITER,SP,IPRT,
     & IOPT,NSIG,MAXFN,EPS,DELTA,PARM
      IF(IP.EQ.0) GO TO 661
      WRITE(6,660) (IB(I),I=1,IP)
660   FORMAT(/' IB=',19I3)
      WRITE(16,660) (IB(I),I=1,IP)
661   WRITE(6,662) FMT
662   FORMAT(/' FMT=',18A4/)
      WRITE(16,662) FMT
      WRITE(6,6000) (B(I),I=1,K)
6000  FORMAT(/' INITIAL PARAMETERS'//(5E16.8))
      WRITE(16,6000) (B(I),I=1,K)
C--INTERFACE WITH ZXSSQ USING MARQRT FCODE
      DO 1 I=1,20
1     INDEX(I)=I
      KIP=K-IP
      IF(IP.EQ.0) GO TO 400
C--REORDER B TO C WHEN IP>0
      IM=0
      DO 202 I=1,K
      DO 201 J=1,IP
      IF(I.EQ.IB(J)) GO TO 202
```

```
201   CONTINUE
      IM=IM+1
      C(IM)=B(I)
      INDEX(IM)=I
202   CONTINUE
      WRITE(6,203) (I,I=1,K)
203   FORMAT(// PARAMETER INDEX:',20I3)
      WRITE(16,203) (I,I=1,K)
      WRITE(6,204) (INDEX(I),I=1,KIP)
204   FORMAT(' REORDERED AS...:',20I3)
      WRITE(16,204) (INDEX(I),I=1,KIP)
      WRITE(6,206) (C(I),I=1,KIP)
206   FORMAT(// REORDERED PARAMETERS'//(5E16.8))
      WRITE(16,206) (C(I),I=1,KIP)
      GO TO 500
400   DO 401 I=1,K
401   C(I)=B(I)
500   CONTINUE
      IF(MODE.EQ.0) GO TO 12
C--LOG PARAMETERS CHOSEN (MODE=1 OR SP.NE.0)
      DO 111 I=1,KIP
      IF(C(I).LE.0.0)CALL ERRMSG('SP.NE.0 & SOME B(I).LE.0.',5,6,16)
111   C(I)=ALOG(C(I))
      CALL ZXSSQ(LNXSSQ,N,KIP,NSIG,EPS,DELTA,MAXFN,IOPT,PARM,
     & C,SSQ,F,XJAC,200,XJTJ,WORK,INFER,IER)
      DO 1111 I=1,KIP
1111  C(I)=EXP(C(I))
      GO TO 21
C--LINEAR PARAMETERS CHOSEN (MODE=0 OR SP=0)
12    CALL ZXSSQ(FPXSSQ,N,KIP,NSIG,EPS,DELTA,MAXFN,IOPT,PARM,
     & C,SSQ,F,XJAC,200,XJTJ,WORK,INFER,IER)
C--ZXSSQ ERRMSG CODE HANDLERS
21    IF(IER.EQ.0) GO TO 100
      IF(IER.EQ.129)
     &CALL ERRMSG('SINGULARITY DETECTED IN JACOBIAN & RECOVERY FAILED',
     & 10,6,16)
      IF(IER.EQ.130)
     &CALL ERRMSG('N,K-IP,IOPT,PARM(1) OR PARM(2) INCORRECT',8,6,16)
      IF(IER.EQ.131)
     &CALL WARN('MARQUARDT PARAMETER EXCEEDED PARM(3)     ',8,6,16,$100)
      IF(IER.EQ.132) CALL ERRMSG(
     &'AFTER RECOVERY FROM SINGULAR JACOBIAN, B CYCLED BACK AGAIN..',
     & 12,6,16)
      IF(IER.EQ.133)CALL WARN('MAXFN EXCEEDED.',3,6,16,$100)
      IF(IER.EQ.38)CALL WARN('JACOBIAN=0. SOLUTION IS STATIONARY POINT',
     & 8,6,16,$100)
100   WRITE(6,603) (WORK(I),I=1,5),INFER,IER,SSQ,(C(I),I=1,KIP)
      WRITE(16,603) (WORK(I),I=1,5),INFER,IER,SSQ,(C(I),I=1,KIP)
603   FORMAT(/// $$$$  IMSLMQ CONVERGENCE INFORMATION:'//
     & ' NORM OF GRADIENT',T32,E16.8/' FUNCTION EVALUATIONS',T32,E16.8/
     & ' EST. SIGN. DIGITS',T32,E16.8/' MARQUARDT PARAMETER',T32,
     & E16.8/' NO. ITERATIONS',T32,E16.8/' TYPE CONVERGENCE (INFER)',
```

```
     & T32,I3/' ERROR CODE (IER)',T32,I5/
     & ' RESIDUAL SUM-OF-SQUARES (SSQ)=',E16.8//
     &' **** FINAL UNSCALED PARAMETERS'//
     & (5E16.8))
99    KK=MAX0((KIP+1)*KIP/2,5)
      DO 80 I=1,KIP
80    GRAD(I)=2.*WORK(KK+I)
      WRITE(6,82) (GRAD(I),I=1,KIP)
82    FORMAT(/' SCALED GRADIENT'//(5E16.8))
      WRITE(16,82) (GRAD(I),I=1,KIP)
      IF(IPRT.EQ.-2) WRITE(6,699)
699   FORMAT(/3X,'I',4X,'OBS.Y(I)',6X,'CAL',11X,'RES',8X,'X(I,1)',8X,
     & 'WT(I)')
      WRITE(16,1699)
1699  FORMAT(/3X,'I',4X,'OBS.Y(I)',6X,'CAL',11X,'RES',8X,'X(I,1)',8X,
     & 'X(I,2)',8X,'X(I,3)',8X,'X(I,4)',8X,'WT(I)')
      SUMF2=0.0
      DO 110 I=1,NOBS
      RES=F(I)/SQWT(I)
      YCAL=Y(I)-RES
      WT=SQWT(I)*SQWT(I)
      SUMF2=SUMF2+F(I)*F(I)
      IF(IPRT.EQ.-2) WRITE(6,210) I,Y(I),YCAL,RES,X(I,1),WT
210   FORMAT(1X,I3,2E14.6,E11.3,2E14.6)
      WRITE(16,211) I,Y(I),YCAL,RES,(X(I,J),J=1,4),WT
211   FORMAT(1X,I3,2E14.6,E11.3,5E14.6)
110   CONTINUE
      IF(N.EQ.KIP) RMSERR=0.0
      IF(N.GT.KIP) RMSERR=SQRT(SUMF2/(N-KIP))
      WRITE(6,604) RMSERR
604   FORMAT(/' **** RMSERR=',E16.8)
      WRITE(16,604) RMSERR
C---PRINT ON FILE16 (ONLY) THE FINAL SCALED PARTIALS (JACOBIAN)
      WRITE(16,605)
605   FORMAT(/' FINAL SCALED PARTIALS (JACOBIAN)')
      DO 112 I=1,NOBS
      WRITE(16,606) I,(XJAC(I,J),J=1,KIP)
606   FORMAT(1X,I3,5E16.8/(4X,5E16.8))
112   CONTINUE
C---GET INVERSE JACOBIAN TRANSPOSE*JACOBIAN (FROM XJTJ SYMMETRIC MATRIX)
      CALL LINV1P(XJTJ,KIP,H,5,D1,D2,IER)
      IF(IER.GT.128) CALL ERRMSG('IN LINV1P CALL.',3,6,16)
C---FINAL STATISTICS
      DO 301 I=1,KIP
      DO 301 J=1,KIP
301   COV(I,J)=H(LOC(I,J))
      IF(IPRT.EQ.-2) WRITE(6,120)
120   FORMAT(/' SCALED COVARIANCE MATRIX (INVERSE OF XJTJ)')
      WRITE(16,120)
      DO 122 I=1,KIP
      SE(I)=SQRT(ABS(COV(I,I)))
      IF(IPRT.EQ.-2) WRITE(6,300) INDEX(I),(COV(I,J),J=1,KIP)
```

```
300    FORMAT(1X,I2,10E12.4/(3X,10E12.4))
       WRITE(16,300) INDEX(I),(COV(I,J),J=1,KIP)
122    CONTINUE
       IF(IPRT.EQ.-2) WRITE(6,304)
304    FORMAT(/' CORRELATION MATRIX')
       WRITE(16,304)
       DO 131 I=1,KIP
       IF(SE(I).EQ.0.0) GO TO 132
       DO 129 J=1,KIP
       IF(SE(J).EQ.0.0) GO TO 129
       COV(I,J)=COV(I,J)/(SE(I)*SE(J))
129    CONTINUE
133    IF(IPRT.EQ.-2) WRITE(6,300) INDEX(I),(COV(I,J),J=1,KIP)
       WRITE(16,300) INDEX(I),(COV(I,J),J=1,KIP)
       GO TO 131
132    COV(I,I)=1.0
       GO TO 133
131    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(SP.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
126    CONTINUE
       DO 600 I=1,KIP
600    H(I)=C(I)
       IF(IP.EQ.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.EQ.IB(J)) GO TO 127
128    CONTINUE
       IM=IM+1
       H(I)=C(IM)
127    CONTINUE
601    CALL SUBEND(Y,X,H,K,N,TITLE,1)
       IF(ISTOP.EQ.1) GO TO 999
       READ(5,4) TITLE
       DO 1000 I=1,20
1000   B(I)=0.0
       GO TO 6
C--FOLLOWING CALL ONLY FOR HONEYWELL MULTICS SYSTEM:
999    CALL CLOSE_FILE('-ALL')
C      STOP
       RETURN
```

```
      END

      SUBROUTINE FPXSSQ(C,N,KIP,F)
C--CALCULATES RESIDUAL VECTOR F(N) FOR 'ZXSSQ' (EXTERNAL FPXSSQ) FOR
C  UNSCALED C(KIP) PARAMETER VECTOR AND DATA X(200,5),Y(200) IN FIXDAT.
C
C       C= INPUT VECTOR OF PARAMETERS (LENGTH KIP)
C       N= NO. OBS. <= 200
C     KIP= NO. PARAMETERS =K-IP (IP>=0)
C       F= OUTPUT VECTOR OF (WEIGHTED) FUNCTION RESIDUALS (LENGTH N)
C
C--CALLS 'FCODE' AS CODED FOR 'MARQRT' WITH FIXED DATA IN COMMON/FIXDAT/
C
      DIMENSION C(1),F(1),PRNT(5),SQWT(200),BIP(20)
      COMMON/FIXDAT/Y(200),X(200,5),BFIX(20),YMAX,IIB(20),IIP,NOBS,K
      EQUIVALENCE (SQWT(1),X(1,5))
      IF(IIP.GT.0) GO TO 2
      DO 1 I=1,N
      CALL FCODE(Y,X,C,PRNT,FF,I,1)
1     F(I)=SQWT(I)*(Y(I)-FF)
      RETURN
2     IM=0
      DO 4 I=1,K
      BIP(I)=BFIX(I)
      DO 3 J=1,IIP
      IF(I.EQ.IIB(J)) GO TO 4
3     CONTINUE
      IM=IM+1
      BIP(I)=C(IM)
4     CONTINUE
      DO 5 I=1,N
      CALL FCODE(Y,X,BIP,PRNT,FF,I,1)
5     F(I)=SQWT(I)*(Y(I)-FF)
      RETURN
      END

      SUBROUTINE LNXSSQ(C,N,KIP,F)
C--INTERFACES TO 'FPXSSQ' TO ALLOW PARMS TO BE IN LOG OR LINEAR
C  SPACE OUTSIDE, BUT ALWAYS LINEAR WITHIN FPXSSQ.
C
C--CALLS 'FPSXXQ' (AND IN TURN 'FCODE')
C
      DIMENSION C(1),F(1),CTEM(20)
      COMMON/PRT/IPRT
      DO 1 I=1,KIP
1     CTEM(I)=EXP_(C(I))
      CALL FPXSSQ(CTEM,N,KIP,F)
      RETURN
      END

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

```
C--GENERAL UTILITY TO MODIFY COMMON/SAVE/ AS FOLLOWS
C  CALL SAVER(I,J) WILL REPLACE FSAVE() ARRAY WITH
C  FSAVE(K)=GSAVE(K)**I * FSAVE(K)**J
C  FOR K=1,NSAVE.
C
C  INPUT PARAMETERS (I,J) MAY BE NEGATIVE,ZERO, OR POSITIVE INTEGERS.
C
C--SUBROUTINE SAVER MAY BE USED IN CONJUNCTION WITH SUBPROGRAM ZHANKS
C  TO MODIFY SAVED KERNELS WHEN USING NEW=0 OPTION (SEE ZHANKS).
C
      COMPLEX FSAVE
      COMMON/SAVE/FSAVE(283),GSAVE(283),NSAVE
      DO 1 K=1,NSAVE
      FSAVE(K)=CMPLX(GSAVE(K)**I,0.0)*(FSAVE(K)**J)
1     CONTINUE
      RETURN
      END

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

```
C               TOL.LE.0.00001 WOULD GIVE ABOUT .01 PER CENT ACCURACY
C               FOR WELL-BEHAVED KERNELS AND MODERATE VALUES OF B.  FOR
C               VERY LARGE OR SMALL B, A VERY SMALL TOL SHOULD BE USED.
C               IN GENERAL, DECREASING THE TOLERANCE WOULD PRODUCE HIGHER
C               ACCURACY IN THE CONVOLUTION SINCE MORE FILTER WEIGHTS ARE
C               USED (UNLESS EXPONENT UNDERFLOWS OCCUR IN THE KERNEL
C               EVALUATION -- SEE NOTE (1) BELOW).
C               FOR MAXIMUM ACCURACY POSSIBLE, TOL=0.0 MAY BE USED.
C     NF      = TOTAL NUMBER OF DIRECT FUN CALLS USED DURING CONVOLUTION
C               FOR ANY VALUE OF NEW (NF IS AN OUTPUT PARAMETER).
C               NF IS IN THE RANGE 21.LE.NF.LE.283 WHEN NEW=1.  USUALLY,
C               NF IS MUCH LESS THAN 283 (OR 0) WHEN NEW=0.
C     NEW   =1 IS REQUIRED FOR THE VERY FIRST CALL TO ZHANKS, OR IF
C               FORCING DIRECT FUNCTION FUN(G) CALLS, E.G., IF USING
C               ZHANKS FOR UNRELATED KERNELS.
C               NEW=1 INITIALIZES COMMON/SAVE/FSAVE(283),GSAVE(283),NSAVE
C               FOR NSAVE COMPLEX KERNEL VALUES IN FSAVE AND CORRESPONDING
C               REAL ARGUMENTS IN GSAVE FOR THE GIVEN PARAMETER B.
C     NEW   =0 TO USE RELATED KERNELS (MODIFIED BY USER) CURRENTLY STORED
C               IN COMMON/SAVE/. FUN IS CALLED ONLY IF REQUIRED
C               DURING THE CONVOLUTION.  ADDITIONAL FUNCTION VALUES WHEN
C               NEEDED ARE AUTOMATICALLY ADDED TO THE COMMON/SAVE/ BLOCK.
C
C     ******* NOTE THAT IT IS THE USERS RESPONSIBILITY TO MODIFY THE
C               COMMON FSAVE() VALUES FOR NEW=0 CALLS, EXTERNALLY IN
C               THE USERS CALLING PROGRAM (SEE SUBPROGRAM USAGE BELOW).
C
C=============================================================================
C--SUBPROGRAM USAGE-- ZHANKS IS CALLED AS FOLLOWS
C     ...
C     COMPLEX Z1,Z2,ZHANKS,FSAVE
C     COMMON/SAVE/FSAVE(283),GSAVE(283),NSAVE
C     EXTERNAL ZF1,ZF2
C     ...
C     Z1=ZHANKS(N1,B,ZF1,TOL,NF1,1)
C     DO 1 I=1,NSAVE
C  C--MODIFY FSAVE IN COMMON/SAVE/ TO OBTAIN RELATED ZF2 FROM ZF1.
C  C--E.G. FSAVE(I)=GSAVE(I)*FSAVE(I) -- FOR RELATION ZF2(G)=G*ZF1(G)
C   1 CONTINUE
C     Z2=ZHANKS(N2,B,ZF2,TOL,NF2,0)
C     ...
C     END
C     COMPLEX FUNCTION ZF1(G)
C     ...USER SUPPLIED CODE FOR DIRECT EVALUATION OF ZF1(G), G.GT.0.
C     END
C     COMPLEX FUNCTION ZF2(G)
C     ...USER SUPPLIED CODE FOR DIRECT EVALUATION OF ZF2(G), G.GT.0.
C     END
C=============================================================================
C--NOTES
C       (1).  EXP-UNDERFLOW MAY OCCUR IN EXECUTING THIS SUBPROGRAM.
C             THIS IS OK PROVIDED THE MACHINE SYSTEM CONDITIONALLY SETS
```

```
C           EXP-UNDERFLOW TO 0.0.
C      (2).  ANSI FORTRAN (AMERICAN STANDARD X3.9-1966) IS USED, EXCEPT
C           DATA STATEMENTS MAY NEED TO BE CHANGED FOR SOME COMPILERS.
C           TO CONVERT ZHANKS TO THE NEW AMERICAN STANDARD FORTRAN
C           (X3.9-1978), ADD THE FOLLOWING DECLARATION TO THIS ROUTINE
C           SAVE Y1,ISAVE
C      (3).  THE FILTER ABSCISSA CORRESPONDING TO EACH FILTER WEIGHT
C           IS GENERATED IN DOUBLE-PRECISION (TO REDUCE ROUND-OFF),
C           BUT IS USED IN SINGLE-PRECISION IN FUNCTION FUN.
C      (4).  NO CHECKS ARE MADE ON CALLING PARAMETERS (TO SAVE TIME),
C           HENCE UNPREDICTABLE RESULTS COULD OCCUR IF ZHANKS
C           IS CALLED INCORRECTLY (OR IF FUN OR COMMON IS IN ERROR).
C=======================================================================
C
      COMPLEX FUN,C,CMAX,FSAVE
      COMMON/SAVE/FSAVE(283),GSAVE(283),NSAVE
      DOUBLE PRECISION E,ER,Y1,Y
      DIMENSION T(2),TMAX(2)
      DIMENSION WT0(283),WA0(76),WB0(76),WC0(76),WD0(55),
     * WT1(283),WA1(76),WB1(76),WC1(76),WD1(55)
      EQUIVALENCE (WT0(1),WA0(1)),(WT0(77),WB0(1)),(WT0(153),WC0(1)),
     * (WT0(229),WD0(1)),(WT1(1),WA1(1)),(WT1(77),WB1(1)),
     * (WT1(153),WC1(1)),(WT1(229),WD1(1))
      EQUIVALENCE (C,T(1)),(CMAX,TMAX(1))
C------E=DEXP(.2D0), ER=1.0D0/E
      DATA E/1.2214027581601698 34 D0/,ER/.81873075307798185 9 D0/
C--J0-TRANSFORM FILTER WEIGHT ARRAYS (EQUIVALENT TO WT0 ARRAY)
      DATA WA0/
     * 2.1969101E-11, 4.1201161E-09,-6.1322980E-09, 7.2479291E-09,
     *-7.9821627E-09, 8.5778983E-09,-9.1157294E-09, 9.6615250E-09,
     *-1.0207546E-08, 1.0796633E-08,-1.1393033E-08, 1.2049873E-08,
     *-1.2708789E-08, 1.3446466E-08,-1.4174300E-08, 1.5005577E-08,
     *-1.5807160E-08, 1.6747136E-08,-1.7625961E-08, 1.8693427E-08,
     *-1.9650840E-08, 2.0869789E-08,-2.1903555E-08, 2.3305308E-08,
     *-2.4407377E-08, 2.6033678E-08,-2.7186773E-08, 2.9094334E-08,
     *-3.0266804E-08, 3.2534013E-08,-3.3672072E-08, 3.6408936E-08,
     *-3.7425022E-08, 4.0787921E-08,-4.1543242E-08, 4.5756842E-08,
     *-4.6035233E-08, 5.1425075E-08,-5.0893896E-08, 5.7934897E-08,
     *-5.6086570E-08, 6.5475248E-08,-6.1539913E-08, 7.4301996E-08,
     *-6.7117043E-08, 8.4767837E-08,-7.2583120E-08, 9.7366568E-08,
     *-7.7553611E-08, 1.1279873E-07,-8.1416723E-08, 1.3206914E-07,
     *-8.3217217E-08, 1.5663185E-07,-8.1482581E-08, 1.8860593E-07,
     *-7.3963141E-08, 2.3109673E-07,-5.7243707E-08, 2.8867452E-07,
     *-2.6163525E-08, 3.6808773E-07, 2.7049871E-08, 4.7932617E-07,
     * 1.1407365E-07, 6.3720626E-07, 2.5241961E-07, 8.6373487E-07,
     * 4.6831433E-07, 1.1916346E-06, 8.0099716E-07, 1.6696015E-06,
     * 1.3091334E-06, 2.3701475E-06, 2.0803829E-06, 3.4012978E-06/
      DATA WB0/
     * 3.2456774E-06, 4.9240402E-06, 5.0005198E-06, 7.1783540E-06,
     * 7.6367633E-06, 1.0522038E-05, 1.1590021E-05, 1.5488635E-05,
     * 1.7510398E-05, 2.2873836E-05, 2.6368006E-05, 3.3864387E-05,
     * 3.9610390E-05, 5.0230379E-05, 5.9397373E-05, 7.4612122E-05,
```

```
     * 8.8951409E-05, 1.1094809E-04, 1.3308026E-04, 1.6511335E-04,
     * 1.9895671E-04, 2.4587195E-04, 2.9728181E-04, 3.6629770E-04,
     * 4.4402013E-04, 5.4589361E-04, 6.6298832E-04, 8.1375348E-04,
     * 9.8971624E-04, 1.2132772E-03, 1.4772052E-03, 1.8092022E-03,
     * 2.2045122E-03, 2.6980811E-03, 3.2895354E-03, 4.0238764E-03,
     * 4.9080203E-03, 6.0010999E-03, 7.3216878E-03, 8.9489225E-03,
     * 1.0919448E-02, 1.3340696E-02, 1.6276399E-02, 1.9873311E-02,
     * 2.4233627E-02, 2.9555699E-02, 3.5990069E-02, 4.3791529E-02,
     * 5.3150319E-02, 6.4341372E-02, 7.7506720E-02, 9.2749987E-02,
     * 1.0980561E-01, 1.2791555E-01, 1.4525830E-01, 1.5820085E-01,
     * 1.6058576E-01, 1.4196085E-01, 8.9781222E-02,-1.0238278E-02,
     *-1.5083434E-01,-2.9059573E-01,-2.9105437E-01,-3.7973244E-02,
     * 3.8273717E-01, 2.2014118E-01,-4.7342635E-01, 1.9331133E-01,
     * 5.3839527E-02,-1.1909845E-01, 9.9317051E-02,-6.6152628E-02,
     * 4.0703241E-02,-2.4358316E-02, 1.4476533E-02,-8.6198067E-03/
      DATA WC0/
     * 5.1597053E-03,-3.1074602E-03, 1.8822342E-03,-1.1456545E-03,
     * 7.0004347E-04,-4.2904226E-04, 2.6354444E-04,-1.6215439E-04,
     * 9.9891279E-05,-6.1589037E-05, 3.7996921E-05,-2.3452250E-05,
     * 1.4479572E-05,-8.9417427E-06, 5.5227518E-06,-3.4114252E-06,
     * 2.1074101E-06,-1.3019229E-06, 8.0433617E-07,-4.9693681E-07,
     * 3.0702417E-07,-1.8969219E-07, 1.1720069E-07,-7.2412496E-08,
     * 4.4740283E-08,-2.7643004E-08, 1.7079403E-08,-1.0552634E-08,
     * 6.5200311E-09,-4.0284597E-09, 2.4890232E-09,-1.5378695E-09,
     * 9.5019040E-10,-5.8708696E-10, 3.6273937E-10,-2.2412348E-10,
     * 1.3847792E-10,-8.5560821E-11, 5.2865474E-11,-3.2664392E-11,
     * 2.0182948E-11,-1.2470979E-11, 7.7057678E-12,-4.7611713E-12,
     * 2.9415274E-12,-1.8170081E-12, 1.1221034E-12,-6.9271067E-13,
     * 4.2739744E-13,-2.6344388E-13, 1.6197105E-13,-9.9147443E-14,
     * 6.0487998E-14,-3.6973097E-14, 2.2817964E-14,-1.4315547E-14,
     * 9.1574735E-15,-5.9567236E-15, 3.9209969E-15,-2.5911739E-15,
     * 1.6406939E-15,-8.8248590E-16, 3.0195409E-16, 2.2622634E-17,
     *-8.0942556E-17,-3.7172363E-17, 1.9299542E-16,-3.3388160E-16,
     * 4.6174116E-16,-5.8627358E-16, 7.2227767E-16,-8.7972941E-16,
     * 1.0211793E-15,-1.0940039E-15, 1.0789555E-15,-9.7089714E-16/
      DATA WD0/
     * 7.4110927E-16,-4.1700094E-16, 8.5977184E-17, 1.3396469E-16,
     *-1.7838410E-16, 4.8975421E-17, 1.9398153E-16,-5.0046989E-16,
     * 8.3280985E-16,-1.1544640E-15, 1.4401527E-15,-1.6637066E-15,
     * 1.7777129E-15,-1.7322187E-15, 1.5247247E-15,-1.1771155E-15,
     * 6.9747910E-16,-1.2088956E-16,-4.8382957E-16, 1.0408292E-15,
     *-1.5220450E-15, 1.9541597E-15,-2.4107448E-15, 2.9241438E-15,
     *-3.5176475E-15, 4.2276125E-15,-5.0977851E-15, 6.1428456E-15,
     *-7.3949962E-15, 8.8597601E-15,-1.0515959E-14, 1.2264584E-14,
     *-1.3949870E-14, 1.5332490E-14,-1.6146782E-14, 1.6084121E-14,
     *-1.4962523E-14, 1.2794804E-14,-9.9286701E-15, 6.8825809E-15,
     *-4.0056107E-15, 1.5965079E-15,-7.2732961E-18,-4.0433218E-16,
     *-6.5679655E-16, 3.3011866E-15,-7.3545910E-15, 1.2394851E-14,
     *-1.7947697E-14, 2.3774303E-14,-3.0279168E-14, 3.9252831E-14,
     *-5.5510504E-14, 9.0505371E-14,-1.7064873E-13/
C--END OF J0 FILTER WEIGHTS
C
```

```
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, 1.8166179E-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,
     *-2.8576356E-05, 2.4711631E-05,-2.1369580E-05, 1.8479514E-05,
     *-1.5980307E-05, 1.3819097E-05,-1.1950174E-05, 1.0334008E-05,
     *-8.9364160E-06, 7.7278366E-06,-6.6827083E-06, 5.7789251E-06,
     *-4.9973715E-06, 4.3215167E-06,-3.7370660E-06, 3.2316575E-06,
     *-2.7946015E-06, 2.4166539E-06,-2.0898207E-06, 1.8071890E-06,
     *-1.5627811E-06, 1.3514274E-06,-1.1686576E-06, 1.0106059E-06,
     *-8.7392952E-07, 7.5573750E-07,-6.5353002E-07, 5.6514528E-07,
     *-4.8871388E-07, 4.2261921E-07,-3.6546333E-07, 3.1603732E-07/
      DATA WD1/
     *-2.7329579E-07, 2.3633470E-07,-2.0437231E-07, 1.7673258E-07,
     *-1.5283091E-07, 1.3216174E-07,-1.1428792E-07, 9.8831386E-08,
     *-8.5465227E-08, 7.3906734E-08,-6.3911437E-08, 5.5267923E-08,
     *-4.7793376E-08, 4.1329702E-08,-3.5740189E-08, 3.0906612E-08,
     *-2.6726739E-08, 2.3112160E-08,-1.9986424E-08, 1.7283419E-08,
     *-1.4945974E-08, 1.2924650E-08,-1.1176694E-08, 9.6651347E-09,
     *-8.3580023E-09, 7.2276490E-09,-6.2501673E-09, 5.4048822E-09,
     *-4.6739154E-09, 4.0418061E-09,-3.4951847E-09, 3.0224895E-09,
     *-2.6137226E-09, 2.2602382E-09,-1.9545596E-09, 1.6902214E-09,
     *-1.4616324E-09, 1.2639577E-09,-1.0930164E-09, 9.4519327E-10,
     *-8.1736202E-10, 7.0681930E-10,-6.1122713E-10, 5.2856342E-10,
     *-4.5707937E-10, 3.9526267E-10,-3.4180569E-10, 2.9557785E-10,
     *-2.5560176E-10, 2.2103233E-10,-1.9113891E-10, 1.6528994E-10,
     *-1.4294012E-10, 1.2361991E-10,-8.2740936E-11/
C--END OF J1 FILTER WEIGHTS
C
      NONE=0
      IF(NEW.EQ.0) GO TO 100
      NSAVE=0
C------INITIALIZE KERNEL ABSCISSA GENERATION FOR GIVEN B
      Y1=0.7358852661479794460D0/DBLE(B)
  100 ZHANKS=(0.0,0.0)
      CMAX=(0.0,0.0)
      NF=0
      Y=Y1
C------BEGIN RIGHT-SIDE CONVOLUTION AT WEIGHT 131 (EITHER NEW=1 OR 0)
      ASSIGN 110 TO M
      I=131
      Y=Y*E
      GO TO 200
  110 TMAX(1)=AMAX1(ABS(T(1)),TMAX(1))
      TMAX(2)=AMAX1(ABS(T(2)),TMAX(2))
      I=I+1
      Y=Y*E
      IF(I.LE.149) GO TO 200
      IF(TMAX(1).EQ.0.0.AND.TMAX(2).EQ.0.0) NONE=1
C------ESTABLISH TRUNCATION CRITERION (CMAX=CMPLX(TMAX(1),TMAX(2))
      CMAX=TOL*CMAX
      ASSIGN 120 TO M
      GO TO 200
C------CHECK FOR FILTER TRUNCATION AT RIGHT END
  120 IF(ABS(T(1)).LE.TMAX(1).AND.ABS(T(2)).LE.TMAX(2)) GO TO 130
```

```
      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
      ASSIGN 140 TO M
      I=130
      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*ER
      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
      ISAVE0=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*WT0(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(MSG,M5,I6,I9)
C--ERROR MESSAGE WRITE ROUTINE AND STOP, WHERE--
C
```

```
C      MSG=    ANY MULTIPLE OF 5 CHARACTERS---MAX. OF 120
C              (USE NH--------- FORM FOR ANSI COMPATABILITY)
C      M5=     NO.CHARS IN MSG/5 (REMAINDER MUST BE 0) 1.LE.M5.LE.24
C      I6=     1ST UNIT FOR WRITE(I6, ) MSG -- USUALLY I6=6 FOR LPT.
C              IF I6.LE.0 UNIT I6 IGNORED.
C      I9=     2ND UNIT FIR WRITE(I9, ) MSG --
C              IF I9.LE.0, UNIT I9 IGNORED.
C--MESSAGE WRITTEN IN FORM--
C       /ERROR--MSG HERE
C
      DIMENSION MSG(30)
      J=5*M5
      K=J/4+MOD(J,4)
      IF(I6.GT.0) WRITE(I6,10) (MSG(I),I=1,K)
   10 FORMAT(/8H ERROR--,30A4)
      IF(I9.GT.0) WRITE(I9,10) (MSG(I),I=1,K)
      CALL CLOSE_FILE('-ALL')
C
      STOP
      END

      SUBROUTINE WARN(MSG,M5,I6,I9,*)
C--WARNING MESSAGE WRITE ROUTINE WHERE:
C
C     MSG=      'ANY MULTIPLE OF 5 CHARACTERS---MAX. OF 120    '
C     M5=       NO.CHAR'S IN MSG/5 (REMAINDER MUST BE 0) 1<=M5<=24
C     I6=       1ST UNIT NO. FOR WRITE(I6, )MSG--USUALLY I6=6 FOR LPT.
C               IF I6<=0 UNIT I6 IGNORED.
C     I9=       2ND UNIT NO. FOR WRITE(I9, )MSG--IF I9<=0, I9 IGNORED
C     *=        $LABEL NO. TO RETURN AFTER ERROR CALLED ($NO. MUST BE
C               GIVEN
C
      DIMENSION MSG(30)
      J=5*M5
      K=J/4+MOD(J,4)
      IF(I6.GT.0) WRITE(I6,6) (MSG(I),I=1,K)
    6 FORMAT('0WARNING--',30A4)
      IF(I9.GT.0) WRITE(I9,6) (MSG(I),I=1,K)
      RETURN (1)
      END
```

Appendix 2.-- Conversion to other systems

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.

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

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

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

4. Subprograms ERRMSG 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, arrays TITLE and FMT).

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.

6. To replace the IMSL interface routines (IMSLMQ, FPXSSQ, LNXSSQ, and all calls to the IMSL Library) with the nonlinear least squares subprogram MARQRT (available in Anderson, 1979c), replace the main program (lines 00000010-00000140) with the following code:

```
C--NON-IMSL MAIN PROGRAM USING MARQRT (ANDERSON, 1979)
      EXTERNAL FCODE,DUMMY_PCODE,ITHXYZ,ETHXYZ
      CALL MARQRT(FCODE,DUMMY_PCODE,ITHXYZ,ETHXYZ)
      STOP
      END
```

Note: Subprogram DUMMY_PCODE is referenced as the second external parameter in the CALL MARQRT, but DUMMY_PCODE will never be called since $parms ider=1 should be used (because analytic derivatives are not used in IMSLMQ, the pcode subprogram was not needed). For systems requiring all external references to be available (even if not called), the following subprogram could be used:

```
SUBROUTINE DUMMY_PCODE(A,B,C,D,E,I,J,K)
PRINT 'USE IDER=1' STOP
END
```

If converting to subprogram MARQRT, all parameters prefixed by an "*" apply; however, parameters iopt,parm(),nsig, eps,delta, and maxfn cannot be used in this case. In addition to subprogram MARQRT, the following subprograms from Anderson (1979c) are also called: GJR, UNSCAL, ASINH, and ERRMSG.

Appendix 3.-- Test problem input/output listing

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

```
mqlvthxyz test problem2, model parms b0=1,1,1,1,.1,150
$parms n=15,ip=1,ib=1,sp=1,sy=1,m=2,iwt=1,iprt=-2,ialt=5,k=6,
b=1,1,1,1,.1,250$
(2e14.5,f13.0,e14.5)
    0.23010e-07   0.50000e-02   2.            0.23   e-07
    0.46094e-07   0.19905e-01   2.            0.469  e-07
    0.86019e-07   0.79245e-01   2.            0.86   e-07
    0.81049e-07   0.79245e+00   2.            0.81   e-07
    0.80052e-07   0.19905e+01   2.            0.8    e-07
    0.30237e-08   0.79245e-02   3.            0.3    e-08
    0.15069e-07   0.31548e-01   3.            0.15   e-07
    0.71822e-07   0.19905e+00   3.            0.72   e-07
    0.79080e-07   0.79245e+00   3.            0.79   e-07
    0.79539e-07   0.31548e+01   3.            0.8    e-07
    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.            0.155  e+01
    0.311   e+01  0.63    e+03  4.            0.31   e+01
    0.426   e+01  0.1     e+04  4.            0.43   e+01
$init  nf=5,mm=2,x=0,y=1000,istep=1,iob=4$
```

```
1T H X Y Z--     mqlvthxyz 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


1i m s l m q  --     mqlvthxyz 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    scalep= 1
 iprt= -2         iopt= 1           nsig=  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
0warning--marquardt parameter exceeded parm(3)
```

$$$$ imslmq 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

0.10042551E+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	0.48307353E-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

```
scaled covariance matrix (inverse of xJtJ)
 2  0.2154E+00  0.2419E-01  0.5905E-01 -0.7005E-01  0.1607E-01
 3  0.2419E-01  0.2442E+00  0.9803E-01 -0.8581E-02  0.3438E-01
 4  0.5905E-01  0.9803E-01  0.4654E+00  0.1937E+01 -0.8423E+00
 5 -0.7005E-01 -0.8581E-02  0.1937E+01  0.2963E+02 -0.9632E+01
 6  0.1607E-01  0.3438E-01 -0.8423E+00 -0.9632E+01  0.3815E+01

correlation matrix
 2  0.1000E+01  0.1055E+00  0.1865E+00 -0.2773E-01  0.1773E-01
 3  0.1055E+00  0.1000E+01  0.2908E+00 -0.3190E-02  0.3562E-01
 4  0.1865E+00  0.2908E+00  0.1000E+01  0.5215E+00 -0.6321E+00
 5 -0.2773E-01 -0.3190E-02  0.5215E+00  0.1000E+01 -0.9060E+00
 6  0.1773E-01  0.3562E-01 -0.6321E+00 -0.9060E+00  0.1000E+01

  ** parameter   std error   std error/parameter  (unscaled)
 2  0.1004E+01  0.2989E-02  0.2976E-02
 3  0.9989E+00  0.3165E-02  0.3169E-02
 4  0.9978E+00  0.4365E-02  0.4374E-02
 5  0.9859E-01  0.3441E-02  0.3490E-01
 6  0.1507E+03  0.1888E+01  0.1252E-01
1T H X Y Z--     mqlvthxyz test problem2, model parms b0=1,1,1,1,.1,250

primary field factors:  0.10000000E+01  0.10042551E+01  0.99890090E+00

final unscaled parameters--
                                resistivity  lyr          depth

     1       0.99777942E+00  0.10022255E+01
     2       0.98586691E-01  0.10143357E+02
     3       0.15074997E+03                   1       0.15074997E+03
```