U.S. Department of the Interior
Geological Survey

Program IMSLEXY:
Marquardt inversion of Ex and Ey frequency soundings from a grounded wire source

by

Walter L. Anderson

Open-File Report 80-1073

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.

----

* Any use of trade names in this report is for descriptive purposes only and does imply endorsement by the U.S. Geological Survey.

By Walter L. Anderson

INTRODUCTION

Program IMSLEXY is a general-purpose program for inversion of horizontal electric field component (Ex and/or Ey) frequency sounding data obtained on an assumed horizontally stratified earth for the quasi-static case (i.e., neglecting displacement currents). The source is a grounded electric dipole or finite-wire source of arbitrary length (positioned along the x-axis and centered at the origin). The Ex and Ey fields are assumed to be measured at the earth's surface. An IMSL (International Mathematical and Statistical Library, 1977) derivative-free Marquardt (1963) nonlinear least-squares subprogram (ZXSSQ) is used for inversion of frequency sounding data. See Appendix 2 notes for conversion to other systems where the IMSL library is not available. An adaptive digital filtering algorithm (ZHANKS) developed by Anderson (1979) is used for evaluating all Hankel transforms. See Anderson (1974), or Kauahikaua and Anderson (1977), for the associated forward problem solutions for Ex and Ey about a finite-wire source of arbitrary length.

The following program options are currently available:

(1) Simultaneous (or joint) inversion of Ex and Ey frequency soundings for a maximum of 9-layer models.
(2) Mixed frequency (parametric) and/or distance (geometric) sounding inversion. Also, mixed observation types can be used (e.g., amplitude, phase, real or imaginary parts).
(3) Inclusion of additional amplitude shift parameters for both Ex and Ey in the least-squares when the correct primary field normalization factors are unknown.
(4) Scaling parameter and observation spaces to constrain the solution space and to reduce round-off effects.
(5) Weighted observations.
(6) Holding certain parameters fixed (constrained).
(7) Object-time format control of reading the observed data matrix.

To provide as much timely computer information as possible, this report is being released without a mathematical formulation section. The interested reader may consult the cited references for more details.

The Fortran source listing is given in Appendix 1. A few notes regarding conversion to other systems are given in Appendix 2. Appendix 3 lists the input/output for a sample test problem run on a Honeywell 68/80 system.

PARAMETERS AND DATA REQUIRED

Parameters required by program IMSLEXY are read using Fortran namelist read statements with specific names: $parms and $init. [Note that some parameter relationships occur (e.g., see $parms "k" and $init "mm") due to the general nature of subprogram ZXSSQ, which was designed for any nonlinear least-squares problem.] 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. Preceeding the $parms statement is a required 80-character (or less) title.

The general input order read by program IMSLEXY 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	on-line printer output file (see file16 for more detailed 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	scratch disk output file used as required during execution of IMSLEXY.
file16	master print-type disk output file--contains complete printable output.

DETAILED PARAMETER AND DATA DEFINITIONS

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

[names below prefixed with a "*" are not used by IMSLEXY, 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+1, where $init parameter mm>0 is the number of layers in the model; the last two parameters represent amplitude shifts for Ex and Ey, respectively (see definitions under parameter b below).
(cref: $init parameter mm 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<=4) 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 -4<=iob<=4 (defines specific observation type in y(i));
= 2 when $init parameter iob=5 (defines mixed observation types in y(i) via x(i,2));
= 4 when $init parameter iob=6 (defines mixed observation types in y(i) via x(i,4) and distance coordinate x0 in x(i,2) and y0 in x(i,3).
(cref: $parms iwt, $init x0, y0, 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 using a revised title, $parms, object-time format, and $init parameters. Note that istop=0 can only be used whenever ialt is not 5 (because 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<=4. Note: wt(i)=1.0 is stored automatically if iwt=0 or when iwt=1 and x(i,m+1)=0.0 (to avoid division by 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, because 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 detailed printout format [for each iteration in MARQRT--but not in IMSLMQ], 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 test is termed "forced off", and will occur whenever niter has been reached and one of the other convergence criteria has not been achieved. Using a small value for 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 IMSLEXY unless one is interested in a detailed nonlinear statistical analysis of the final solution.
(See IBM Share program 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 IMSLEXY, e is equivalent to $parms eps (see below)].
(cref: Marquardt, 1963).

* tau= Convergence criterion test parameter (default 1.E-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 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 1.E-5). Note del is used only when ider=1 for estimated partial derivative calculations.
(cref: $parms ider).

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

* iout= Print output to file06 and file16 control option.
= 1 (default) for print output on 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). For this version, file06 output has been purposely reduced for time-sharing terminal use; 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., unscaled parameters).
= 1 (recommended for program IMSLEXY) 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 IMSLEXY, the initial parameters b(j)>0 must be given; hence sp=2 should not be used (sp=2 is defined here 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., 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).
Note: Due to the possible wide range of numbers commonly encountered in electromagnetic problems, it is recommended that scalep=1 and scaley=2 be generally used if converting to CALL MARQRT (as described in Appendix 2, paragraph 6). A special case automatically occurs whenever sy=2 and iob>=5 and both amplitude and phase data are included in the data matrix; in this case, the program will use ln(amplitude) or arcsinh(phase) accordingly.
(cref: $init iob and $parms b,k,n)

b()= Array of initial guesses for all k-parameters. These values must be supplied greater than zero for program IMSLEXY (i.e., positive conductivities and thicknesses). 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.

The parameter order must be given as follows:

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

b(mm+1),b(mm+2),...,b(2*mm-1) are the mm-1 layer thicknesses (in meters); in addition, include

b(2*mm)>0 as the estimated Ex amplitude shift parameter used in the model as b(2*mm)*Ex/Exp, where Exp is the primary Ex field; and

b(2*mm+1)>0 as the estimated Ey amplitude shift parameter used in the model as b(2*mm+1)*Ey/Eyp, where Eyp is the primary Ey field.

Warning: The user should be aware that use of amplitude data alone ($init iob=-1 or 1) will not in general determine absolute layer conductivities, since the shift parameters will scale all conductivities accordingly. However, layer conductivity ratios and layer thicknesses are independent of the amplitude shift parameters. The shift parameters should cause no ambiguity whenever both amplitude and phase (or real and imaginary) data are used jointly ($init iob=5 or 6) during inversion.

Note: If only phase data ($init iob=-2 or 2) or multiple distance soundings (iob=6) are used, then the shift parameters should be fixed using $parms ip and ib. Similarly, if only Ex (or Ey) data is given, then the corresponding Ey (or Ex) shift parameter should be fixed (e.g., set to 1.0) via $parms ip and ib.
(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 (prefixed by a "#") 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 strictly 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 allow convergence 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) [where we define Zx=b(2*mm)*Ex/Exp and Zy=b(2*mm+1)*Ey/Eyp]:
- = 1 (default) defines y(i) as the amplitude of Zx;
- = 2 defines y(i) as the phase of Zx, expressed in (-180,+180) degrees. [Note b(2*mm) should be fixed whenever iob=2.]
- = 3 defines y(i) as the real-part of Zx;
- = 4 defines y(i) as the imaginary-part of Zx;
- =-1 defines y(i) as the amplitude of Zy;
- =-2 defines y(i) as the phase of Zy, expressed in (-180,+180) degrees. [Note b(2*mm+1) should be fixed whenever iob=-2.]
- =-3 defines y(i) as the real-part of Zy;
- =-4 defines y(i) as the imaginary-part of Zy; (Note: for |iob|<=4, m=1 must also be given in $parms.)
- = 5 defines mixed observation-type frequency soundings, where the i-th observation type is given by x(i,2)=1.0 for amplitude of Zx, =2.0 for phase of Zx, =3.0 for real of Zx, =4.0 for imaginary of Zx, =-1.0 for amplitude of Zy, =-2.0 for phase of Zy, =-3.0 for real of Zy, or =-4.0 for imaginary of Zy. (Note that for iob=5, m=2 must also be given in $parms.)
- = 6 defines mixed observation-type frequency and/or distance soundings, where the i-th observation type is given by x(i,4) between -4.0 and 4.0 (same definitions as in iob=5 case), and x0=x(i,2) and y0=x(i,3) defines the wire-source and receiver separation. (Note that for iob=6, m=4 must also be given in $parms; also, the Ex and Ey shift parameters b(2*mm) and b(2*mm+1) should be fixed whenever iob=6 option is used.) (cref: $parms m, b(), $init mm, and DATA MATRIX NOTES).

mm= Number of layers in the model (1<=mm<=9; default mm=1).
Note: make sure $parms k=2*mm+1.
(cref: $parms k,b(), $init iob).

x0= Transmitter-receiver x-separation, where x0>0.0 meters when iob<0 (i.e., Ey data only). Note x0=0.0 is allowed when 0<iob<5 for Ex data only. Also, x0 must be given, unless iob=6 is used for distance soundings.
(cref: $init iob and DATA MATRIX NOTES).

y0= Transmitter-receiver y-separation, where y0>0 meters when iob<0 (i.e., Ey data only). Note y0=0.0 is allowed when 0<iob<5 for Ex data only. In general, y0 must be given unless iob=6 is used for distance soundings.
(cref: $init iob and DATA MATRIX NOTES).

l= 0.0 (default) defines a dipole source (recommended for initial studies for any receiver-transmitter separation). For l>0.0 meters, a finite electric wire source is assumed to be positioned along the x-axis from x=-l to x=+l (i.e., the total wire-length is 2*l meters) as described in Anderson (1974). For many cases where the radial separation distance sqrt(x0*x0+y0*y0) is much greater than 2*l, a dipole source may generally be assumed; however, one may always consider the initial dipole solution (when l=0) as a good first approximation to the layering when "near-source" distances are used. Then one may use l>0.0 (and ider=1, which is required if method=0) for the final layered solution after obtaining the l=0 solution from an initial study.
(cref: $parms ider and $init method).

ep= Requested integration accuracy for all finite integrals when l>0.0 is used. (default ep=.1e-2).
(cref: $init parmeters eps,neps,method,ier).

eps= Requested convolution integration tolerance used to compute all Hankel transforms using subprogram ZHANKS (default .1e-5). (Note that eps<=ep is required when l>0.0.)
(cref: $init parameters ep,l).

neps= Approximate number of calls to subprogram fcode before setting ep=eps (default neps=10). This option applies only when l>0.0 finite-wire option is used. Note that neps, ep, and eps may be used to significantly reduce the total computer

CPU-time when l>0.0 and xl>=.01, since higher accuracy in the finite integrals are usually not required in the early gradient search stages of the Marquardt algorithm.
(cref: $parms xl and $init parameters ep,eps,l).

method= 0 (default) to use lagged-convolution method (see Anderson, 1975) for all Hankel transforms, and adaptive quintic-spline interpolation integration over the finite-wire length (-l,l) when l>0.0. When l=0 (dipole), method=0 behaves like method=2, and direct convolution is used for all hankel transforms; method=0 is about 10-times faster than method=2 when l>0.0, and usually gives about 3 or more figure accuracy. When method=0 and l>0.0, only ider=1 can be used in the present version.
(cref: $parms ider and $init l).

= 2 to use direct convolution method for all Hankel transforms and direct adaptive integration over the finite-wire length when l>0.0. [Note: method=2 is capable of yielding higher accuracy (via parameters ier, ep and eps), but method=2 is not recommended for routine use.]
(cref: $init parameters ep,eps,neps,ier,nfin).

nfin= 1 (default) is used when method=0 to indicate the number of interpolation passes to sample in the lagged-convolution; i.e., the interpolation interval=.2/float(nfin). Normally nfin=1 is adequate for about 3-figure accuracy; however, nfin=2 or 3 will give greater accuracy, but the run time will be 2 or 3 times longer, respectively.
(cref: $init parameter method).

ier= Type of adaptive quadrature and convergence test to select for definite integration over a finite wire source (when l>0.0). Parameter ier is ignored for a dipole source (l=0.0).

= 1 for absolute error test (not generally recommended to use first).

= 2 (default) for L-one norm error test (recommended first).

= 3 for L-infinity norm error test (e.g., try if ier=2 fails). (Note that ier<=3 uses an adaptive Newton-Cotes quadrature.)

= 4 for relative error test using adaptive Gaussian quadrature. (Note: ier=4 may be faster than ier=2 because adaptive Gaussian quadrature is generally more efficient than adaptive Newton-Cotes quadrature--but not always.)

= 5 for relative error test using non-adaptive

Gaussian quadrature.
= 0 for special case to force ier=2 and to ignore all "ep warning messages", as noted under parameter mev below.
(cref: $init l,ep,mev).

mev= Maximum allowable complex function evaluations permitted in the adaptive integration procedure when l>0.0 (default mev=300).

Note: The program will print the message "warning--ep accuracy not achieved in..." if ep accuracy cannot be achieved in approximately mev function evaluations when l>0.0. In this case, and if ier<=5, the program accepts the integral and continues processing after setting ier=ier+1 (if however, ier>5 is generated, then ier=0 is automatically set to suppress any further ep warning messages). At each warning message, some additional integration information is printed regarding the accuracy (cerr) or actual error obtained. If cerr is reasonably small, one may decide to accept the results and ignore the warning. On the other hand, it may be desirable to rerun the problem with different parameter values used for method, ep, eps, neps, mev, and ier. Also, a data and parameter check should be done before rerunning the program. As a last resort, a special run may be made using ier=0 to bypass any and all ep error messages; however, this may be dangerous. Therefore ier=0 is not recommended for routine work.
(cref: $init l,method,ep,eps,neps,mev,and ier).

$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<=4. 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 type, Ex or Ey frequency sounding (-4<=iob<=4, m=1, and max. 3 items per record):

1. y(i)= i-th observation, where $init -4<=iob<=4 defines the particular type.
2. x(i,1)= i-th frequency (x(i,1)>0.0 Hz.).
3. x(i,2)= standard deviation of observation i (include only if iwt=1).

(b) Mixed observation types, Ex and/or Ey frequency soundings (iob=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 frequency (x(i,1)>0.0 Hz.).
3. x(i,2)= observation type in y(i); for Zx=b(2*mm)*Ex/Exp, use x(i,2)= 1.0 for amplitude, =2.0 for phase (degrees), =3.0 for real part, or =4.0 for imaginary part of Zx; for Zy=b(2*mm+1)*Ey/Eyp, use x(i,2)=-1.0 for amplitude, =-2.0 for phase (degrees), =-3.0 for real part, or =-4.0 for imaginary part of Zy.
4. x(i,3)= standard deviation of observation i (include only if iwt=1).

(c) Mixed observation types, Ex and/or Ey frequency and distance soundings (iob=6, m=4, and max. 6 items per record):

1. y(i)= i-th observation (where actual type is defined by x(i,4)).
2. x(i,1)= i-th frequency (x(i,1)>0.0 Hz.).
3. x(i,2)= i-th transmitter-receiver x0-coordinate, in meters (same rules as with $init x0).
4. x(i,3)= i-th transmitter-receiver y0-coordinate, in meters (same rules as with $init y0).
5. x(i,4)= observation type in y(i), where x(i,4) must be between -4.0 and 4.0 as defined in (b)3 above.
6. x(i,5)= standard deviation of observation i (include only if iwt=1).

The data matrix should be grouped or ordered with consecutive frequencies that are equal (and/or distances, if used) with respect to each observation type (for example, see the grouping used in Appendix 3). This ordering is not mandatory, but it will significantly reduce the total calculation time when l=0.0 (default case).

EXAMPLES OF INPUT PARAMETERS AND DATA ORDERING

1. Specific observation type; phase of Ey (iob=-2), finite-wire source (l>0.0), fixed shift parameter (required because iob=-2):

```
example 1.
$parms n=60,k=7,m=1,iprt=-1,sp=1,ialt=5,
 nsig=2,
 ip=2,ib=6,7,
 b=.1,.2,.3,10,20,2*1$
(2f10.0)
-2.8      1.
-4.15     1.2
-8.1      1.6
-10.2     2.
--(etc. for 56 more observations)--
$init mm=3,iob=-2,y0=100,x0=100,l=200,
 ep=.01,eps=.001,neps=30,ier=4$
```

2. Mixed observation types (real and imaginary parts), Ex and Ey soundings, dipole-source (l=0.0), unknown shift parameters:

```
example 2.
$parms n=100,k=7,m=2,iprt=-2,sp=1,ialt=5,
 b=.1,.2,.3,10,20,.5,1.5$
(3f10.0)
1.01      1.        3.
-2.3      1.        4.
0.987     1.        -3.
-5.23     1.        -4.
0.79      1.6       3.
-2.34     1.6       4.
0.867     1.6       -3.
-10.23    1.6       -4.
--(etc. for rest of soundings)--
$init mm=3,iob=5,x0=100,y0=200$
```

3. Distance Ex and Ey amplitude soundings, dipole-source (l=0), weighted observations (iwt=1), and fixed Ey shift parameter:

```
example 3.
$parms n=50,k=7,m=4,iprt=-1,sp=1,ialt=5,iwt=1,
 b=.1,.2,.3,10,20,2,1, ip=1,ib=7$
(6f10.0)
1.98      1.        100.      200.      1.        .02
0.998     1.        100.      200.      -1.       .03
--(etc. for rest of freq. sounding at this spacing)--
1.56      1.2       300.      400.      1.        .02
0.97      1.2       300.      400.      -1.       .025
```

```
1.32      4.      300.     400.      1.      .04
0.832     4.      300.     400.     -1.      .045
--(etc. for rest of freq. sounding at this spacing)--
$init mm=3,iob=6$
```

SPECIAL OBJECT FORMAT PHRASES

One may use special Fortran object formats to skip observations without changing the data matrix. For example, if we wish to use only the Ey amplitude data in example 3 above, we could set n=25 and use the format (/6f10.0). Similarly, if we wanted only Ex amplitudes to be used in example 3, then the format (6f10.0/) would accomplish the desired result.

Also, 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 after the working directory: >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 IMSLEXY by typing: imslexy

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 found on disk on Multics), and "file06" is always the on-line terminal print file. File16 should either be deleted or printed on a line printer after running program IMSLEXY. Also, file13 (if used) should be deleted after running the program. To submit the job as a batch job (called an absentee job 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 1.E-38 on Multics; this is not fatal because 0.0 replaces all underflows. To suppress the underflow messages, the command "set_ufl -off" can be used prior to executing IMSLEXY.

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.

PRINTED OUTPUT

Results are printed on logical unit 6 (file06) and on unit 16 (file16). Refer to Appendix 3 for a sample output listing of file16.

The following table defines additional names (or terms) used in the printed output files, other than $parms and $init parameters previously defined:

[names below prefixed with a "*" are not used by IMSLEXY, but are included for conversion compatibility if the CALL IMSLMQ is replaced by CALL MARQRT as described in Appendix 2, paragraph 6 (also see Marquardt (1963) and IBM Share program 1428 for more details on several of these output names); names prefixed with a "#" apply only to IMSLMQ printout (also see IMSL (1977) documentation of ZXSSQ for details on several of these output names); names without any prefix apply to both IMSLMQ or MARQRT printed output]

names/terms	definitions
sigma(i)	conductivity (in mhos/meter) of layer i, i=1,...,mm.
thick(i)	thickness (in meters) of layer i, i=1,...,mm-1.
# gradient	the gradient vector (scaled via $parms sp) corresponding to the final solution vector in IMSLMQ.
# norm gradient	square root (sqrt) of sum-of-squares of the gradient vector.

# infer	type of convergence obtained in IMSLMQ (see IMSL routine ZXSSQ for more details). Briefly, 	
	infer=0 indicates convergence failed (see # ier below for explanation).
	infer=1 indicates the first criterion (# nsig) was satisfied.
	infer=2 indicates the second criterion (# eps) was satisfied.
	infer=4 indicates the third criterion (# delta) was satisfied.
	(Note: if more than one convergence criterion were satisifed, then infer contains the sum; e.g., infer=3 means the first and second criteria were satisfied simultaneously.)

# ier		error code in IMSLMQ (see IMSL routine ZXSSQ for more details). Briefly,
	ier=0	implies no error.
	ier=129	implies a singularity in the Jacobian matrix.
	ier=130	implies that some ZXSSQ calling parameter is incorrect.
	ier=131	implies Marquardt parameter > # parm(3) was encountered. Note that this is printed as an "IMSL terminal error" in ZXSSQ--but is considered only a "warning error" in IMSLMQ (i.e., the results may still be usable even if ier=131).
	ier=132	implies a singular Jacobian matrix.
	ier=133	implies # maxfn was exceeded. Note that this is printed as an "IMSL terminal error" in ZXSSQ--but is considered only a "warning error" in IMSLMQ (i.e., the results may still be usable even if ier=133).
	ier=38	implies the Jacobian is zero (i.e., the solution is a stationary point).
# marquardt parm		same as "* lambda" below; but in IMSLMQ, the value printed is the lambda factor on the last Marquardt iteration, which is given (along with other self-explanatory terms) under the general heading: "$$$$ imslmq convergence information".
* phi -or- # ssq		weighted sum-of-squares of the residual function defined over n observations; i.e., the objective function to be minimized by nonlinear least-squares using MARQRT-* (Marquardt, 1963), or IMSLMQ-# (IMSL, 1977).
* s_e (or se) -or- # rmserr		standard error of estimate (or weighted root mean square error) defined as se=sqrt(phi/(n-k+ip)) for MARQRT-*, or rmserr=sqrt(ssq/(n-k+ip)) for IMSLMQ-#.
* iter		Marquardt (1963) major iteration count, where 1<=iter<=niter.
* length		length of the Marquardt (1963) adjustment vector delta(j),j=1,k at each iteration.
* gamma		angle (in degrees) between the gradient and Marquardt (1963) adjustment vector

	at each iteration.
* lambda	Marquardt (1963) lambda factor (=xl on iter=1) to be added to the diagonal of the Jacobian transpose times the Jacobian matrix at each iteration.
* -epsilon test	standard convergence test passed whenever abs(delta(j))/(tau+abs(b(j)))<e for all j in (1,k), where delta(j) is the Marquardt (1963) adjustment vector.
* gamma lambda test	alternate convergence test passed whenever lambda>1 and gamma>90 degrees. This criterion is used, rather than the standard epsilon test, when the parameter corrections are dependent on large rounding errors--almost certainly due to the presence of very high correlations among the parameter estimates.
*gamma epsilon test	alternate convergence test passed whenever gamma<gamcr. This criterion is used if parameter increments become small enough to pass the epsilon test as a result of successive halving of the increments. When this occurs, the value of phi is presumed minimized within the limits of the rounding error.
* -force off	no convergence occurred after niter iterations. Upon branching to the confidence limit calculations, the program will use the parameter values on the last iteration (i.e., when iter=niter).
obs.y(i)	observed y(i) input dependent variable for i=1,...,n.
cal	calculated dependent variable for i=1,...,n.
res	residual=(obs.y(i)-cal) for i=1,...,n.
* %res.err	percent residual error=100*res/cal for i=1,...,n.
x(i,j)	input x(i,j),j=1,m independent variables for i=1,...,n. (see DATA MATRIX NOTES above for specific definitions of

	x(i,j)).
* -unscaled	forced scalep=scaley=0 after the last iteration to produce unscaled statistics on convergence (or if forced off after niter).
* partials -or- # jacobian (xjtj)	(*-unscaled;#-scaled) partial derivative Jacobian matrix on the last iteration for each parameter (j=1,k), evaluated at observation i=1,...,n.
* ptp inverse -or- # xjtj inverse	inverse of the Jacobian transpose matrix times Jacobian matrix (order k).
correlation matrix	parameter correlation coefficient matrix (order k) derived from the ptp-* or xjtj-# inverse matrix.
std error(j)	parameter standard error defined as error(j)=("-unscaled-"se)*sqrt(ptp(j,j)), for j=1,...,k.
* one-parameter	one-parameter lower and upper linear confidence limits, based on Student's t=2.0 (default).
* support plane	linear lower and upper support plane confidence limits, based on variance F-ratio statistic ff=4.0 (default).
std.error/parm	parameter relative error defined as std error(j)/parameter value(j), for j=1,k.
resistivity(i)	final resistivity (in ohm-meters) of layer i, i=1,...,mm.
depth(i)	final depth (in meters) to bottom of layer i, i=1,...,mm-1.

REFERENCES

Anderson, W.L., 1974, Electromagnetic fields about a finite electric wire source: U.S. Geological Survey Report USGS-GD-74-041, 205 p. avail. from U.S. Department Commerce National Technical Information Service (NTIS), Springfield, Va., 22161 as Report PB-238-199/4WC.

-------, 1975, Improved digital filters for evaluating Fourier and Hankel transform integrals: U.S. Geological Survey Report USGS-GD-75-012, 223 p. avail. from U.S. Department Commerce NTIS, Springfield, Va., 22161 as Report PB-242-800/1WC.

-------, 1979, Numerical integration of related Hankel transforms of orders 0 and 1 by adaptive digital filtering: Geophysics, vol. 44, no. 7, p. 1287-1305.

-------, 1980, Program MARQHXY: Marquardt inversion of Hx and Hy frequency soundings from a grounded wire source: U.S. Geological Survey Open-File Report 80-901, 111 p.

Herriot, J.G., and Reinsch, C.H., 1976, Algorithm 507, Procedures for quintic natural spline interpolation: ACM Transactions on Mathematical Software, v. 2, no. 3, p. 281-289.

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

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. Geological Survey Report USGS-GD-77-007, 63 p. avail. from U.S. Department Commerce NTIS, Springfield, Va. 22161 as Report PB-274-119.

Marquardt, D.W., 1963, An algorithm for least-squares estimation of nonlinear parameters: Journal of the Society for Industrial and Applied Mathematics, v. 11, no. 2, p. 431-441.

Patterson, T.N.L., 1973, Algorithm for automatic numerical integration over a finite interval [D1]: Association for Computing Machinery Communication, v. 16, no. 11, 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, p. 250-251.

Appendix 1.-- Source listing

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

```
C--IMSLEXY: EXY INVERSION USING IMSLMQ   (2/4/80)                          00000010
      SUBROUTINE IMSLEXY_SUBZ(Y,X,B,PRNT,NPRNT,N,TITLE,IOUT)               00000110
      SUBROUTINE FCODE(Y,X,B,PRNT,F,I,IDER)                                00001270
      SUBROUTINE IMSLEXY_SUBEND(Y,X,B,K,N,TITLE,IOUT)                      00002620
      COMPLEX FUNCTION EX03(X)                                             00003000
      COMPLEX FUNCTION EY03(X)                                             00003490
      COMPLEX FUNCTION F12_2(G)                                            00003930
      COMPLEX FUNCTION F11_2(G)                                            00003990
      COMPLEX FUNCTION F7_2(G)                                             00004110
      SUBROUTINE PRMEXY                                                    00004250
      SUBROUTINE FINF7(B1,B2,F71,F72)                                      00004510
      SUBROUTINE ERRMSG(MSG,M5,I6,I9)                                      00004630
      SUBROUTINE WARN(MSG,M5,I6,I9,*)                                      00004860
      INTEGER FUNCTION LOC(I,J)                                            00005060
      COMPLEX FUNCTION CANC4(A1,B1,EP,M,N,FUN,MF,ESUM)                     00005170
      COMPLEX FUNCTION CQSUB(A, B, EPSIL, NPTS, ICHECK, RELERR, F,MEV)     00006740
      COMPLEX FUNCTION CQSUBA(A, B, EPSIL, NPTS, ICHECK, RELERR, F,MEV)    00008270
      SUBROUTINE CQUAD(A,B,RESULT,K,EPSIL,NPTS,ICHECK,F,MEV)               00009680
      COMPLEX FUNCTION FINEX(B)                                            00013200
       COMPLEX FUNCTION FINEY(B)                                           00013470
      SUBROUTINE POLAR2(Z,AMP,PHZ180)                                      00013680
      SUBROUTINE SETRHO(X)                                                 00013970
      COMPLEX FUNCTION FINITE(FUNC,BFIN)                                   00014230
      SUBROUTINE RECUR1(G,V1,F1)                                           00015210
      SUBROUTINE RECUR2(G,V1,F1,L1)                                        00015520
      COMPLEX FUNCTION ZEX(B,NEW,R)                                        00015870
      COMPLEX FUNCTION FUNINT(X)                                           00016060
      SUBROUTINE QUINT(NY,Y,B,C,D,E,F)                                     00016220
      SUBROUTINE QPOINT(NY,Y,B,C,D,E,F,X1,DELX,XX,YY)                      00016990
      COMPLEX FUNCTION F3(G)                                               00017180
        COMPLEX FUNCTION ZLAGHO(X,FUN,TOL,L,NEW)                           00017260
      SUBROUTINE IMSLMQ(SUBZ,SUBEND)                                       00019500
      SUBROUTINE FPXSSQ(C,N,KIP,F)                                         00022710
      SUBROUTINE LNXSSQ(C,N,KIP,F)                                         00023040
      COMPLEX FUNCTION ZHANKS(N,B,FUN,TOL,NF,NEW)                          00023170
      SUBROUTINE SWAP(ICODE)                                               00026590
      SUBROUTINE MODIFY(N)                                                 00026830
```

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 is usually recorded in the following mode (unless otherwise requested):

> Industry compatible: 9-track, unlabeled, EBCDIC mode, odd-parity, 800 bpi density, 80-character records (blocked, 50-card images per block), 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 reference list.

Copyright Notices

----

* present address is:

U.S. Geological Survey
Mail Stop 964
Box 25046, Federal Center
Denver, Colorado 80225

```
C--IMSLEXY: EXY INVERSION USING IMSLMQ  (2/4/80)
C
      EXTERNAL IMSLEXY_SUBZ,IMSLEXY_SUBEND
C--FOLLOWING CALL INITREF ONLY FOR HONEYWELL MULTICS SYSTEM.
C  DELETE FOR OTHER SYSTEMS.
      CALL INITREF(">UDD>EMODL_INV>WANDERSON>LIB_EM","IMSLEXY",
     & "FCODE")
      CALL IMSLMQ(IMSLEXY_SUBZ,IMSLEXY_SUBEND)
      STOP
      END

      SUBROUTINE IMSLEXY_SUBZ(Y,X,B,PRNT,NPRNT,N,TITLE,IOUT)
C--INITIALIZATION ROUTINE FOR "IMSLEXY"
C--FOLLOWING CHARACTER STMT ONLY FOR HONEYWELL MULTICS SYS:
      CHARACTER*5 TITLE(16)
      COMPLEX ZFLD,ZI1,ERRFIN
      REAL   Y(1),X(200,5),B(1),PRNT(1),EPS
      REAL   K(10),D(9),L
      COMMON/MODEL/K,D,MM
      COMMON/SHARE/EPS,C2,C3,C4,X0,Y0,YY2,RHO,RHO2,DELRHO,BB,
     1 L,DEL,DEL2,METHOD,IREST(2)
      COMMON/CTL/ZI1,ZFLD,
     1 AMP,SIG1,EXPR,EYPR,FREQ,LCOMP,ICOMP,
     2 EP,NEPS,IEPS,IOB,M1,M21,M2,M2P1,IER,IERR,MEV,IIOB
      COMMON/FINERR/HAKTOL,FINTOL,INTYPE,NFIN,NEVFIN,MEVFIN,ERRFIN,LW
      NAMELIST/INIT/IOB,MM,X0,Y0,METHOD,EPS,EP,NEPS,L,IER,MEV
        DATA ISUBZ/0/
        IF(ISUBZ.NE.0) GO TO 10
C--PRESET
        ISUBZ=1
      MM=1
      MEV=300
      IER=2
      L=0.0
      IOB=1
      X0=0.0
      Y0=0.0
      METHOD=0
      NFIN=1
      EPS=.1E-3
   10 EP=.1E-2
      NEPS=10
      READ(5,INIT)
      WRITE(6,20) TITLE
   20 FORMAT(17H1I M S L E X Y --,5X,16A5/)
      IF(IOUT.EQ.1) WRITE(16,20) TITLE
      WRITE(6,30) IOB,MM,X0,Y0,L,METHOD,IER,MEV,NFIN,EPS,EP,NEPS
      IF(IOUT.EQ.1)
     1 WRITE(16,30) IOB,MM,X0,Y0,L,METHOD,IER,MEV,NFIN,EPS,EP,NEPS
   30   FORMAT(7H IOB = ,I2,8X,5HMM = ,I2,9X,3HX0=,E12.5,4H Y0=,E12.5,
     1 3H L=,E12.5/10H METHOD = ,I1,6X,
     2 6HIER = ,I1,9X,6HMEV = ,I5,5X,5HNFIN=,I3/5H EPS=,E11.5,
```

```
     3 4H EP=,E11.5,2X,7HNEPS = ,I4)
C--TEST $INIT PARMS
      IF(MM.LT.1.OR.MM.GT.9.OR.(X0.EQ.0.0.AND.IOB.LT.0)
     1 .OR.(Y0.EQ.0.0.AND.IOB.LT.0).OR.(X0.EQ.0.0.AND.Y0.EQ.0.0.AND.
     2 IOB.NE.6).OR. EP.LT.EPS.OR.L.LT.0.0.OR.
     3 IER.LT.0.OR.IER.GT.5.OR.
     4 METHOD.LT.0.OR.METHOD.GT.2.OR.
     5 IOB.EQ.0.OR.IOB.LT.-4.OR.IOB.GT.6.OR.NEPS.LT.1)
     6 CALL ERRMSG(30HSOME $INIT PARMS OUT OF RANGE ,6,6,16)
      IIOB=IABS(IOB)
C--TEST X(I, ) DATA FOR GIVEN IOB BEFORE PROCEEDING--
      DO 60 I=1,N
      IF(X(I,1).LE.0.0) CALL ERRMSG(
     1 21HSOME FREQ=X(I,1).LE.0,5,6,16)
      IF(IIOB-5) 60,40,50
   40 J=IFIX(X(I,2))
      IF(J.LT.-4.OR.J.GT.4.OR.J.EQ.0)
     & CALL ERRMSG(
     140HSOME IOBS=X(I,2) OUT OF RANGE WHEN IOB=5,8,6,16)
      GO TO 60
   50 J=IFIX(X(I,4))
      IF(J.LT.-4.OR.J.GT.4.OR.J.EQ.0) CALL ERRMSG(
     1 41HSOME IOBS=X(I,4) OUT OF RANGE WHEN IOB=6 ,9,6,16)
      IF(J.LT.0.AND.X(I,2).EQ.0.0.OR.X(I,3).EQ.0.0)
     1 CALL ERRMSG(
     2 57HSOME X0=X(I,2) OR Y0=X(I,3)=0 WHEN X(I,4).LT.0.AND.IOB=6 ,
     3 12,6,16)
   60 CONTINUE
C--PRESET SOME GLOBAL CONSTANTS
      IERR=IER
      IF(IER.EQ.0) IERR=2
      HAKTOL=1.0E-6*EPS
      FINTOL=1.0E-3*EP
      INTYPE=IERR
      MEVFIN=2*MEV
      IF(IOB.EQ.6) GO TO 150
      YY2=Y0*Y0
      CALL SETRHO(0.0)
   70 WRITE(6,80) RHO
      IF(IOUT.EQ.1) WRITE(16,80) RHO
   80 FORMAT(///40H RECEIVER-TRANSMITTER SEPARATION (RHO) =,E12.5
     * ///18H PARAMETER ORDER--/)
   90 M1=MM-1
      M2=2*MM
      M21=M2-1
      M2P1=M2+1
      WRITE(6,100) (I,I,I=1,MM)
      IF(IOUT.EQ.1) WRITE(16,100) (I,I,I=1,MM)
  100 FORMAT(5X,I3,6X,6HSIGMA(,I3,1H))
      IF(MM.EQ.1) GO TO 132
      DO 110 I=1,M1
      J=MM+I
```

```
      IF(IOUT.EQ.1) WRITE(16,120) J,I
  110 WRITE(6,120) J,I
  120 FORMAT(5X,I3,6X,6HTHICK(,I3,1H))
  132 WRITE(6,131) M2,M2
  131 FORMAT(5X,I3,10X,2HB(,I3,25H)  EX/EXP SHIFT PARAMETER)
      IF(IOUT.EQ.1) WRITE(16,131) M2,M2
      WRITE(6,133) M2P1,M2P1
  133 FORMAT(5X,I3,10X,2HB(,I3,25H)  EY/EYP SHIFT PARAMETER)
      IF(IOUT.EQ.1) WRITE(16,133) M2P1,M2P1
C--X(I,1)=FREQ, X(I,2)=IOB TYPE (IF IOB=5), X(I,M+1)=STD.DEV. (IF IWT=1)
C  NOTE-- M=1 REQUIRED IN PGM IMSLMQ WHEN -4<=IOB<=4, AND
C  M=2 IS NECESSARY WHEN IOB=5...
C  ALSO, M=4 IS NECESSARY WHEN IOB=6...
  130 NPRNT=2
      IF(IOB.EQ.5) NPRNT=3
      IF(IOB.EQ.6) NPRNT=5
      IEPS=0
  140 RETURN
  150 WRITE(6,160)
      IF(IOUT.EQ.1) WRITE(16,160)
  160 FORMAT(///18H PARAMETER ORDER--/)
      GO TO 90
      END

      SUBROUTINE FCODE(Y,X,B,PRNT,F,I,IDER)
C--FUNCTION EVALUATION FOR NORMALIZED EX OR EY FOR "IMSLEXY"
C
C--PARAMETERS--
C
C       Y=       OBSERVED DEPENDENT VARIABLE ARRAY (DIM. N)
C       X=       OBSERVED INDEPENDENT VARIABLE ARRAY (DIM. N,5)
C       B=       CURRENT PARAMETER ARRAY ESTIMATES (DIM. K)
C       PRNT=    WORK AND PRINT ARRAY (DIM. 5)
C       F=       OUTPUT FUNCTION VALUE EVAL. FOR GIVEN Y,X,B AT OBS. I
C       I=       OBSERVATION NO. TO EVAL. F (1<=I<=N)
C       IDER=    0 IF ANALYTIC DERIVATIVES ARE USED LATER (PCODE CALLED)
C                1 IF ESTIMATED DERIVATIVES USED ONLY (PCODE NOT CALLED)
C
      REAL Y(1),X(200,5),B(1),PRNT(5),F,EPS
      COMPLEX EX03,EY03,CERR,CANC4,ZFLD,ZI1,TWOI,ECON,
     * ERRFIN,FINEX,FINEY
      REAL K(10),D(9),L
      COMMON/FX/XR2
      COMMON/MODEL/K,D,MM
      COMMON/SHARE/EPS,C2,C3,C4,XX,YY,YY2,RHO,RHO2,DELRHO,BB,
     1 L,DEL,DEL2,METHOD,IREST(2)
      COMMON/CTL/ZI1,ZFLD,
     1 AMP,SIG1,EXPR,EYPR,FREQ,LCOMP,ICOMP,
     2 EP,NEPS,IEPS,IOB,M1,M21,M2,M2P1,IER,IERR,MEV,IIOB
      COMMON/FIN/R1,R2,R0,XLEN,SIG1A,XFIN,YFIN
      COMMON/THICK/DIN(9)
      COMMON/FINERR/HAKTOL,FINTOL,INTYPE,NFIN,NEVFIN,MEVFIN,ERRFIN,LW
```

```
      EXTERNAL EX03,EY03
      DATA FREQL/0.0/,LCOMP/0/,TWOI/(0.0,2.0)/
      IF(I.GT.1.OR.MM.EQ.1) GO TO 20
      DO 10 J=2,MM
      IF(B(J).EQ.B(J-1)) CALL ERRMSG(20HSOME SIG(J)=SIG(J-1),4,6,16)
   10 CONTINUE
   20 DO 30 J=1,5
   30 PRNT(J)=X(I,J)
      IF(IOB.NE.6) GO TO 40
      ISEP=0
      IF(XX.NE.PRNT(2).OR.YY.NE.PRNT(3)) ISEP=1
   40 CONTINUE
      FREQ=PRNT(1)
      IF(I.EQ.1.OR.IDER.NE.0.OR.FREQ.NE.FREQL) GO TO 50
      JUMP=1
      IF(IOB.EQ.5) GO TO 220
      IF(IOB.EQ.6.AND.ISEP.EQ.0) GO TO 240
   50 JUMP=0
      SIG1=B(1)
      DEL2=1.0/(39.47841762E-7*SIG1*FREQ)
      ECON=TWOI/(SIG1*DEL2)
      DEL=SQRT(DEL2)
      IF(IOB.NE.6.OR.ISEP.EQ.0) GO TO 60
      XX=PRNT(2)
      YY=PRNT(3)
   60 CALL PRMEXY
   70 IF(I.EQ.1) IEPS=IEPS+1
      IF(2*IEPS.EQ.NEPS) EP=EPS
      IF(L.GT.0.0) GO TO 80
      XR2=XX*XX/RHO2
      C4=1.-2.*XR2
   80 IF(MM.EQ.1) GO TO 100
      DO 90 J=1,M1
      K(J)=B(J)/SIG1
      DIN(J)=B(J+MM)
   90 D(J)=2.0*B(J+MM)/DEL
  100 K(MM)=B(MM)/SIG1
      ICOMP=0
      IF(IOB.EQ.5) GO TO 220
      IF(IOB.EQ.6) GO TO 240
      LCOMP=0
      ICOMP=1
      IF(IOB.LT.0) ICOMP=-1
C
C--GET COMPLEX NORMALIZED FUNCTION (EX/EXP OR EY/EYP)
  110 IF(L.GT.0.0) GO TO 130
      IF(ICOMP.EQ.1) ZFLD=ECON*EX03(DUM)/EXPR
      IF(ICOMP.NE.1) ZFLD=ECON*EY03(DUM)/EYPR
  120 IF(ICOMP.EQ.1) ZFLD=B(M2)*ZFLD
      IF(ICOMP.NE.1) ZFLD=B(M2P1)*ZFLD
      IF(JUMP.EQ.1) GO TO (170,180,190,200),IOBS
      GO TO (170,180,190,200,220,240),IIOB
```

```
  130 IF(METHOD.NE.0) GO TO 131
C--FINITE WIRE METHOD=0 (L>0.0)
      XLEN=L
      SIG1A=SIG1
      XFIN=XX
      YFIN=YY
      R0=SQRT(XX*XX+YY2)
      R1=SQRT((XX+L)**2+YY2)
      R2=SQRT((XX-L)**2+YY2)
      BB=R0/DEL
      IF(ICOMP.EQ.1) ZFLD=-FINEX(BB)/EXPR
      IF(ICOMP.NE.1) ZFLD=FINEY(BB)/EYPR
      NEV=NEVFIN/2
      CERR=ERRFIN
      GO TO 150
  131 IF(ICOMP.EQ.1) GO TO 140
      IF(XX.EQ.0.0)ZFLD=(0.0,0.0)
      IF(XX.NE.0.0)ZFLD=ECON*CANC4(-L,L,EP,NEV,IERR,EY03,MEV,CERR)/EYPR
      GO TO 150
  140 IF(XX.EQ.0.0)ZFLD=-2.*ECON*CANC4(0.,L,EP,NEV,IERR,EX03,MEV,CERR)/
     * EXPR
      IF(XX.NE.0.0)ZFLD=-ECON*CANC4(-L,L,EP,NEV,IERR,EX03,MEV,CERR)/EXPR
  150 IF(NEV.LT.MEV.OR.IER.EQ.0.OR.
     1 (REAL(CERR).LE.EP.AND.AIMAG(CERR).LE.EP)) GO TO 120
      WRITE(16,160) NEV,CERR,ZFLD,ICOMP,FREQ,BB,I,EP
  160 FORMAT(/45H WARNING--EP ACCURACY NOT ACHIEVED IN FCODE--/
     1 5H NEV=,I5,6H CERR=,2E12.5,6H ZFLD=,2E12.5,7H ICOMP=,I2/
     2 6H FREQ=,E12.5,4H BB=,E12.5,3H I=,I5,4H EP=,E12.5)
      WRITE(6,160) NEV,CERR,ZFLD,ICOMP,FREQ,BB,I,EP
      IERR=IERR+1
      IF(IERR.LE.5) GO TO 120
      IERR=2
      IER=0
      GO TO 120
  170 F=CABS(ZFLD)
      AMP=F
      GO TO 210
  180 CALL POLAR2(ZFLD,AMP,PHZ)
      F=PHZ
      GO TO 210
  190 F=REAL(ZFLD)
      GO TO 210
  200 F=AIMAG(ZFLD)
  210 RETURN
  220 IOBS=PRNT(2)
  230 FREQL=FREQ
      LCOMP=ICOMP
      ICOMP=1
      IF(IOBS.LT.0) ICOMP=-1
      IOBS=IABS(IOBS)
      IF(ICOMP.EQ.LCOMP) GO TO (170,180,190,200),IOBS
      GO TO 110
```

```
  240 IOBS=PRNT(4)
      GO TO 230
      END

      SUBROUTINE IMSLEXY_SUBEND(Y,X,B,K,N,TITLE,IOUT)
C-- "IMSLEXY" TERMINATION ROUTINE (CALLED ONCE BY IMSLMQ)
C  (PARAMETERS SAME AS IN SUBROUTINE FCODE,PCODE, OR SUBZ)
C  B= FINAL SOLUTION VECTOR OBTAINED BY PGM MARQRT.
C
C--FOLLOWING CHARACTER STMT. ONLY FOR HONEYWELL MULTICS SYS:
      CHARACTER*5 TITLE(16)
      REAL Y(1),X(200,5),B(1)
      WRITE(6,10) TITLE
   10 FORMAT(//26H ********  E N D  ********,6X,16A5//
     1 28H FINAL UNSCALED PARAMETERS--,10X,11HRESISTIVITY,11X,5HDEPTH/)
      IF(IOUT.EQ.1) WRITE(16,10) TITLE
      MM=(K-1)/2
      DO 30 I=1,MM
      R=1.0/B(I)
      WRITE(6,20) I,B(I),I,R
   20 FORMAT(5X,I3,4X,E16.8,2X,I3,1X,E16.8)
      IF(IOUT.EQ.1) WRITE(16,20) I,B(I),I,R
   30 CONTINUE
      IF(K.LE.3) GO TO 52
      M2=MM+1
      K1=K-2
      D=0.0
      DO 50 I=M2,K1
      D=D+B(I)
      L=I-MM
      WRITE(6,40) I,B(I),L,D
   40 FORMAT(5X,I3,4X,E16.8,24X,I3,1X,E16.8)
      IF(IOUT.EQ.1) WRITE(16,40) I,B(I),L,D
   50 CONTINUE
   52 K1=K-1
      DO 53 I=K1,K
      WRITE(6,51) I,B(I)
   51 FORMAT(5X,I3,4X,E16.8)
      IF(IOUT.EQ.1) WRITE(16,51) I,B(I)
   53 CONTINUE
   60 RETURN
      END

      COMPLEX FUNCTION EX03(X)
C--EX COMPONENT/(ECON*C1) FOR A=0 (GROUND CASE)
C  PARAMETER
C     X      = REAL*4 ARGUMENT..NOTE: X-XX DISPLACEMENT USED IN RHO IF
C              L>0; ELSE (L=0) X IS DUMMY PARM AND WHERE RHO IS GIVEN IN
C              COMMON/SHARE/--PLUS OTHER PARAMETERS IN
C              COMMON/MODEL/--SEE COMMON STATEMENTS BELOW--
C
      REAL L,K(10),D(9)
```

```
      COMMON/MODEL/K,D,M
      COMMON/SHARE/
     * EPS,
     * C2,C3,C4,
     * XX,YY,YY2,RHO,RHO2,DELRHO,B,
     * L,DEL,DEL2,IREST(3)
      COMMON/CTL/ZI1,ZFLD,
     * AMP,SIG1,EXPR,EYPR,FREQ,LCOMP,ICOMP,
     * EP,NEPS,IEPS,IOB,M1,M21,M2,M2P1,IER,IERR,MEV,IIOB
      COMMON/SAVE/FSAVE(283),GSAVE(283),NSAVE
      COMMON/SAVE2/FSAVE2(283)
      COMMON/FX/XR2
      COMPLEX F7_2,F11_2,ZHANKS,FSAVE,FSAVE2,ZFLD,ZI1,
     * TERM1,TERM2,I1
      EXTERNAL F7_2,F11_2
      DATA I1/(0.0,-1.0)/
      EX03=(0.0,0.0)
      XD=XX
      IF(L.EQ.0.0) GO TO 10
      XD=X-XX
      CALL SETRHO(X)
10    XR2=XD**2/RHO2
      C4=1.0-2.0*XR2
      IF(M.EQ.1) GO TO 3
      IF(IOB.LT.5.OR.L.GT.0.0) GO TO 1
      IF(LCOMP.NE.0.AND.(LCOMP.NE.ICOMP)) GO TO 11
1     ZI1=ZHANKS(1,B,F7_2,EPS,LL,1)
11    TERM1=CMPLX(1.-XR2,0.0)
      IF(IOB.GT.4) CALL SWAP(1)
      DO 12 I=1,NSAVE
12    FSAVE(I)=CMPLX(GSAVE(I),0.0)*(TERM1*FSAVE(I)-FSAVE2(I))
      TERM2=ZHANKS(0,B,F11_2,EPS,LL,0)
      IF(IOB.GT.4) CALL SWAP(-1)
2     EX03=C4*ZI1/RHO-TERM2/DEL
3     TERM1=CMPLX(B,B)
      TERM2=I1*DEL2*(1.0-3.0*YY2/RHO2+(1.0+TERM1)*CEXP(-TERM1))
     $/(RHO*RHO2)
      EX03=EX03+TERM2
      RETURN
      END

      COMPLEX FUNCTION EY03(X)
C--EY COMPONENT/(ECON*C1) FOR A=0 (GROUND CASE)
C  PARAMETER
C     X      = REAL*4 ARGUMENT..NOTE: X-XX DISPLACEMENT USED IN RHO IF
C              L>0; ELSE (L=0) X IS DUMMY PARM AND WHERE RHO IS GIVEN IN
C              COMMON/SHARE/--PLUS OTHER PARAMETERS IN
C              COMMON/MODEL/--SEE COMMON STATEMENTS BELOW--
C
      REAL L,K(10),D(9)
      COMMON/FX/XR2
      COMMON/MODEL/K,D,M
```

```
      COMMON/SHARE/
     * EPS,
     * C2,C3,C4,
     * XX,YY,YY2,RHO,RHO2,DELRHO,B,
     * L,DEL,DEL2,IREST(3)
      COMMON/CTL/ZI1,ZFLD,
     * AMP,SIG1,EXPR,EYPR,FREQ,LCOMP,ICOMP,
     * EP,NEPS,IEPS,IOB,M1,M21,M2,M2P1,IER,IERR,MEV,IIOB
      COMPLEX F7_2,F12_2,ZHANKS,ZFLD,ZI1,
     * TERM1,TERM2,I3
      EXTERNAL F7_2,F12_2
      DATA I3/(0.0,-3.0)/
      EY03=(0.0,0.0)
      IF(YY.EQ.0.0) GO TO 5
      XD=XX
      IF(L.EQ.0.0) GO TO 10
      XD=X-XX
      CALL SETRHO(X)
10    IF(XD.EQ.0.0) GO TO 5
      XR2=XD**2/RHO2
      IF(M.EQ.1) GO TO 3
      IF(IOB.LT.5.OR.L.GT.0.0) GO TO 1
      IF(LCOMP.NE.0.AND.(LCOMP.NE.ICOMP)) GO TO 11
1     ZI1=ZHANKS(1,B,F7_2,EPS,LL,1)
11    IF(IOB.GT.4) CALL SWAP(1)
      CALL MODIFY(1)
      TERM1=ZHANKS(0,B,F12_2,EPS,LL,0)
      IF(IOB.GT.4) CALL SWAP(-1)
2     EY03=TERM1/DEL-CMPLX(2./RHO,0.0)*ZI1
3     TERM2=I3*DEL2/(RHO*RHO2)
      EY03=CMPLX(XD*YY/RHO2,0.0)*(EY03+TERM2)
5     RETURN
      END

      COMPLEX FUNCTION F12_2(G)
C--KERNEL FOR IMSLEXY
      COMPLEX F7_2
      F12_2=G*F7_2(G)
      RETURN
      END

      COMPLEX FUNCTION F11_2(G)
C--KERNEL FOR IMSLEXY. USES FSAVE2=I*V1*(L1-1) IN /SAVE2/ VIA NSAVE.
C
       COMPLEX FSAVE,FSAVE2,T1,F7_2,ONE
      COMMON/FX/XR2
      COMMON/SAVE/FSAVE(283),GSAVE(283),NSAVE
      COMMON/SAVE2/FSAVE2(283)
      DATA ONE/(1.0,0.0)/
      T1=F7_2(G)*(ONE-CMPLX(XR2,0.0))
      F11_2=CMPLX(G,0.0)*(T1-FSAVE2(NSAVE))
      RETURN
```

```
      END

      COMPLEX FUNCTION F7_2(G)
C--KERNEL FOR IMSLEXY. SAVES FSAVE2=I*V1*(L1-1) IN /SAVE2/ VIA NSAVE.
C
      COMPLEX V1,F1,L1,I1,ONE,TWO,C,FSAVE,FSAVE2
      COMMON/SAVE/FSAVE(283),GSAVE(283),NSAVE
      COMMON/SAVE2/FSAVE2(283)
      DATA I1,ONE,TWO/(0.0,1.0),(1.0,0.0),(2.0,0.0)/
      CALL RECUR2(G,V1,F1,L1)
      C=G
      F7_2=I1*V1*(L1-ONE)
      FSAVE2(NSAVE)=F7_2
      F7_2=F7_2+(TWO*V1*(ONE-F1))/((C+V1*F1)*(C+V1))
      RETURN
      END

      SUBROUTINE PRMEXY
C--PRIMARY EXPR AND EYPR-FIELDS, YY2, RHO,ETC FOR GIVEN X0,Y0, WHERE
C  ALL PARAMETERS (IN AND OUT) ARE STORED IN COMMON BLOCKS...
C
      COMPLEX ZFLD,ZI1
      REAL L
      COMMON/SHARE/EPS,C2,C3,C4,X0,Y0,YY2,RHO,RHO2,DELRHO,BB,
     * L,DEL,DEL2,IREST(3)
      COMMON/CTL/ZI1,ZFLD,
     * AMP,SIG1,EXPR,EYPR,FREQ,LCOMP,ICOMP,
     * EP,NEPS,IEPS,IOB,M1,M21,M2,M2P1,IER,IERR,MEV,IIOB
      YY2=Y0*Y0
      CALL SETRHO(0.0)
      IF(L.GT.0.0) GO TO 20
      R5=1.0/(RHO*RHO2*RHO2)
      EXPR=2.*(2.*X0*X0-YY2)*R5/SIG1
      EYPR=6.*X0*Y0*R5/SIG1
10    RETURN
20    T1=X0-L
      T2=X0+L
      R0=SQRT(T1*T1+YY2)**3
      R1=SQRT(T2*T2+YY2)**3
      EXPR=2.*(T2/R1-T1/R0)/SIG1
      EYPR=2.*Y0*(1./R1-1./R0)/SIG1
      RETURN
      END

      SUBROUTINE FINF7(B1,B2,F71,F72)
C--FINF7 FOR IMSLEXY (USES ZHANKS)
      COMMON/FINERR/TOL,T,IT,N,NEV,MEV,ES,LW
      COMPLEX ZHANKS,F71,F72,ES
      EXTERNAL F7_2
      F71=ZHANKS(1,B1,F7_2,TOL,LW,1)
      IF(B1.EQ.B2) GO TO 10
      F72=ZHANKS(1,B2,F7_2,TOL,LW,1)
```

```
      RETURN
10    F72=F71
      RETURN
      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 COMPATIBILITY)
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  FOR 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 # FOR WRITE(I6, ) MSG -- USUALLY I6=6 FOR LPT.
C               IF I6<=0 UNIT I6 IGNORED.
C     I9=       2ND UNIT # FOR WRITE(I9, ) MSG -- IF I9<=0, UNIT I9 IGNO
C     *=        $LABEL NO. TO RETURN AFTER ERROR CALLED ($NO. MUST BE GI
C--MESSAGE WRITTEN IN FORM:
C     'OWARNING--'MSG HERE''
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('OWARNING--',30A4)
      IF(I9.GT.0) WRITE(I9,6) (MSG(I),I=1,K)
      RETURN (1)
      END

      INTEGER FUNCTION LOC(I,J)
C--GETS ACTUAL ADDR OF A(I,J)=A(J,I) SYMMETRIC MATRIX
```

```
C  STORED AS THE VECTOR A(LOC(I,J)) OF N*(N+1)/2 ELEMENTS--
C  WHERE ANY I,J.LE.N MAY BE USED (N NOT EXPLICITLY NEEDED)...
C
      IF(I-J) 10,20,20
   10 LOC=I+(J*J-J)/2
      RETURN
   20 LOC=J+(I*I-I)/2
      RETURN
      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
      F(X)=CMPLX(ABS(REAL(X)),ABS(AIMAG(X)))
C     THE PARAMETER SETUP FOR THE INITIAL CALL
      IF(N.LE.0)GO TO 210
      IF(N.GT.5)GO TO 211
      GO TO (10,10,10,400,500),N
10      A=A1
```

```
      B=B1
      EPS=EP*63.0
      ESUM=(0.0,0.0)
      TSUM=(0.0,0.0)
      LVL=1
      DA=B-A
      FA=FUN(A)
      FS=FUN((3.0  *A+B)/4.0  )
      FM=FUN((A+B)*0.5  )
      FT=FUN((A+3.0  *B)/4.0  )
      FB=FUN(B)
      M=5
      FMAX=F(FA)
      FTST(1)=FMAX
      FTST(2)=F(FS)
      FTST(3)=F(FM)
      FTST(4)=F(FT)
      FTST(5)=F(FB)
      DO 100 I=2,5
      IF(FMX(1).GE.REAL(FTST(I)))GO TO 101
      FMX(1)=REAL(FTST(I))
  101 IF(FMX(2).GE.AIMAG(FTST(I)))GO TO 100
      FMX(2)=AIMAG(FTST(I))
  100 CONTINUE
      EST=(7.0  *(FA+FB)+32.0  *(FS+FT)+12.0  *FM)*DA/90.0
      ABSAR=(7.0  *(FTST(1)+FTST(5))+32.0  *(FTST(2)+FTST(4))+12.0  *FTS
     1T(3))*DA/90.0
      AEST=ABSAR
C     1=RECUR
    1 SX=(DA/(2.0  **LVL))/90.0
      F1=FUN((7.0  *A+B)/8.0  )
      F3=FUN((5.0  *A+3.0  *B)/8.0  )
      F2(LVL)=FUN((3.0  *A+5.0  *B)/8.0  )
      F4(LVL)=FUN((A+7.0  *B)/8.0  )
      EST1=SX*(7.0  *(FA+FM)+32.0  *(F1+F3)+12.0  *FS)
      FBP(LVL)=FB
      FTP(LVL)=FT
      XB(LVL)=B
      EST2(LVL)=SX*(7.0  *(FM+FB)+32.0  *(F2(LVL)+F4(LVL))+12.0  *FT)
      SUM=EST1+EST2(LVL)
      FTST(1)=F(F1)
      FTST(2)=F(F2(LVL))
      FTST(3)=F(F3)
      FTST(4)=F(F4(LVL))
      FTST(5)=F(FM)
      AEST1=SX*(7.0  *(F(FA)   +FTST(5))+32.0  *(FTST(1)+FTST(3))+12.0
     X*F(FS))
      AEST2(LVL)=SX*(7.0  *(FTST(5)+F(FB)   )+32.0  *(FTST(2)+FTST(4))+1
     X2.0*F(FT))
      ABSAR=ABSAR-AEST+AEST1+AEST2(LVL)
      M=M+4
      IF(M.GE.MF) GO TO 5
```

```
      GO TO (201,200,202),N
  200 DELTA=ABSAR
      GO TO 205
  210 WRITE(6,39)
   39 FORMAT(' CANC4- ERROR RETURN-N.LE.0')
      GO TO 999
  211 WRITE(6,40)
   40 FORMAT(' CANC4- ERROR RETURN-N.GT.5')
      GO TO 999
  201 DELTA=(1.0,1.0)
      GO TO 205
  202 DO 203 I=1,4
      IF(FMX(1).GE.REAL(FTST(I)))GO TO 2031
      FMX(1)=REAL(FTST(I))
 2031 IF(FMX(2).GE.AIMAG(FTST(I)))GO TO 203
      FMX(2)=AIMAG(FTST(I))
  203 CONTINUE
      DELTA=FMAX
  205 DAFT=EST-SUM
      DIFF=F(DAFT)
      DAFT=DAFT/63.0
      Z=DIFF-EPS*DELTA
      IF(REAL(Z).LE.0.0.AND.AIMAG(Z).LE.0.0) GO TO 6
    3 IF(LVL-30)4,2,2
    6 IF(LVL-1)2,4,2
C     2=UP
    2 A=B
      ESUM=ESUM+DAFT
      TSUM=TSUM+SUM
    9 LVL=LVL-1
      L=NRTR(LVL)
      GO TO (11,12),L
C     11=R1,12=R2
    4 NRTR(LVL)=1
      EST=EST1
      AEST=AEST1
      FB=FM
      FT=F3
      FM=FS
      FS=F1
      B=(A+B)/2.0
      EPS=EPS/2.0
    7 LVL=LVL+1
      GO TO 1
   11 NRTR(LVL)=2
      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

      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
C RATHER DIFFICULT FOR A SPURIOUS CONVERGENCE TO SLIP
C THROUGH.
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
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 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)/
     *  0.77459666924148337704E 00,0.55555555555555555556E 00,
     *  0.88888888888888888889E 00,0.26848808986833344073E 00,
     *  0.96049126870802028342E 00,0.10465622602646726519E 00,
     *  0.43424374934680255800E 00,0.40139741477596222291E 00,
     *  0.45091653865847414235E 00,0.13441525524378422036E 00,
     *  0.51603282997079739697E-01,0.20062852937698902103E 00,
     *  0.99383196321275502221E 00,0.17001719629940260339E-01,
     *  0.88845923287225699889E 00,0.92927195315124537686E-01,
     *  0.62110294673722640294E 00,0.17151190913639138079E 00,
     *  0.22338668642896688163E 00,0.21915685840158749640E 00,
     *  0.22551049979820668739E 00,0.67207754295990703540E-01,
     *  0.25807598096176653565E-01,0.10031427861179557877E 00,
     *  0.84345657393211062463E-02,0.46462893261757986541E-01,
```

```
*  0.85755920049990351154E-01,0.10957842105592463824E 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)/
*   0.99909812496766759766E 00,0.25447807915618744154E-02,
*   0.98153114955374010687E 00,0.16446049854387810934E-01,
*   0.92965485742974005667E 00,0.35957103307129322097E-01,
*   0.83672593816886873550E 00,0.56979509494123357412E-01,
*   0.70249620649152707861E 00,0.76879620499003531043E-01,
*   0.53131974364437562397E 00,0.93627109981264473617E-01,
*   0.33113539325797683309E 00,0.10566989358023480974E 00,
*   0.11248894313318662575E 00,0.11195687302095345688E 00,
*   0.11275525672076869161E 00,0.33603877148207730542E-01,
*   0.12903800100351265626E-01,0.50157139305899537414E-01,
*   0.42176304415588548391E-02,0.23231446639910269443E-01,
*   0.42877960025007734493E-01,0.54789210527962865032E-01,
*   0.12651565562300680114E-02,0.82230079572359296693E-02,
*   0.17978551568128270333E-01,0.28489754745833548613E-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)/
*   0.38439810249455532039E-01,0.46813554990628012403E-01,
*   0.52834946790116519862E-01,0.55978436510476319408E-01,
*   0.99987288812035761194E 00,0.36322148184553065969E-03,
*   0.99720625937222195908E 00,0.25790497946856882724E-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/
 DATA
* P(225),P(226),P(227),P(228),P(229),P(230),P(231),
* P(232),P(233),P(234),P(235),P(236),P(237),P(238),
* P(239),P(240),P(241),P(242),P(243),P(244),P(245),
* P(246),P(247),P(248),P(249),P(250),P(251),P(252)/
*   0.12843824718970101768E-02,0.17864463917586498247E-02,
*   0.23355251860571608737E-02,0.29217249379178197538E-02,
*   0.35362449977167777340E-02,0.41714193769840788528E-02,
*   0.48205888648512683476E-02,0.54778666939189508240E-02,
*   0.61379152800413850435E-02,0.67957855048827733948E-02,
*   0.74468208324075910174E-02,0.80866093647888599710E-02,
*   0.87109650797320868736E-02,0.93159241280693950932E-02,
*   0.98977475240487497440E-02,0.10452925722906011926E-01,
*   0.10978183152658912470E-01,0.11470482114693874380E-01,
*   0.11927026053019270040E-01,0.12345262372243838455E-01,
*   0.12722884982732382906E-01,0.13057836688353048840E-01,
*   0.13348311463725179953E-01,0.13592756614812395910E-01,
*   0.13789874783240936517E-01,0.13938625738306850804E-01,
*   0.14038227896908623303E-01,0.14088159516508301065E-01/
 DATA
* P(253),P(254),P(255),P(256),P(257),P(258),P(259),
* P(260),P(261),P(262),P(263),P(264),P(265),P(266),
```

```
*  P(267),P(268),P(269),P(270),P(271),P(272),P(273),
*  P(274),P(275),P(276),P(277),P(278),P(279),P(280)/
*   0.99999759637974846462E 00,0.69379364324108267170E-05,
*   0.99994399620705437576E 00,0.53275293669780613125E-04,
*   0.99976049092443204733E 00,0.13575491094922871973E-03,
*   0.99938033802502358193E 00,0.24921240048299729402E-03,
*   0.99874561446809511470E 00,0.38974528447328229322E-03,
*   0.99780535449595727456E 00,0.55429531493037471492E-03,
*   0.99651414591489027385E 00,0.74028280424450333046E-03,
*   0.99483150280062100052E 00,0.94536151685852538246E-03,
*   0.99272134428278861533E 00,0.11674841174299594077E-02,
*   0.99015137040077015918E 00,0.14049079956551446427E-02,
*   0.98709252795403406719E 00,0.16561127281544526052E-02,
*   0.98351865757863272876E 00,0.19197129710138724125E-02,
*   0.97940628167086268381E 00,0.21944069253638388388E-02,
*   0.97473445975240266776E 00,0.24789582266575679307E-02/
  DATA
*  P(281),P(282),P(283),P(284),P(285),P(286),P(287),
*  P(288),P(289),P(290),P(291),P(292),P(293),P(294),
*  P(295),P(296),P(297),P(298),P(299),P(300),P(301),
*  P(302),P(303),P(304),P(305),P(306),P(307),P(308)/
*   0.96948465950245923177E 00,0.27721957645934509940E-02,
*   0.96364062156981213252E 00,0.30730184347025783234E-02,
*   0.95718821610986096274E 00,0.33803979910869203823E-02,
*   0.95011529752129487656E 00,0.36933779170256508183E-02,
*   0.94241156519108305981E 00,0.40110687240750233989E-02,
*   0.93406843615772578800E 00,0.43326409680929828545E-02,
*   0.92507893290707565236E 00,0.46573172997568547773E-02,
*   0.91543758715576504064E 00,0.49843645647655386012E-02,
*   0.90514035881326159519E 00,0.53130866051870565663E-02,
*   0.89418456833555902286E 00,0.56428181013844441585E-02,
*   0.88256884024734190684E 00,0.59729195655081658049E-02,
*   0.87029305554811390585E 00,0.63027734490857587172E-02,
*   0.85735831088623215653E 00,0.66317812429018878941E-02,
*   0.84376688267270860104E 00,0.69593614093904229394E-02/
  DATA
*  P(309),P(310),P(311),P(312),P(313),P(314),P(315),
*  P(316),P(317),P(318),P(319),P(320),P(321),P(322),
*  P(323),P(324),P(325),P(326),P(327),P(328),P(329),
*  P(330),P(331),P(332),P(333),P(334),P(335),P(336)/
*   0.82952219463740140018E 00,0.72849479805538070639E-02,
*   0.81462878765513741344E 00,0.76079896657190565832E-02,
*   0.79909229096084140180E 00,0.79279493342948491103E-02,
*   0.78291939411828301639E 00,0.82443037630328680306E-02,
*   0.76611781930376009072E 00,0.85565435613076896192E-02,
*   0.74869629361693660282E 00,0.88641732094824942641E-02,
*   0.73066452124218126133E 00,0.91667111635607884067E-02,
*   0.71203315536225203459E 00,0.94636899938300652943E-02,
*   0.69281376977911470289E 00,0.97546565363174114611E-02,
*   0.67301883023041847920E 00,0.10039172044056840798E-01,
*   0.65266166541001749610E 00,0.10316812330947621682E-01,
*   0.63175643771119423041E 00,0.10587167904885197931E-01,
```

```
     *  0.61031811371518640016E 00,0.10849844089337314099E-01,
     *  0.58836243444766254143E 00,0.11104461134006926537E-01/
      DATA
     * P(337),P(338),P(339),P(340),P(341),P(342),P(343),
     * P(344),P(345),P(346),P(347),P(348),P(349),P(350),
     * P(351),P(352),P(353),P(354),P(355),P(356),P(357),
     * P(358),P(359),P(360),P(361),P(362),P(363),P(364)/
     *  0.56590588542365442262E 00,0.11350654315980596602E-01,
     *  0.54296566649831149049E 00,0.11588074033043952568E-01,
     *  0.51955966153745702199E 00,0.11816385890830235763E-01,
     *  0.49570640791876146017E 00,0.12035270785279562630E-01,
     *  0.47142506587165887693E 00,0.12244424981611985899E-01,
     *  0.44673538766202847374E 00,0.12443560190714035263E-01,
     *  0.42165768662616330006E 00,0.12632403643542078765E-01,
     *  0.39621280605761593918E 00,0.12810698163877361967E-01,
     *  0.37042208795007823014E 00,0.12978202239537399286E-01,
     *  0.34430734159943802278E 00,0.13134690091960152836E-01,
     *  0.31789081206847668318E 00,0.13279951743930530650E-01,
     *  0.29119514851824668196E 00,0.13413793085110098513E-01,
     *  0.26424337241092676194E 00,0.13536035934956213614E-01,
     *  0.23705884558982972721E 00,0.13646518102571291428E-01/
      DATA
     * P(365),P(366),P(367),P(368),P(369),P(370),P(371),
     * P(372),P(373),P(374),P(375),P(376),P(377),P(378),
     * P(379),P(380),P(381)/
     *  0.20966523824318119477E 00,0.13745093443001896632E-01,
     *  0.18208649675925219825E 00,0.13831631909506428676E-01,
     *  0.15434681148137810869E 00,0.13906019601325461264E-01,
     *  0.12647058437230196685E 00,0.13968158806516938516E-01,
     *  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 FINEX(B)
C-- EX FIELD FOR FINITE WIRE (L.GT.0) AND GROUND (H=0) USING
C  LAG-CONVOLUTION AND QUINTIC SPLINE INTERPOLATION.
C  CALLS SUBR 'FINITE' (WHICH CALLS 'ZEX').
C
      EXTERNAL ZEX
      COMPLEX FINITE,F71,F72,FINEY
      REAL L
      COMMON/FIN/R1,R2,R,L,SIG1,X,Y
      COMMON/PASS/FINEY
      COMMON/MODEL/RK(10),DD(9),M
      DEL=R/B
      DEL2=DEL*DEL
      FINEX=FINITE(ZEX,B)
      FINEY=CMPLX(0.0,0.0)
```

```
      IF(M.EQ.1) GO TO 10
      B1=R1/DEL
      B2=R2/DEL
      CALL FINF7(B1,B2,F71,F72)
      FINEX=FINEX-CMPLX(0.0,1./DEL2)*((X+L)*F71/R1-(X-L)*F72/R2)
   10 FINEX=-(FINEX+(X+L)/R1**3-(X-L)/R2**3)/(.5*SIG1)
      IF(X*Y.EQ.0.0) GO TO 20
      IF(M.EQ.1) GO TO 15
      FINEY=CMPLX(0.0,Y/DEL2)*(F71/R1-F72/R2)
   15 FINEY=-(FINEY-(Y/R1**3-Y/R2**3))/(.5*SIG1)
   20 RETURN
      END

       COMPLEX FUNCTION FINEY(B)
C-- EY FIELD FOR FINITE WIRE (L.GT.0) AND GROUND (H=0) CASE.
       COMMON/FIN/R1,R2,R,L,SIG1,X,Y
       COMMON/THICK/D(9)
       COMMON/MODEL/K(10),DD(9),M
       REAL L,K
       COMPLEX F71,F72
       DEL=R/B
       DEL2=DEL*DEL
       FINEY=CMPLX(0.0,0.0)
       IF(M.EQ.1) GO TO 10
       B1=R1/DEL
       B2=R2/DEL
       M1=M-1
       DO 1 I=1,M1
1      DD(I)=2.*D(I)/DEL
       CALL FINF7(B1,B2,F71,F72)
       FINEY=CMPLX(0.0,Y/DEL2)*(F71/R1-F72/R2)
   10  FINEY=-(FINEY-Y*(1./R1**3-1./R2**3))/(.5*SIG1)
       RETURN
       END

      SUBROUTINE POLAR2(Z,AMP,PHZ180)
C        PARMS  Z  = GIVEN COMPLEX COORDS Z=(X,Y)
C               AMP= COMPUTED AMPLITUDE.
C               PHZ180 = COMPUTED PHASE IN (-180.0,180.0) DEGREES.
C
      COMPLEX Z
      DATA PI,PI2/3.1415927,6.2831853/
      ZR=REAL(Z)
      ZI=AIMAG(Z)
      IF(ZR.EQ.0.AND.ZI.EQ.0) GO TO 9
      PV=ATAN2(ABS(ZI),ABS(ZR))
      IF(ZI.GE.0.AND.ZR.GE.0) GO TO 10
      IF(ZI.GE.0.AND.ZR.LT.0) GO TO 20
      IF(ZI.LT.0.AND.ZR.LE.0) GO TO 30
      RAD=PI2-PV
      GO TO 40
    9 PHZ180=0.
```

```
      AMP=0.
      RETURN
   10 RAD=PV
      GO TO 40
   20 RAD=PI-PV
      GO TO 40
   30 RAD=PI+PV
   40 AMP=SQRT(ZR*ZR+ZI*ZI)
      PHZ180=57.29577951*RAD
      IF(PHZ180.GT.180.0) PHZ180=PHZ180-360.0
      RETURN
      END

      SUBROUTINE SETRHO(X)
C** SP-VERSION (FOR EPS IN COMMON/SHARE/) **
C--SET RHO-DEPENDENT CONSTANTS IN COMMON/SHARE/  WHERE
C  PARAMETER
C     X     = REAL*4 ARGUMENT..NOTE: X-XX DISPLACEMENT USED IN RHO IF
C              L>0; ELSE (L=0) X IS DUMMY PARM AND WHERE RHO IS GIVEN IN
C              COMMON/SHARE/--SEE COMMON STATEMENT BELOW--
C
      REAL   EPS
      REAL   L
      COMMON/SHARE/
     * EPS,
     * C2,C3,C4,
     * XX,YY,YY2,RHO,RHO2,DELRHO,B,
     * L,DEL,DEL2,
     * METHOD,NZ,NW
      IF(L.EQ.0.0) GO TO 1
      XXX2=(X-XX)**2
    2 RHO2=XXX2+YY2
      RHO=SQRT(RHO2)
      DELRHO=DEL*RHO
      IF(DEL.NE.0.0) B=RHO/DEL
    3 RETURN
    1 XXX2=XX**2
      GO TO 2
      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 ZLAGHO'), '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 ZLAGHO)
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/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=MAXO(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)
      X0=X0-DELRLM
      DO 5 ITIME=1,NFIN
      NEW=1
      X0=X0+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

      SUBROUTINE RECUR1(G,V1,F1)
C--BACKWARD RECURRENCE FOR COMPLEX V1,F1 GIVEN REAL*4 ARGUMENT G AND:
COMMON/MODEL/ PARAMETERS:
C     K(10) = NORMALIZED CONDUCTIVITY ARRAY (M VALUES,WHERE K(1)=1.0).
C     D(9)  = LAYER THICKNESS ARRAY (M-1 VALUES) D=2*THICKNESS/DEL.
C     M     = NUMBER LAYERS (M.GE.1.AND.M.LE.10)
C             SPECIAL CASE WHEN M=1 (HOMOGENEOUS--D IGNORED)
C
C--NOTE: G,K,D ARE REAL*4
C
C
      COMMON/MODEL/K,D,M
      REAL*4 K(10),D(9)
      COMPLEX C,VM,V1,F1,EVD,ONE
      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 2
      J=M-1
    1 V1=CSQRT(CMPLX(G2,2.0*K(J)))
      EVD=CEXP(-V1*D(J))
      C=(ONE-EVD)/(ONE+EVD)
      F1=(VM*F1+V1*C)/(V1+VM*F1*C)
      IF(J.EQ.1) GO TO 3
      J=J-1
      VM=V1
      GO TO 1
    2 V1=VM
    3 RETURN
      END

      SUBROUTINE RECUR2(G,V1,F1,L1)
C--BACKWARD RECURRENCE FOR COMPLEX V1,F1,L1 GIVEN REAL*4 ARGUMENT G AND:
COMMON/MODEL/ PARAMETERS:
C     K(10) = NORMALIZED CONDUCTIVITY ARRAY (M VALUES,WHERE K(1)=1.0).
C     D(9)  = LAYER THICKNESS ARRAY (M-1 VALUES) D=2*THICKNESS/DEL.
C     M     = NUMBER LAYERS (M.GE.1.AND.M.LE.10)
C             SPECIAL CASE WHEN M=1 (HOMOGENEOUS--D IGNORED)
C
C--NOTE: G,K,D ARE REAL*4
C
C
      COMMON/MODEL/K,D,M
      REAL*4 K(10),D(9)
      COMPLEX C,VM,V1,F1,L1,E,ONE
      DATA ONE/(1.0,0.0)/
      F1=ONE
      L1=ONE
      G2=G*G
      VM=CSQRT(CMPLX(G2,2.0*K(M)))
      IF(M.EQ.1) GO TO 2
      J=M-1
    1 V1=CSQRT(CMPLX(G2,2.0*K(J)))
      E=CEXP(-V1*D(J))
      C=(ONE-E)/(ONE+E)
      F1=(VM*F1+V1*C)/(V1+VM*F1*C)
      E=K(J+1)*V1+K(J)*VM*L1*C
      IF(REAL(E).EQ.0.0.AND.AIMAG(E).EQ.0.0) E=(1.0E-30,1.0E-30)
      L1=(K(J)*VM*L1+K(J+1)*V1*C)/E
      IF(J.EQ.1) GO TO 3
      J=J-1
      VM=V1
      GO TO 1
    2 V1=VM
    3 RETURN
      END
```

```
      COMPLEX FUNCTION ZEX(B,NEW,R)
C-- EX FUNC FOR GROUNDED ELE. DIPOLE USING LAG-CONVOLUTION
C  SUBR 'ZLAGHO' FOR GIVEN IND.NO. B.GT.0.
C-- NOTE: THIS IS JUST THE HANKEL TRANSFORM TERM OF EX USED IN
C  SUBR 'FINEX'.
C  SEE FINEX SUBR FOR HALF-SPACE TERMS, ETC.
C
      COMPLEX ZLAGHO,Z2DEL3,ONE,ERRFIN
      EXTERNAL F3
      COMMON/FINERR/HAKTOL,FINTOL,INTYPE,NFIN,NEVFIN,MEVFIN,ERRFIN,LW
      COMMON/CONST/DEL,DEL2,Z2DEL3
      COMMON/MODEL/RK(10),DD(9),M
      DATA ONE/(1.0,0.0)/
      ZEX=CMPLX(0.0,0.0)
      IF(M.EQ.1) GO TO 2
      ZEX=ZLAGHO(ALOG(B),F3,HAKTOL,LW,NEW)/B
    2 ZEX=Z2DEL3*ZEX+(ONE-(ONE+CMPLX(B,B))*CEXP(-CMPLX(B,B)))/R**3
      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,RO,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

      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 F3(G)
      COMPLEX V1,F1,C,ONE
      DATA ONE/(1.0,0.0)/
      CALL RECUR1(G,V1,F1)
      C=G
      F3=(V1*C*(ONE-F1))/((C+V1*F1)*(C+V1))
      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--J0-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
      ZLAGHO=(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)
      ZLAGHO=ZLAGHO+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 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

      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/1.221402758160169834 D0/,ER/.818730753077981859 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, 2.2017964E-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 SWAP(ICODE)
C--UTILITY TO SWAP COMMON/SAVE/ AS FOLLOWS:
C  ICODE =1 TO SWAP COMMON/SAVE/ TO INTERNAL TEMP STORAGE.
C        =-1 TO RESWAP INTERNAL TEMP STORAGE TO COMMON/SAVE/.
C
C--THIS MAY BE USED IN CONJUNCTION WITH SUBPROGRAM 'ZHANKS' TO USE
C  DIFFERENT CLASSES OF INTEGRALS.  ALSO, SEE THE UTILITY
C  SUBROUTINE 'MODIFY'.
C
      COMPLEX FSAVE,FSWAP
```

```
      DIMENSION FSWAP(283),GSWAP(283)
      COMMON/SAVE/FSAVE(283),GSAVE(283),NSAVE
      IF(ICODE) 3,1,1
1     DO 2 I=1,NSAVE
      FSWAP(I)=FSAVE(I)
2     GSWAP(I)=GSAVE(I)
      NSWAP=NSAVE
      RETURN
3     DO 4 I=1,NSWAP
      FSAVE(I)=FSWAP(I)
4     GSAVE(I)=GSWAP(I)
      NSAVE=NSWAP
      RETURN
      END

      SUBROUTINE MODIFY(N)
C--UTILITY TO MODIFY COMMON/SAVE/ AS FOLLOWS:
C  N >0 TO REPLACE FSAVE(I)=FSAVE(I)*(GSAVE(I)**N), I=1,NSAVE.
C  N <0 TO REPLACE FSAVE(I)=FSAVE(I)/(GSAVE(I)**IABS(N)), I=1,NSAVE.
C--THIS MAY BE USED IN CONJUNCTION WITH SUBPROGRAM 'ZHANKS' TO
C  MODIFY SAVED KERNELS WHEN USING NEW=0 (SEE ZHANKS).
C
      COMPLEX FSAVE
      COMMON/SAVE/FSAVE(283),GSAVE(283),NSAVE
      IF(N) 5,9,1
1     IF(N.GT.1) GO TO 3
      DO 2 I=1,NSAVE
2     FSAVE(I)=FSAVE(I)*CMPLX(GSAVE(I),0.0)
      GO TO 9
3     DO 4 I=1,NSAVE
4     FSAVE(I)=FSAVE(I)*CMPLX(GSAVE(I)**N,0.0)
      GO TO 9
5     IF(N.LT.-1) GO TO 3
      DO 6 I=1,NSAVE
6     FSAVE(I)=FSAVE(I)/CMPLX(GSAVE(I),0.0)
9     RETURN
      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            (delete unless supported on system)
CALL OPEN_             (delete)
CALL CLOSE_            (delete)
EXP_                   (replace by EXP)
DEXP_                  (replace by DEXP)
CEXP_                  (replace by CEXP)
PRINT...               (replace by WRITE(6, )... if necessary)
```

3. All Multics 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. IMSLEXY_SUBZ, IMSLEXY_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, 1980), replace the main program (lines 00000010-00000100) with the following code:

```
C--NON-IMSL MAIN PROGRAM USING MARQRT (ANDERSON, 1980)
      EXTERNAL FCODE,DUMMY_PCODE,IMSLEXY_SUBZ,IMSLEXY_SUBEND
      CALL MARQRT(FCODE,DUMMY_PCODE,IMSLEXY_SUBZ,IMSLEXY_SUBEND)
      STOP
      END
```

Note that 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)
CALL ERRMSG('USE_IDER=1',2,6,16)
END
```

If converting to subprogram MARQRT, then the following subprograms from Anderson (1980) are also required: GJR, UNSCAL, and ASINH. In this case, all parameters prefixed by a "*" apply; however, parameters prefixed by a "#" (i.e., iopt,parm,nsig, eps,delta, and maxfn) cannot be used.

Appendix 3.-- Test problem input/output listing

The following input files (file05 and file10) were used to run a test problem on a Honeywell Multics system. The output listing (file16) follows beginning on the next page.

file05

```
test2_exy_reim
$parms n=36,m=2,k=5,sp=1,iprt=-2,
 b=.025,2.5,250,1.5,1.5,
 iopt=0,parm(3)=300,maxfn=200$
(2e16.8,f10.0)
$init eps=.1e-8, mm=2,iob=5,x0=50,y0=200$
```

file10

```
  0.80758867E+00  0.10000000E+00         3.
  0.96208180E+00  0.10000000E+00        -3.
  0.52626855E-03  0.10000000E+00         4.
  0.17283457E-04  0.10000000E+00        -4.
  0.80770081E+00  0.31622776E+00         3.
  0.96208242E+00  0.31622776E+00        -3.
  0.15734669E-02  0.31622776E+00         4.
  0.54484584E-04  0.31622776E+00        -4.
  0.80812377E+00  0.99999999E+00         3.
  0.96208851E+00  0.99999999E+00        -3.
  0.45748751E-02  0.99999999E+00         4.
  0.16990547E-03  0.99999999E+00        -4.
  0.80946163E+00  0.31622776E+01         3.
  0.96212579E+00  0.31622776E+01        -3.
  0.12979320E-01  0.31622776E+01         4.
  0.51481205E-03  0.31622776E+01        -4.
  0.81320451E+00  0.99999999E+01         3.
  0.96228810E+00  0.99999999E+01        -3.
  0.36638934E-01  0.99999999E+01         4.
  0.14938597E-02  0.99999999E+01        -4.
  0.82542656E+00  0.31622776E+02         3.
  0.96290014E+00  0.31622776E+02        -3.
  0.10485075E+00  0.31622776E+02         4.
  0.42054495E-02  0.31622776E+02        -4.
  0.88768530E+00  0.99999999E+02         3.
  0.96576842E+00  0.99999999E+02        -3.
  0.29715617E+00  0.99999999E+02         4.
  0.11498303E-01  0.99999999E+02        -4.
  0.12398090E+01  0.31622776E+03         3.
  0.98107996E+00  0.31622776E+03        -3.
  0.68701491E+00  0.31622776E+03         4.
  0.24755859E-01  0.31622776E+03        -4.
  0.20977235E+01  0.99999998E+03         3.
  0.10115121E+01  0.99999998E+03        -3.
  0.69602930E+00  0.99999998E+03         4.
  0.12054910E-01  0.99999998E+03        -4.
```

```
i m s l e x y --     test2_exy_reim

iob = 5         mm = 2          x0= 0.50000E+02 y0= 0.20000E+03 l= 0.00000E+00
method = 0      ier = 2         mev =   300     nfin= 1
eps=0.10000E-08 ep=0.10000E-02  neps =   10


receiver-transmitter separation (rho) = 0.20616E+03


parameter order--

      1       sigma(  1)
      2       sigma(  2)
      3       thick(  1)
      4           b(  4)  ex/exp shift parameter
      5           b(  5)  ey/eyp shift parameter
```

```
i m s l m q  --     test2_exy_reim

n=   36          k=   5          ip=   0         m=  2               e= 0.000E+00
ialt= 10         istop= 1        iwt= 0          niter=     10       scalep= 1
iprt= -2         iopt= 0         nsig=  3        maxfn=    200       eps= 0.000E+00
delta= 0.000E+00
parm= 0.100E-01 0.200E+01 0.300E+03 0.100E+00

fmt=(2e16.8,f10.0)


initial parameters

 0.25000000E-01  0.25000000E+01  0.25000000E+03  0.15000000E+01  0.15000000E+01


$$$$  imslmq convergence information:

norm of gradient                0.82893931E-06
function evaluations            0.57000000E+02
est. sign. digits               0.37034330E+01
marquardt parameter             0.10000000E+01
no. iterations                  0.60000000E+01
type convergence (infer)        1
error code (ier)                  0
residual sum-of-squares (ssq)=  0.13595515E-09

**** final unscaled parameters

 0.19999844E-01  0.20001815E+01  0.20000135E+03  0.10000023E+01  0.99999528E+00

scaled gradient

-0.32985277E-06  0.60572008E-07  0.11890488E-06 -0.72187468E-06  0.19856266E-06
```

i	obs.y(i)	cal	res	x(i,1)	x(i,2)	x(i,3)	x(i,4)	wt(i)
1	0.807589E+00	0.807588E+00	0.253E-06	0.100000E+00	0.300000E+01	0.000000E+00	0.000000E+00	0.100000E+01
2	0.962082E+00	0.962082E+00	-0.395E-06	0.100000E+00	-0.300000E+01	0.000000E+00	0.000000E+00	0.100000E+01
3	0.526269E-03	0.526277E-03	-0.813E-08	0.100000E+00	0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01
4	0.172835E-04	0.172902E-04	-0.675E-08	0.100000E+00	-0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01
5	0.807701E+00	0.807701E+00	0.261E-06	0.316228E+00	0.300000E+01	0.000000E+00	0.000000E+00	0.100000E+01
6	0.962082E+00	0.962083E+00	-0.753E-06	0.316228E+00	-0.300000E+01	0.000000E+00	0.000000E+00	0.100000E+01
7	0.157347E-02	0.157347E-02	-0.726E-08	0.316228E+00	0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01
8	0.544846E-04	0.544957E-04	-0.111E-07	0.316228E+00	-0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01
9	0.808124E+00	0.808124E+00	0.209E-06	0.100000E+01	0.300000E+01	0.000000E+00	0.000000E+00	0.100000E+01
10	0.962089E+00	0.962089E+00	-0.976E-06	0.100000E+01	-0.300000E+01	0.000000E+00	0.000000E+00	0.100000E+01
11	0.457488E-02	0.457487E-02	0.570E-08	0.100000E+01	0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01
12	0.169905E-03	0.169920E-03	-0.146E-07	0.100000E+01	-0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01
13	0.809462E+00	0.809461E+00	0.194E-06	0.316228E+01	0.300000E+01	0.000000E+00	0.000000E+00	0.100000E+01
14	0.962126E+00	0.962127E+00	-0.103E-05	0.316228E+01	-0.300000E+01	0.000000E+00	0.000000E+00	0.100000E+01
15	0.129793E-01	0.129793E-01	0.551E-07	0.316228E+01	0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01
16	0.514812E-03	0.514829E-03	-0.169E-07	0.316228E+01	-0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01
17	0.813205E+00	0.813204E+00	0.477E-06	0.100000E+02	0.300000E+01	0.000000E+00	0.000000E+00	0.100000E+01
18	0.962288E+00	0.962289E+00	-0.812E-06	0.100000E+02	-0.300000E+01	0.000000E+00	0.000000E+00	0.100000E+01
19	0.366389E-01	0.366388E-01	0.160E-06	0.100000E+02	0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01
20	0.149386E-02	0.149385E-02	0.113E-07	0.100000E+02	-0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01
21	0.825427E+00	0.825426E+00	0.797E-06	0.316228E+02	0.300000E+01	0.000000E+00	0.000000E+00	0.100000E+01
22	0.962900E+00	0.962901E+00	-0.946E-06	0.316228E+02	-0.300000E+01	0.000000E+00	0.000000E+00	0.100000E+01
23	0.104851E+00	0.104851E+00	-0.230E-06	0.316228E+02	0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01
24	0.420545E-02	0.420532E-02	0.129E-06	0.316228E+02	-0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01

25	0.887685E+00	0.887684E+00	0.939E-06	0.100000E+03	0.300000E+01	0.000000E+00	0.000000E+00	0.100000E+01
26	0.965768E+00	0.965770E+00	-0.162E-05	0.100000E+03	-0.300000E+01	0.000000E+00	0.000000E+00	0.100000E+01
27	0.297156E+00	0.297158E+00	-0.149E-05	0.100000E+03	0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01
28	0.114983E-01	0.114974E-01	0.917E-06	0.100000E+03	-0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01
29	0.123981E+01	0.123981E+01	-0.222E-05	0.316228E+03	0.300000E+01	0.000000E+00	0.000000E+00	0.100000E+01
30	0.981080E+00	0.981082E+00	-0.162E-05	0.316228E+03	-0.300000E+01	0.000000E+00	0.000000E+00	0.100000E+01
31	0.687015E+00	0.687018E+00	-0.355E-05	0.316228E+03	0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01
32	0.247559E-01	0.247535E-01	0.234E-05	0.316228E+03	-0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01
33	0.209772E+01	0.209772E+01	0.262E-05	0.100000E+04	0.300000E+01	0.000000E+00	0.000000E+00	0.100000E+01
34	0.101151E+01	0.101150E+01	0.755E-05	0.100000E+04	-0.300000E+01	0.000000E+00	0.000000E+00	0.100000E+01
35	0.696029E+00	0.696032E+00	-0.303E-05	0.100000E+04	0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01
36	0.120549E-01	0.120499E-01	0.502E-05	0.100000E+04	-0.400000E+01	0.000000E+00	0.000000E+00	0.100000E+01

**** rmserr= 0.20941943E-05

final scaled partials (jacobian)

1	-0.35045456E-02	0.31956095E-02	-0.43171730E+00	-0.80754598E+00	0.00000000E+00
2	-0.63039082E-03	0.66323971E-03	-0.14854975E+00	0.00000000E+00	-0.96178225E+00
3	-0.52312755E-03	0.16898952E-04	0.91555471E-04	-0.52615427E-03	0.00000000E+00
4	-0.16913138E-04	-0.38456892E-06	0.39404282E-04	0.00000000E+00	-0.17284209E-04
5	-0.35686531E-02	0.32559040E-02	-0.43165418E+00	-0.80754598E+00	0.00000000E+00
6	-0.55559869E-03	0.66323971E-03	-0.14843142E+00	0.00000000E+00	-0.96178225E+00
7	-0.15626839E-02	0.84435878E-04	0.26500548E-03	-0.15731564E-02	0.00000000E+00
8	-0.53261221E-04	-0.72129673E-06	0.12456078E-03	0.00000000E+00	-0.54467580E-04
9	-0.39532984E-02	0.33764930E-02	-0.43138596E+00	-0.80796397E+00	0.00000000E+00
10	-0.76929049E-03	0.78382874E-03	-0.14854186E+00	0.00000000E+00	-0.96178225E+00
11	-0.45386970E-02	0.35423029E-03	0.65299913E-03	-0.45749706E-02	0.00000000E+00
12	-0.16606764E-03	0.22080512E-06	0.39236766E-03	0.00000000E+00	-0.16990823E-03
13	-0.54170872E-02	0.37985547E-02	-0.43028939E+00	-0.80963590E+00	0.00000000E+00
14	-0.88682098E-03	0.90441778E-03	-0.14856553E+00	0.00000000E+00	-0.96178225E+00
15	-0.12839038E-01	0.11644379E-02	0.97774217E-03	-0.12977111E-01	0.00000000E+00
16	-0.50195662E-03	0.16074613E-04	0.12122856E-02	0.00000000E+00	-0.51472500E-03
17	-0.99152997E-02	0.53662121E-02	-0.42751246E+00	-0.81297978E+00	0.00000000E+00
18	-0.77997508E-03	0.48235615E-03	-0.14817108E+00	0.00000000E+00	-0.96178225E+00
19	-0.36018421E-01	0.28112319E-02	-0.12750586E-02	-0.36625889E-01	0.00000000E+00
20	-0.14476159E-02	0.90795065E-04	0.35734831E-02	0.00000000E+00	-0.14931515E-02
21	-0.27929519E-01	0.92853558E-02	-0.42770180E+00	-0.82551931E+00	0.00000000E+00
22	-0.18911725E-02	0.00000000E+00	-0.14701140E+00	0.00000000E+00	-0.96303620E+00

```
23 -0.10113098E+00  0.48687824E-02 -0.14728751E-01 -0.10486185E+00  0.00000000E+00
24 -0.39830149E-02  0.23740967E-03  0.10069130E-01  0.00000000E+00 -0.42059686E-02
25 -0.13269193E+00  0.14892746E-01 -0.47889348E+00 -0.88738101E+00  0.00000000E+00
26 -0.64855462E-02  0.18088356E-02 -0.14309057E+00  0.00000000E+00 -0.96554410E+00
27 -0.26210902E+00  0.36779656E-02 -0.41200270E-01 -0.29718695E+00  0.00000000E+00
28 -0.98036124E-02  0.24212018E-03  0.28505646E-01  0.00000000E+00 -0.11494573E-01
29 -0.58942610E+00  0.16400109E-01 -0.65129954E+00 -0.12389060E+01  0.00000000E+00
30 -0.26209300E-01 -0.78382874E-03 -0.11670977E+00  0.00000000E+00 -0.98100954E+00
31 -0.32420252E+00 -0.11817726E-01  0.20665219E+00 -0.68674847E+00  0.00000000E+00
32 -0.80571827E-02 -0.76121830E-03  0.81939527E-01  0.00000000E+00 -0.24752517E-01
33 -0.66966737E+00 -0.86824107E-02  0.12575056E+00 -0.20966101E+01  0.00000000E+00
34 -0.99580380E-02  0.20500136E-02  0.28163708E-01  0.00000000E+00 -0.10115224E+01
35  0.40702946E+00 -0.75368148E-02  0.57418463E+00 -0.69594413E+00  0.00000000E+00
36  0.32060115E-01 -0.11371169E-02  0.10028119E+00  0.00000000E+00 -0.12049708E-01

scaled covariance matrix (inverse of xjtj)
 1  0.1422E+01  0.2364E+01 -0.8599E-02 -0.2748E+00 -0.5068E-02
 2  0.2364E+01  0.3468E+04  0.6757E+02 -0.7168E+01 -0.5981E+01
 3 -0.8599E-02  0.6757E+02  0.1923E+01 -0.2600E+00 -0.1933E+00
 4 -0.2748E+00 -0.7168E+01 -0.2600E+00  0.1796E+00  0.2916E-01
 5 -0.5068E-02 -0.5981E+01 -0.1933E+00  0.2916E-01  0.1381E+00

correlation matrix
 1  0.1000E+01  0.3365E-01 -0.5199E-02 -0.5437E+00 -0.1143E-01
 2  0.3365E-01  0.1000E+01  0.8273E+00 -0.2872E+00 -0.2733E+00
 3 -0.5199E-02  0.8273E+00  0.1000E+01 -0.4424E+00 -0.3751E+00
 4 -0.5437E+00 -0.2872E+00 -0.4424E+00  0.1000E+01  0.1851E+00
 5 -0.1143E-01 -0.2733E+00 -0.3751E+00  0.1851E+00  0.1000E+01

 ** parameter   std error   std error/parameter  (unscaled)
 1  0.2000E-01  0.4995E-07  0.2498E-05
 2  0.2000E+01  0.2467E-03  0.1233E-03
 3  0.2000E+03  0.5809E-03  0.2904E-05
 4  0.1000E+01  0.8875E-06  0.8875E-06
 5  0.1000E+01  0.7783E-06  0.7783E-06

********  E N D  ********     test2_exy_reim

final unscaled parameters--         resistivity          depth

      1       0.19999844E-01   1   0.50000389E+02
      2       0.20001815E+01   2   0.49995464E+00
      3       0.20000135E+03                          1   0.20000135E+03
      4       0.10000023E+01
      5       0.99999528E+00
```