U.S. DEPARTMENT OF THE INTERIOR
GEOLOGICAL SURVEY

A general interface for producing forward solution programs (Subprogram FWDSOL)

by

Walter L. Anderson

Open-File Report 84-348

1984

DISCLAIMER

This program was written in FORTRAN-77 for a VAX-11/780 system*. Although program tests have been made, no guarantee (expressed or implied) is made by the author or the Geological Survey regarding program correctness, accuracy, or proper execution on all computer systems.

* Any use of trade names in this report is for descriptive purposes only and does not imply endorsement by the U.S. Geological Survey. This report is preliminary and has not been reviewed for conformity with U.S. Geological Survey editorial standards.

CONTENTS

INTRODUCTION

Forward modeling is the evaluation of a theoretical response function, given the parameters of a known model. Conversely, inverse modeling is to determine the unknown model parameters from an observed data function. An inverse program will obviously require several forward calculations to obtain a solution. If the response function is nonlinear in the model parameters, then a large number of forward evaluations and partial derivatives will also be required. Generally, a nonlinear least-squares (NLS) algorithm is the heart of inverse modeling, where the forward response function is called iteratively (e.g., see Dennis and others, 1979, 1981).

Normally, a stand-alone forward modeling program is written first, and then embedded into a generalized nonlinear least-squares procedure to obtain the desired inverse program. However, this report assumes either the existance of an inverse program in the form described by Anderson (1982a), or that a new series of future forward and inverse programs are to be developed in parallel using the same forward subprogram requirements.

The purpose of this report is to present a new forward solution (FWDSOL) interface that is based on the same inverse subprogram requirements as defined for the inverse interface routine NLSOL (Anderson, 1982a). Ultimately, this usage will reduce future overall coding requirements, and provide uniformity in corresponding stand-alone forward and inverse programs. In other words, once the basic forward subprograms FCODE and SUBZ (Anderson, 1982a) are written for NLSOL, they can automatically be used with FWDSOL to produce a separate foward program.

A useful byproduct of the FWDSOL interface is to provide plotting capabilities to display changes in a set of model curves for various changes in the model parameters. This is accomplished by subsequent processing using as input an external output ASCII plot file (see Appendix 3) written by FWDSOL. Non-USGS users could develop external plotting code for local systems and (or) plotters as required; see Appendix 3 for a complete description of the optional output plot file format. (The USGS plotting routines used in this report are not easily transported to other systems and plotters, and therefore are not included with the FWDSOL source code.)

The remainder of this report contains 1) a summary of the basic computations and coding requirements, 2) a detailed description of the program parameters, and 3) the VAX operating instructions. Appendix 1 offers some suggestions in converting the VAX program to other computer systems; Appendix 2 lists a simple input/output test example; Appendix 3 provides several families of sounding curves computed by varying certain model parameters and for several

FWD-programs corresponding to published NLS-programs; and Appendix 4 gives a FORTRAN-77 source listing.

SUMMARY OF CALCULATIONS

Following Anderson (1982a, p. 12), the forward problem solution used in NLSOL can be expressed in the form

$$F(I) = F[X(I,L);\ B(J)], \qquad (1)$$
$$I=1,\ldots,N;\ L=1,\ldots,M;\ J=1,\ldots,K,$$

where F is, in general, a continuous nonlinear function of M independent variables X(I,L) and K model parameters B(J); N is the number of observed dependent values used in NLSOL, where $N>K$ is required for a least-squares solution. The latter requirement is of course not imposed in FWDSOL, because F can be evaluated for any X(,L) and B(J) desired. For many one-dimensional electromagnetic (EM) and electrical problems, the functional form of F in (1) is expressed as an integral, where the K model parameters occur inside the integral. For the EM problems considered here, the M independent variables are given as frequency, time, or distance, where M and K have the ranges $1 \le M < 5$ and $1 \le K < 21$, respectively.

Examples of specific forward modeling equations can be found in the references cited in Table 1.

Table 1.--Some FWD-programs on the USGS VAX-system.

FWD-program name	Type of model: type of solution	Reference report for inverse solution
FWDHZ FWDHXY FWDEXY	wire-loop (1D): all E,H-fields, freq-domain	Anderson (1977) Anderson (1980a) Anderson (1980b)
FWDLOOP3	loop-loop (1D): 5 loop types, freq-domain	Anderson (1979a)
FWDTHC	loop-loop (1D): Horiz. coplanar, transient solution.	Anderson and Kauahikaua (1979)
DFWDTHC (double-precision)	loop-loop (1D): Horiz. coplanar, transient solution.	Anderson and Kauahikaua (1979)
FWDTCI	central-ind. loop or wire-loop (1D): transient solution.	Anderson (1982c)

DFWDTCI (double-precision)	central-ind. loop or wire-loop (1D): transient solution.	Anderson (1982c)
FWDTCO	coincident loop (1D): transient solution.	Anderson (1982b)
FWDDCLAG	Schlumberger (1D): DC soundings.	Anderson (1979c)
FWDPW	Plane-wave (1D): MT, AMT soundings.	Anderson (1979b)

The FWDSOL subprogram is listed in Appendix 4; and a double-precision version (called DFWDSOL) is also listed. The listings contain detailed comments that duplicate the same calling parameter requirements for forward subprograms FCODE and SUBZ as used by NLSOL (Anderson, 1982a). Appendix 2 gives an example using FWDSOL with a elementary function (also used in Anderson, 1982a, Appendix 2) that can be followed to produce other or more complicated FWD- and NLS-programs. The main point to note, after writing subprograms FCODE and SUBZ, is that the FWD-program is automatically available by coding the MAIN program

```
EXTERNAL FCODE,SUBZ
CALL FWDSOL(FCODE,SUBZ)
CALL EXIT
END
```

Conversely, the corresponding NLS-inversion MAIN program would be

```
EXTERNAL FCODE,SUBZ,PCODE,SUBEND
CALL NLSOL2(FCODE,PCODE,SUBZ,SUBEND)
CALL EXIT
END
```

where the additional subprograms (PCODE and SUBEND) are optional or dummy routines as described by Anderson (1982a, p. 37-38), and NLSOL2 is a VAX-NAMELIST (version 3.5) replacement for the original NLSOL subprogram.

An observed data curve could obviously be appended to any set of FWD-curves in the plot file; however, repeated executions of an FWD-program is not recommended as a way to get "graphical" NLS-inverse solutions.

PARAMETERS REQUIRED

Parameters required by subprogram FWDSOL are read using FORTRAN NAMELIST input on the VAX/VMS system (version 3.5). The namelist name used is $FWD. Default values are assumed whenever any parameter is omitted, except as noted otherwise. Preceding the $FWD statement is an 80-character title.

The general input order read by subprogram FWDSOL is as follows:

1. Title record (always required, maximum of 80-characters).
2. $FWD --nondefault parameters--$END. All $FWD parameters allowed are described in the section DETAILED PARAMETER DEFINITIONS below. Note that "$FWD " may begin in column 1, and records may be continued to succeeding records until the final $ or $END is encountered, where the "END" in $END is optional.
3. $INIT --nondefault model parameters--$END. This step is controlled by the user's subroutine SUBZ (see Appendix 3), and may be omitted in some cases. For most NLS-inverse subprograms, SUBZ uses a NAMELIST/INIT/ to input additional model parameters, which in turn is used to preset various COMMON blocks needed in FCODE and any other user subprograms. Note that SUBZ for any FWD-program runs identical to the corresponding NLS-program insofar as the $INIT parameters are concerned; however, the FWD-program name printed by SUBZ is usually changed from the NLS-program name.
4. Optionally, subsequent problem sets using changed $FWD and $INIT parameters may be given by repeating steps 1-3. All subsequent results will be appended to the output files FOR016 and FOR012 (if IPLT>0) described below. Null parameter lists "$FWD $" or "$INIT $" should be used as required (see the EXAMPLE below).

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

PROGRAM FILES

FOR098 and FOR099-- Temporary work files used during execution; all work files are deleted on program end or error return to VMS.

FOR005-- Title, $FWD, and $INIT input parameters as described above.

FOR006-- Output on-line terminal file.

FOR012-- Optional output disk plot file (only written if $FWD parameter IPLT>0; see Appendix 3 for format of FOR012).

FOR016-- Primary output disk print-file.

DETAILED PARAMETER DEFINITIONS

<u>$FWD parameters</u> (nondefault parameters must always be given)

MM= Number of layers in the model (if MODE=1 or -1), or the number of parameters given in array B() if MODE=0. (Note that $FWD MM must always be chosen the same as $INIT MM, if the latter is used in subprogram SUBZ.)

MODE= 1 (default) implies a layered model is given in arrays SIG() and H().

MODE= -1 implies a layered model is given in arrays RHO() and H().

MODE= 0 implies exactly MM-values (no layers) are given in array B(). For most FWD-programs, MODE=1 is used for EM-problems and MODE=-1 for DC-problems; MODE=0 is a special case for other possible uses.

SIG()= Conductivity (mhos/m.) array for MM layers when MODE=1.

RHO()= Resistivity (ohm-m.) array for MM layers when MODE=-1. (Note that a mapping to the opposite MODE will be printed in the output files as additional information.)

H()= Thickness (m.) array for MM-1 layers when MODE=1 or -1; if MM=1 (half-space), then array H() is not required.

SHIFT= Multiplier (or amplitude shift) factor used by some $INIT MM options as possibly defined in subprogram SUBZ. The shift factor will be automatically stored in array element B(2*MM) by the FWDSOL interface. If a "2nd shift" is needed in B(2*MM+1), then this B() element must be given explicitly in the $FWD input.

B()= Input array passed to FCODE and SUBZ subprograms (same as used in any NLS-inversion program). This array is automatically obtained from SIG(), RHO(), and H() when MODE=1 or -1; however, when MODE=0, then B() must be given in the $FWD input.

X1= Initial X(I,1) for I=1 in the generated data matrix (refer to the corresponding NLS-inversion program for X1 type; e.g., for FWDLOOP3, X1 is frequency >0 Hz.). X1 is required when NX>0.

NX= Number of points per decade to generate from X1 to XM.

NX>0 to select the X1-to-XM option.

NX<0 to select specific X(I,1)>0 ascending points in array XNX(), where NX≥-500 (i.e., maximum |NX|=500).

XM= Final X(I,1) for I=N (N determined automatically) in the generated data matrix. XM≥X1 when NX>0 is required.

XNX()= Array of specific X(I,1) points to use for X1 in the generated matrix. Note that XNX(I),I=1,|NX| is required in ascending order only when NX<0 is selected.

X2= (default 1.0) is the X(I,2) constant to use in the generated data matrix for all I; e.g., for FWDLOOP3, X2 is the loop-type (X2=1 for horiz. coplanar, X2=2 for perpendicular loops, etc.). Refer to the NLS-inversion report in Table 1 for the corresponding FWD-program for any X2 options.

X3= (default 1.0) is the X(I,3) constant to use in the generated data matrix for all I. Refer to the NLS-inversion report in Table 1 for the corresponding FWD-program for any X3 options.

X4= (default 1.0) is the X(I,4) constant to use in the generated data matrix for all I. Refer to the NLS-inversion report in Table 1 for the corresponding FWD-program for any X4 options.

IPLT= 0 (default) to NOT create the output plot file, FOR012.DAT.

IPLT= 1 to create stacked plot data on plot file FOR012.DAT using the given XT and YT axes labels.

XT= 'X' (default); up to 40-characters may be used for any x-axis title.

YT= 'Y' (default); up to 40-characters may be used for any y-axis title.

$END [end of $FWD parameters; the "END" in $END may be omitted, if desired.]

AN EXAMPLE OF INPUT PARAMETERS

```
Example Title for FWDLOOP3: HORIZ. LOOPS X2=1
$FWD MM=3,MODE=1,SIG=.02,1,.001,H=200,300,
NX=5,X1=31.5,XM=31500, X2=1,
IPLT=1,XT='FREQ. (HZ.)',YT='AMPLITUDE Z/ZO'$
$INIT MM=3,IOB=1,YO=1000$
Curve 2: PERP. LOOPS X2=2
$FWD X2=2$
$INIT $
```

(See Appendix 2 for a complete input/output example.)

VAX OPERATING INSTRUCTIONS

Assuming subprograms FWDSOL, FCODE, SUBZ, and a suitable MAIN program, were previously compiled and linked using the VAX/VMS operating system (for USGS users, use LINK main,FCODE,SUBZ,[WANDERSON]LIB1/LIBR), the following steps are general execution guidelines (note that many variations are possible using VMS in either time-sharing or batch modes):

1. Either assign (via $ASSIGN command) an input parameter file name to the logical name FOR005, or let FOR005 default to the users terminal input (if logged-in on-line). The order of the parameters on FOR005 must be given exactly as defined in the section PARAMETERS REQUIRED above. To assign FOR005, use the DCL command:

```
$ASSIGN parameter_file_name FOR005
```

2. If IPLT>0 is selected, then a specific output file name may be assigned to FOR012 (as in step 1); otherwise, the system will assume FOR012.DAT as a file name. When IPLT=0 (default), this step may be ignored.

3. The FWD-program may be executed with the DCL command:

   $RUN main_program_name (for non-USGS users).

   On the USGS system, use the command:

   $RUN [WANDERSON]FWD_name (e.g., see names available in Table 1)

The above execution steps could also be submitted (via a $SUBMIT command) to be run in batch mode. For this reason, prompting messages and user responses have been excluded from subprogram FWDSOL. VAX system-dependent commands and calls have been minimized in FWDSOL for ease of program conversion to other systems (see Appendix 1 for information on conversion problems).

Note that FOR016 is the master print (disk) file (normally called FOR016.DAT, unless assigned otherwise), and file FOR006 is usually the on-line terminal print file (or LOG file if $SUBMIT was used).

ERROR MESSAGES

Most $FWD syntactical errors are flagged and printed on files FOR006 and FOR016, and the job is aborted. If FOR005 was assigned to a disk parameter file, then correct the parameter file using any VAX editor and rerun the job (e.g., use $RUN or $SUBMIT). Other parameter errors (or omissions) are also flagged by the FWD-program, and the job is terminated.

PRINTED OUTPUT

Results are printed on file FOR016 (and possibly on FOR006 in the user's SUBZ code). Refer to Appendix 2 for a sample output listing of file FOR016 and corresponding input file FOR005. For each problem (title, $FWD, $INIT) set, a title line is printed along with any SUBZ parameter output, and a complete NAMELIST write is given for all default and initial $FWD values, as defined above. The next page repeats the title line, followed by several lines of results giving the computed sounding curve, denoted by columns headed as X(I) and Y(I), where X(I) is defined by $FWD parameters X1, NX, XM (or XNX if NX<0), and Y(I) is the computed sounding curve.

REFERENCES

Anderson, W.L., 1977, Marquardt inversion of vertical magnetic field measurements from a grounded wire source: NTIS (Nat. Tech. Info. Serv., Springfield, Va.) Rept. PB-263-924, 76 p.

----------, 1979a, Marquardt inversion of loop-loop frequency soundings: USGS Open-File Rept. 79-240, 75 p.

----------, 1979b, Marquardt inversion of plane-wave frequency soundings: USGS Open-File Rept. 79-586, 37 p.

----------, 1979c, Marquardt inversion of DC Schlumberger soundings by lagged-convolution: USGS Open-File Rept. 79-1432, 58 p.

Anderson, W.L., and Kauahikaua, J., 1979, Marquardt inversion of transient horizontal coplanar loop soundings: USGS Open-File Rept. 79-773, 75 p.

Anderson, W.L., 1980a, Marquardt inversion of Hx and Hy frequency soundings from a grounded wire source: USGS Open-File Rept. 80-901, 111 p.

----------, 1980b, Marquardt inversion of Ex and Ey frequency soundings from a grounded wire source: USGS Open-File Rept. 80-1073, 87 p.

----------, 1982a, Adaptive nonlinear least-squares solution for constrained or unconstrained minimization problems (Subprogram NLSOL): USGS Open-File Rept. 82-68, 65 p.

----------, 1982b, Nonlinear least-squares inversion of transient soundings for a coincident loop system (Program NLSTCO): USGS Open-File Rept. 82-1064, 81 p.

----------, 1982c, Nonlinear least-squares inversion of transient soundings for a central induction loop system (Program NLSTCI): USGS Open-File Rept. 82-1129, 85 p.

Dennis, J.E., Gay, D.M., and Welsh, R.E., 1979, An adaptive nonlinear least-squares algorithm: Univ. of Wisconsin MRC Tech. Sum. Rept. 2010 (also available as NTIS Rept. AD-A079-716), 40 p.

----------, 1981, An adaptive nonlinear least-squares algorithm: ACM Trans. on Math. Software, Vol. 7, No. 3, p. 348-368.

Appendix 1.-- Conversion to other systems

This program (and associated subprograms) was written in extended ANSI-standard FORTRAN-77 for the VAX-11/780 system. Conversion to systems without an ANSI-FORTRAN-77 compiler would necessitate extensive changes, particularly for all CHARACTER-type variables, IF-THEN-ELSE phrases, etc.

Changes for non-VAX systems might include some (or all) of the following FORTRAN-77 constructs and VAX concepts:

(1) Variable names with more than 6-characters.
(2) Character strings delimited by single-quote characters (e.g., 'STRING'); also, character string concatination (e.g., 'STRING1'//'STRING2').
(3) Passing variable-length character strings in subroutine calls; e.g., CHARACTER*(*) passed length character arguments.
(4) Suppression of arithmetic or exponential underflow messages; note that a VAX-11 result is automatically set to 0.0 after any underflow--which is assumed for this program package. If the target system does not set underflows to 0.0, and suppress warning messages, then a suitable conversion procedure must be used for proper operation of this program package.
(5) VAX non-ANSI NAMELIST input and output statements.

Appendix 2.-- Test problem input/output listing

The following FWD-program code (called T1FWDSOL) is taken from the same FCODE and SUBZ subprograms as found in the NLS-inversion test program in Anderson (1982a, p. 30-34); duplicate listings of the T1FWDSOL driver, FCODE, and SUBZ subprograms are also included in Appendix 4 for possible testing on other systems.

```
C <T1FWDSOL>: TEST1 FOR FWDSOL USING A 7-PARAMETER PROBLEM
C
       EXTERNAL T1FCODE,T1SUBZ
       CALL FWDSOL(T1FCODE,T1SUBZ)
       CALL EXIT
       END

       SUBROUTINE T1FCODE(Y,X,B,PASS,F,IN,IDER)
C--FUNCTION EVALUATION FOR 7-PARAMETER PROBLEM
       DIMENSION Y(1),X(500,5),B(1),PASS(5)
       PASS(1)=X(IN,1)
       T=PASS(1)
       F=B(1)+B(2)*T+B(3)*T**2+B(4)*SIN(B(5)*T)+B(6)*
      1 EXP(-B(7)*T)
       RETURN
       END

       SUBROUTINE T1SUBZ(Y,X,B,PASS,NPASS,N,TITLE,IOUT)
C--INITIALIZATION FOR 7-PARAMETER PROBLEM
C  ($INIT INPUT IS NULL HERE TO MAINTAIN COMPATIBILITY WITH
C   OTHER FWD-PROGRAMS AND NLS-PROGRAMS.)
       DIMENSION Y(1),X(500,5),B(1),PASS(5)
       CHARACTER*80 TITLE
       NAMELIST/INIT/NULL
       NPASS=1
       IF(IOUT.EQ.1) WRITE(16,10) TITLE
10     FORMAT('0<T1FWDSOL>:',5X,A)
       READ(99,INIT)  !SEE REASON FOR UNIT=99 IN APPENDIX 4
       RETURN
       END
```

The following input file (FOR005) was used to run the TEST1 problem for subprogram FWDSOL on a VAX system. The corresponding output file (FOR016) is given following FOR005. (A plot showing the results of this test is provided in Appendix 3, along with a description of the output plot file FOR012.)

FOR005

```
TEST1 (See Anderson, 1982a, p.30-34)
$FWD MM=7,MODE=0,B=1,.75,-.5,4.5,1.8,5.5,.87,
NX=-21,XNX=0,.25,.5,.75,1,1.25,1.5,1.75,2,2.25,2.5,
2.75,3,3.25,3.5,3.75,4,4.25,4.5,4.75,5, IPLT=1$
$INIT NULL=0$
CURVE 2 (same as FOR010 in Anderson, 1982a, p.32 and p.34)
$FWD B=2*1,-.5,4,2,6,1$
$INIT $
```

FOR016

```
<T1FWDSOL>:     TEST1 (See Anderson, 1982a, p.30-34)


<FWDSOL>:     TEST1 (See Anderson, 1982a, p.30-34)
 ------

$FWDO
MM       =           7,
MODE     =           0,
B        =    1.000000     ,  0.7500000     , -0.5000000     ,   4.500000     ,
              1.800000     ,   5.500000     ,  0.8700000     , 13*0.0000000E+00,
X1       =  0.0000000E+00,
NX       =         -21,
XM       =  0.0000000E+00,
XNX      =  0.0000000E+00,  0.2500000     ,  0.5000000     ,  0.7500000     ,
              1.000000     ,   1.250000     ,   1.500000     ,   1.750000     ,   2.000000     ,
              2.250000     ,   2.500000     ,   2.750000     ,   3.000000     ,   3.250000     ,
              3.500000     ,   3.750000     ,   4.000000     ,   4.250000     ,   4.500000     ,
              4.750000     ,   5.000000     , 479*0.0000000E+00,
X2       =    1.000000     ,
X3       =    1.000000     ,
X4       =    1.000000     ,
IPLT     =           1,
XT       = 'X                                         ',
YT       = 'Y                                         ',
$END
```

```
<FWDSOL>:     TEST1 (See Anderson, 1982a, p.30-34)

  I    X(I)            Y(I)

  1  0.00000000E+00  0.65000000E+01
  2  0.25000000E+00  0.75384965E+01
  3  0.50000000E+00  0.83349266E+01
  4  0.75000000E+00  0.85360880E+01
  5  0.10000000E+01  0.79365478E+01
  6  0.12500000E+01  0.65113988E+01
  7  0.15000000E+01  0.44146585E+01
  8  0.17500000E+01  0.19433299E+01
  9  0.20000000E+01 -0.52597934E+00
 10  0.22500000E+01 -0.26154525E+01
 11  0.25000000E+01 -0.40240407E+01
 12  0.27500000E+01 -0.45896087E+01
 13  0.30000000E+01 -0.43230004E+01
 14  0.32500000E+01 -0.34073052E+01
 15  0.35000000E+01 -0.21625590E+01
 16  0.37500000E+01 -0.98294276E+00
 17  0.40000000E+01 -0.25905442E+00
 18  0.42500000E+01 -0.30072534E+00
 19  0.45000000E+01 -0.12758222E+01
 20  0.47500000E+01 -0.31772103E+01
 21  0.50000000E+01 -0.58244791E+01

<T1FWDSOL>:     CURVE 2 (same as FOR010 in Anderson, 1982a, p.32 and p.34)
```

```
<FWDSOL>:     CURVE 2 (same as FOR010 in Anderson, 1982a, p.32 and p.34)
 ------

$FWDO
MM      =           7,
MODE    =           0,
B       = 2*1.000000    , -0.5000000    ,  4.000000    ,  2.000000    ,
            6.000000    ,   1.000000    , 13*0.0000000E+00,
X1      =  0.0000000E+00,
NX      =         -21,
XM      =  0.0000000E+00,
XNX     =  0.0000000E+00,  0.2500000    ,  0.5000000    ,  0.7500000    ,
            1.000000    ,   1.250000    ,   1.500000    ,   1.750000    ,   2.000000    ,
            2.250000    ,   2.500000    ,   2.750000    ,   3.000000    ,   3.250000    ,
            3.500000    ,   3.750000    ,   4.000000    ,   4.250000    ,   4.500000    ,
            4.750000    ,   5.000000    , 479*0.0000000E+00,
X2      =   1.000000    ,
X3      =   1.000000    ,
X4      =   1.000000    ,
IPLT    =           1,
XT      = 'X                                  ',
YT      = 'Y                                  '
$END
```

```
<FWDSOL>:      CURVE 2 (same as FOR010 in Anderson, 1982a, p.32 and p.34)

  I     X(I)             Y(I)

  1   0.00000000E+00   0.70000000E+01
  2   0.25000000E+00   0.78092570E+01
  3   0.50000000E+00   0.83800678E+01
  4   0.75000000E+00   0.82929296E+01
  5   0.10000000E+01   0.73444667E+01
  6   0.12500000E+01   0.55816674E+01
  7   0.15000000E+01   0.32782612E+01
  8   0.17500000E+01   0.85826075E+00
  9   0.20000000E+01  -0.12151983E+01
 10   0.22500000E+01  -0.25589752E+01
 11   0.25000000E+01  -0.29681871E+01
 12   0.27500000E+01  -0.24698441E+01
 13   0.30000000E+01  -0.13189396E+01
 14   0.32500000E+01   0.61875194E-01
 15   0.35000000E+01   0.11841307E+01
 16   0.37500000E+01   0.16118566E+01
 17   0.40000000E+01   0.10673268E+01
 18   0.42500000E+01  -0.50171608E+00
 19   0.45000000E+01  -0.29098721E+01
 20   0.47500000E+01  -0.57799439E+01
 21   0.50000000E+01  -0.86356573E+01
```

Appendix 3.--Description of output plot file

When $FWD parameter IPLT=1, an output ASCII file is written to logical device FOR012 (file is called FOR012.DAT unless otherwise assigned). The general format of this file can be used by a separate program to produce x-y plots with appropriate axis labels, etc. The easiest way to describe the format is by listing the example FOR012 file generated in the input/output problem listed in Appendix 2.

FOR012

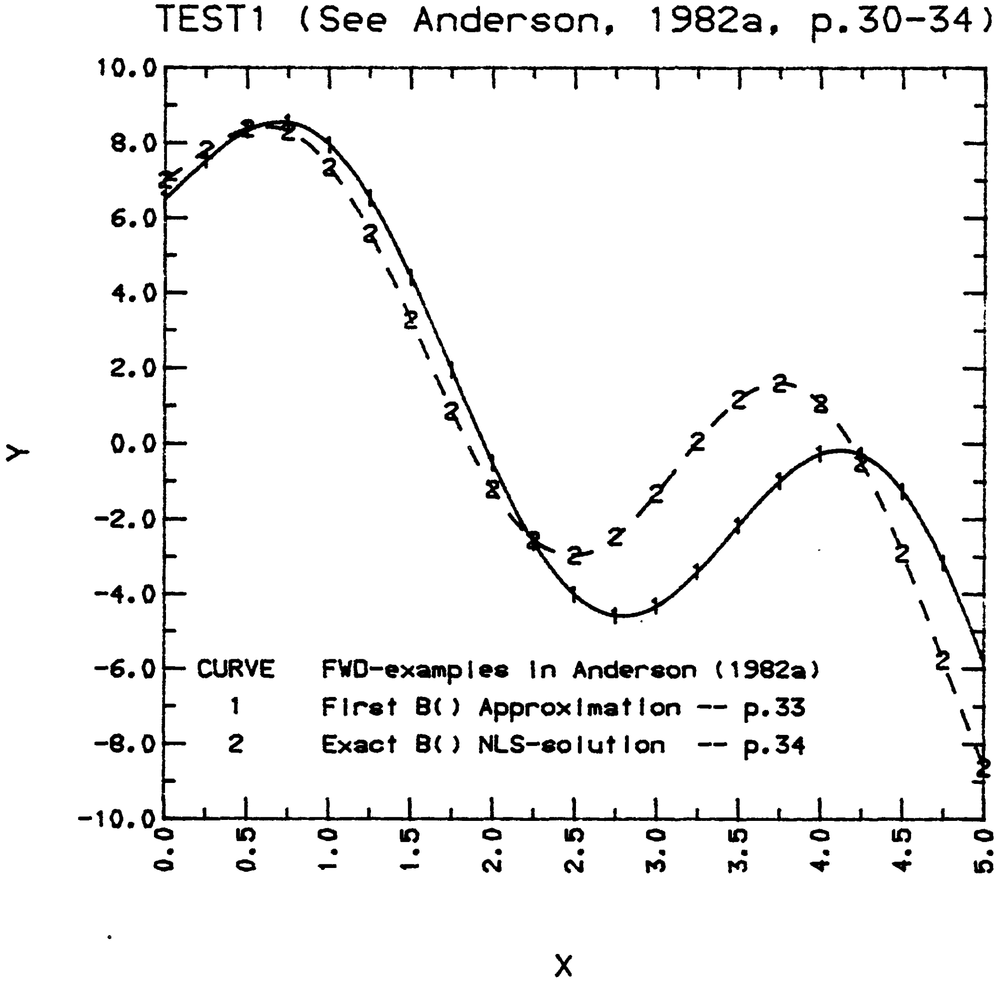
```
3
X
Y
TEST1 (See Anderson, 1982a, p.30-34)
          21
  0.00000000E+00   6.5000000
  0.25000000       7.5384965
  0.50000000       8.3349266
  0.75000000       8.5360880
   1.0000000       7.9365478
   1.2500000       6.5113988
   1.5000000       4.4146585
   1.7500000       1.9433299
   2.0000000     -0.52597934
   2.2500000      -2.6154525
   2.5000000      -4.0240407
   2.7500000      -4.5896087
   3.0000000      -4.3230004
   3.2500000      -3.4073052
   3.5000000      -2.1625590
   3.7500000     -0.98294276
   4.0000000     -0.25905442
   4.2500000     -0.30072534
   4.5000000      -1.2758222
   4.7500000      -3.1772103
   5.0000000      -5.8244791
4
X
Y
CURVE 2 (same as FOR010 in Anderson,   -
1982a, p.32 and p.34)
          21
  0.00000000E+00   7.0000000
  0.25000000       7.8092570
  0.50000000       8.3800678
  0.75000000       8.2929296
   1.0000000       7.3444667
   1.2500000       5.5816674
   1.5000000       3.2782612
   1.7500000      0.85826075
   2.0000000      -1.2151983
   2.2500000      -2.5589752
   2.5000000      -2.9681871
   2.7500000      -2.4698441
   3.0000000      -1.3189396
   3.2500000      0.61875194E-01
   3.5000000       1.1841307
   3.7500000       1.6118566
   4.0000000       1.0673268
   4.2500000     -0.50171608
   4.5000000      -2.9098721
   4.7500000      -5.7799439
   5.0000000      -8.6356573
0
001
```

The first record in FOR012 is an integer in column 1 (a "3" in this case), which indicates the number of title lines (max. 40 characters each) that immediately follow; the first title line is any desired x-axis title (via $FWD parameter XT), the second title line is any desired y-axis title (via $FWD parameter YT), and the third is the heading used in the input file FOR005 and list file FOR016 (see Appendix 2 example). Since FOR012 is an ASCII file, any VAX editor can be used to change these plot title lines later if desired. Following the last heading line (note CURVE 2 has 4 title lines), the number of data points N is given in list integer format, and is followed by the array points (X(I),Y(I),I=1,N) in the format (2G16.8). Subsequent curves are appended in the exact format. Finally, consecutive records containing "0" and "001" terminate file FOR012, which means a pseudo end-of-file is flagged.

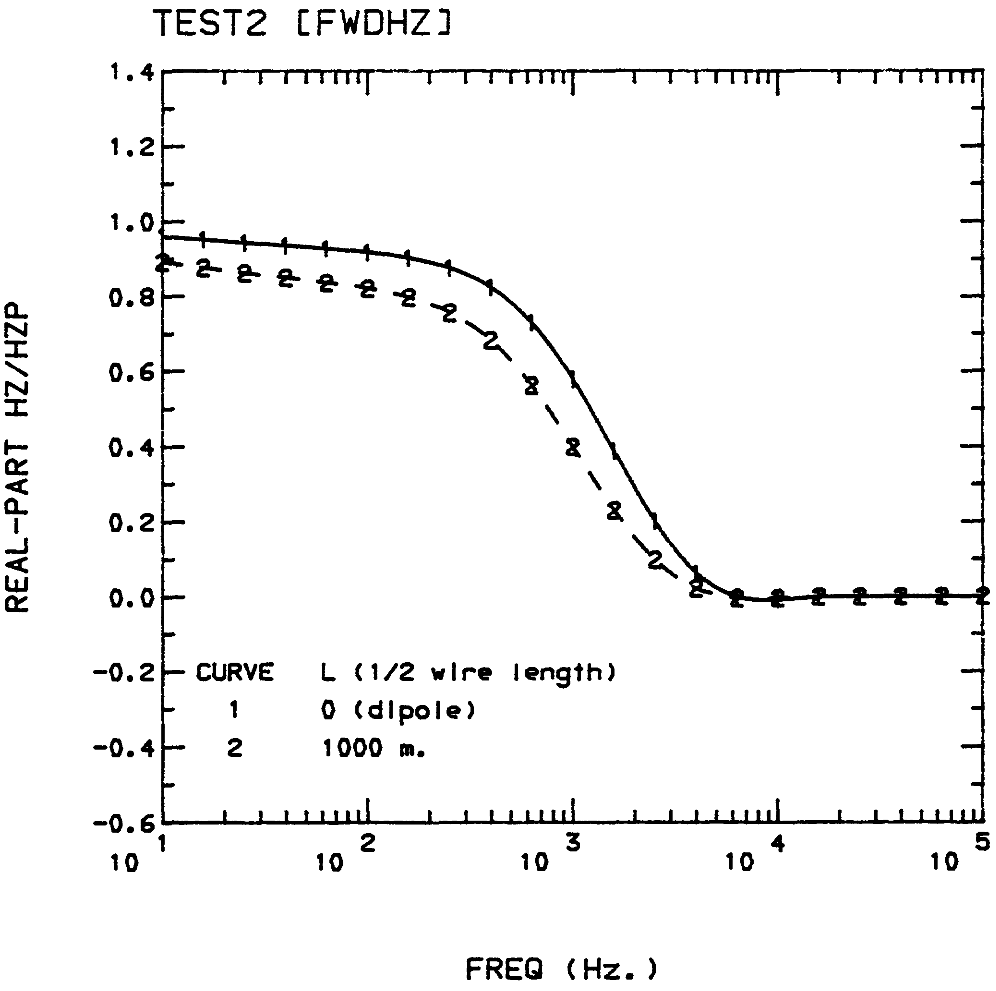
Some sounding curve example plots

The attached plots were produced (after using IPLT=1) on the USGS VAX-11/780 system and an HP-pen plotter device. The corresponding input parameter file FOR005 used is listed after each plot. Several FWD-programs from Table 1, plus the TEST1 example in Appendix 2, were run for this appendix by varying the model parameters as noted in the following table:

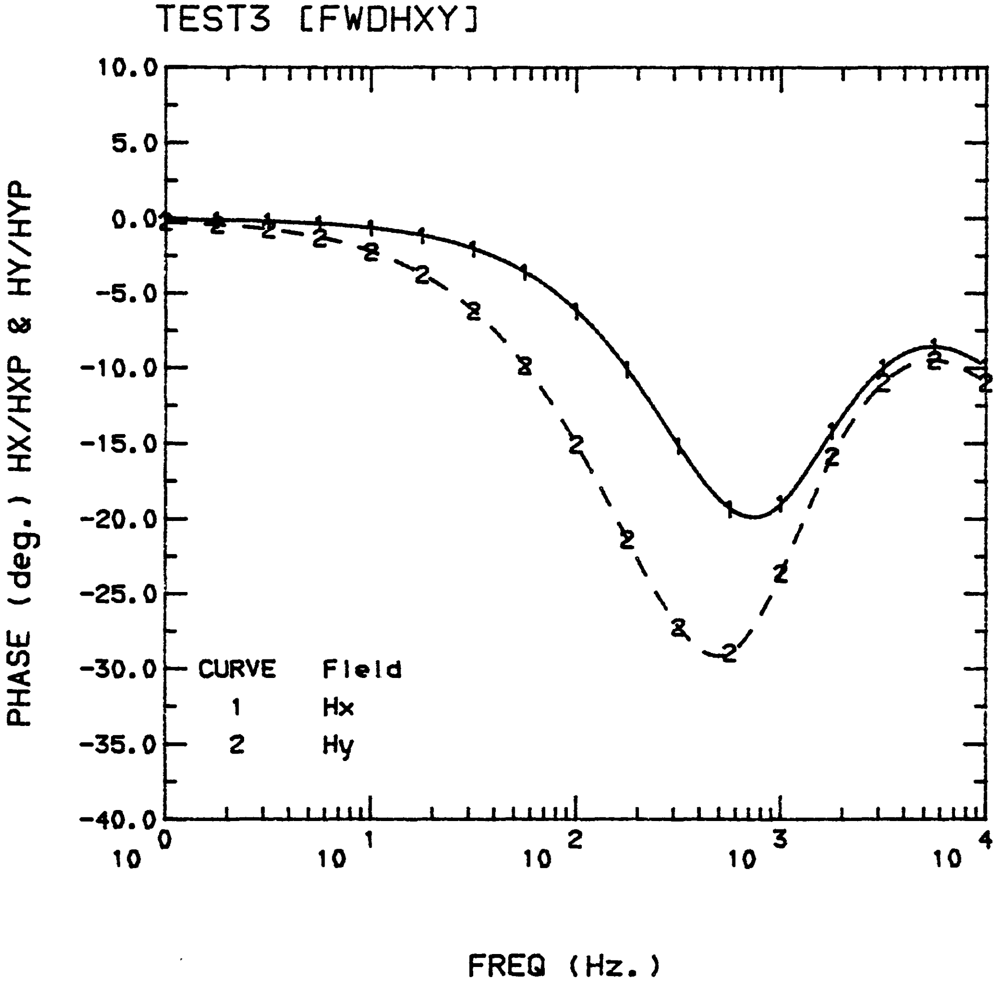
TEST	FWD-program	MM	VARIED	REFERENCE
1	T1FWDSOL	7	B()	Appendix 2 (TEST1)
2	FWDHZ	2	L	Table 1
3	FWDHXY	3	Field	Table 1
4	FWDEXY	2	H(1)	Table 1
5	FWDLOOP3	3	X2	Table 1
6	FWDTHC	2	SIG(2)	Table 1
7	FWDTCI	3	H()	Table 1
8	FWDTCO	2	A	Table 1
9	FWDDCLAG	5	RHO()	Table 1
10	FWDPW	4	RHO(),H()	Table 1



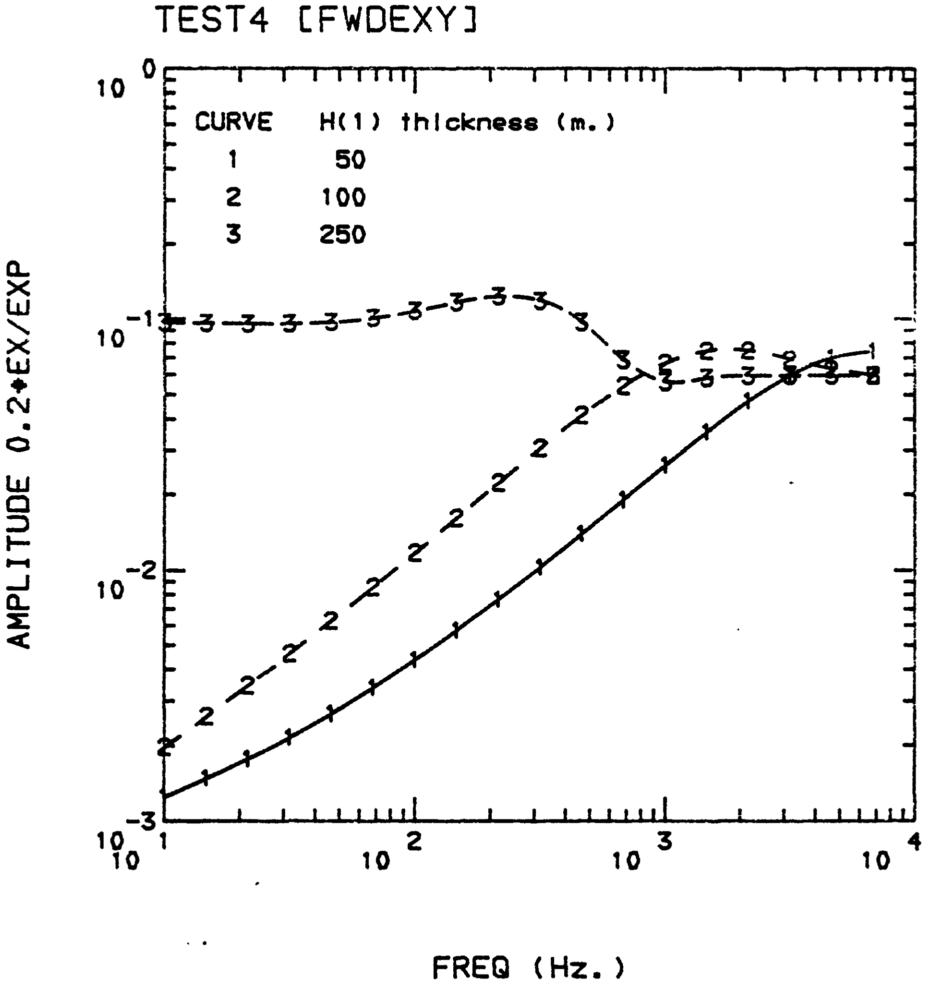
```
TEST1 (See Anderson, 1982a, p.30-34)
$FWD MM=7,MODE=0,B=1,.75,-.5,4.5,1.8,5.5,.87,
NX=-21,XNX=0,.25,.5,.75,1,1.25,1.5,1.75,2,2.25,2.5,
2.75,3,3.25,3.5,3.75,4,4.25,4.5,4.75,5, IPLT=1$
$INIT NULL=0$
CURVE 2 (same as FOR010 in Anderson, 1982a, p.32 and p.34)
$FWD B=2*1,-.5,4,2,6,1$
$INIT $
```



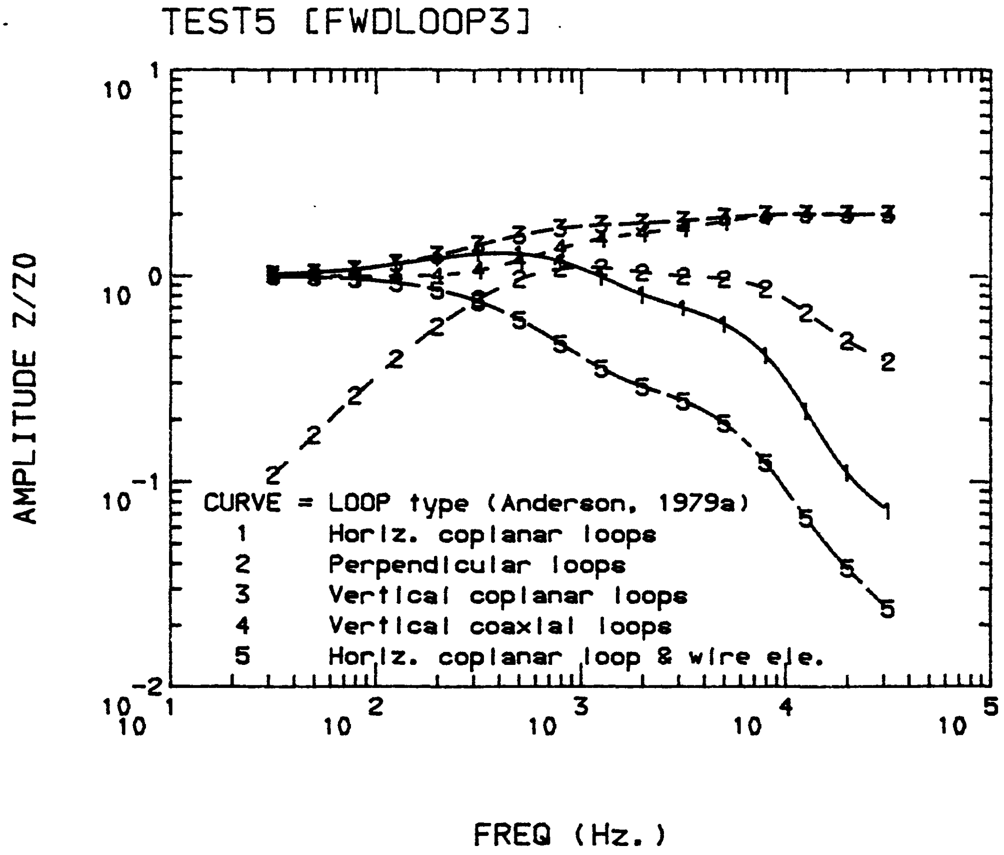
```
TEST2 [FWDHZ]
$FWD MM=2,MODE=1,NX=5,X1=10,XM=1.E5,
IPLT=1,XT='FREQ (Hz.)',YT='REAL-PART HZ/HZP',
SIG=.02,2, H=200$
$INIT MM=2,L=0,Y0=200,IOB=3$
CURVE 2
$FWD $
$INIT L=1000$
```



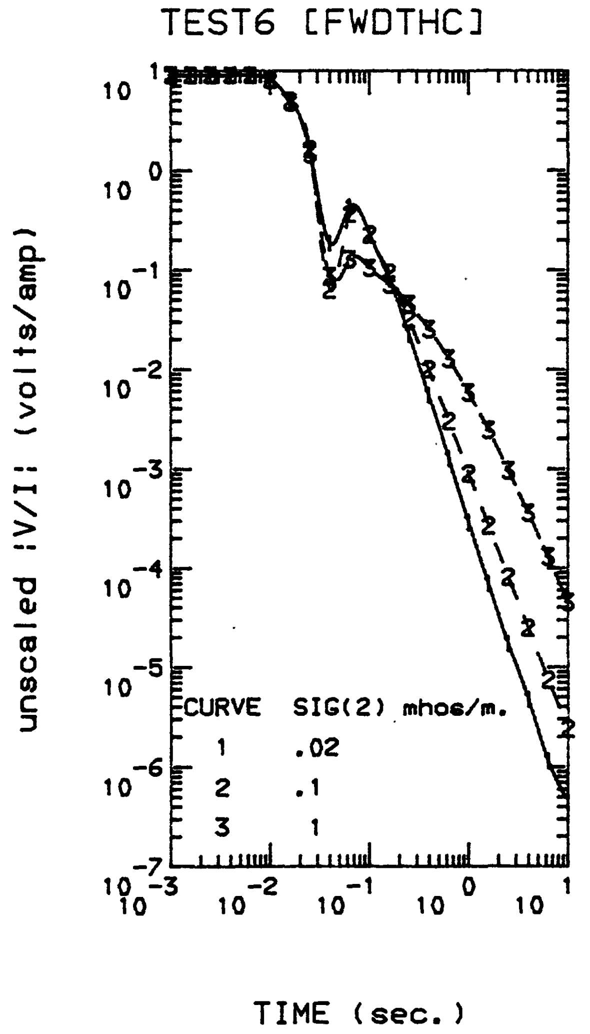
```
TEST3 [FWDHXY]
$FWD MM=3,MODE=1, B(7)=1,
NX=4,X1=1,XM=1.E4,
IPLT=1,XT='FREQ (Hz.)',YT='PHASE (deg.) HX/HXP & HY/HYP',
SIG=.001,.3,.001, H=75,10$
$INIT MM=3,X0=400,Y0=100,IOB=2$
CURV 2
$FWD $
$INIT IOB=-2$
```



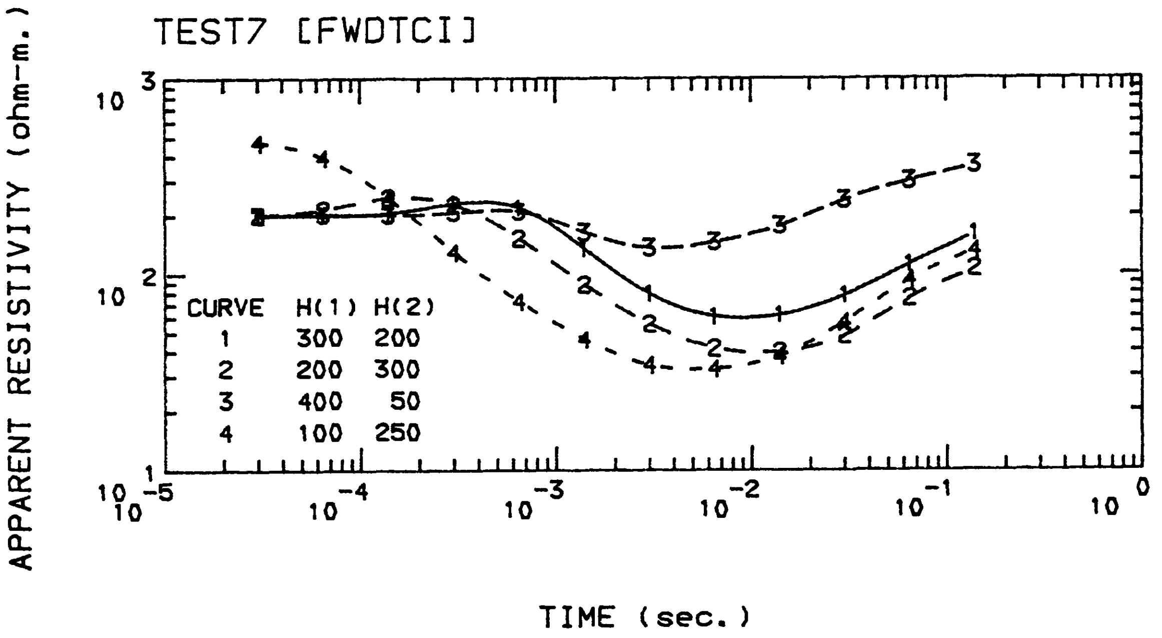
```
TEST4 [FWDEXY]
$FWD MM=2,B(5)=1,MODE=1, X2=1,
NX=6,X1=10,XM=7000, SHIFT=.2,
IPLT=1,XT='FREQ (Hz.)',YT='AMPLITUDE 0.2*EX/EXP',
SIG=.02,2, H=50$
$INIT MM=2,L=0,X0=600,Y0=500,IOB=5,EPS=.1E-5$
CURV 2
$FWD H=100$
$INIT $
CURV 3
$FWD H=250$
$INIT $
```



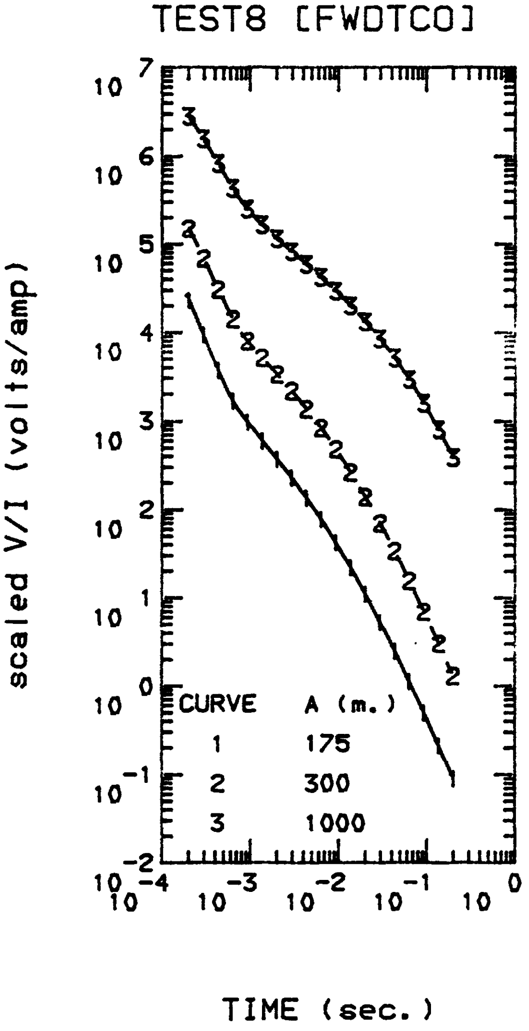
```
TEST5 [FWDLOOP3]
$FWD MM=3,MODE=1, X3=1,X2=1,NX=5,X1=31.5,XM=31500,
IPLT=1,XT='FREQ (Hz.)',YT='AMPLITUDE Z/ZO',
SIG=  .001,.01,.0001,H= 250,100$
$INIT MM=3,IOB=1,YO=1000$
C2: X2=2
$FWD X2=2$
$INIT $
C3: X2=3
$FWD X2=3$
$INIT $
C4: X2=4
$FWD X2=4$
$INIT $
C5: X2=5
$FWD X2=5$
$INIT $
```



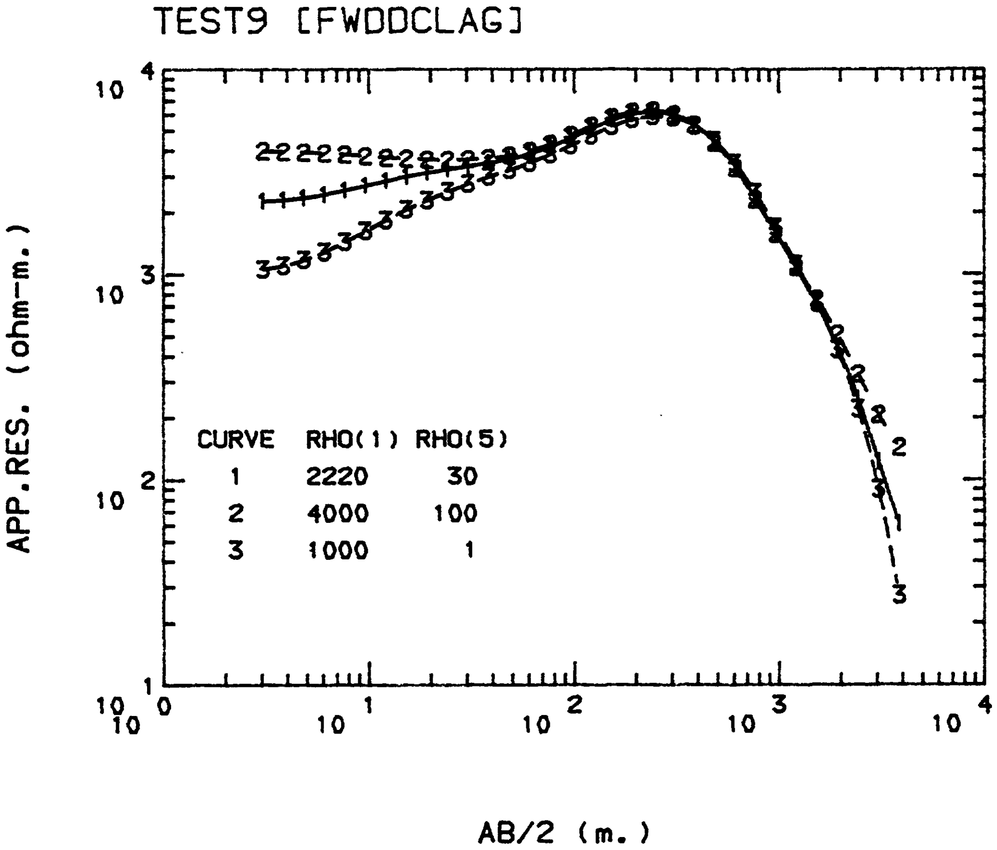
```
TEST6 [FWDTHC]
$FWD MM=2,SHIFT=1,MODE=1,SIG=.2,.02,H=500,
X1=.1E-2,NX=5,XM=10,
IPLT=1,XT='TIME (sec.)',YT='unscaled |V/I| (volts/amp)'$
$INIT IHALF=1,MM=2,Y0=1000,IOPT=0$
curve 2
$FWD SIG(2)=.1$
$INIT $
curve 3
$FWD SIG(2)=1$
$INIT $
```



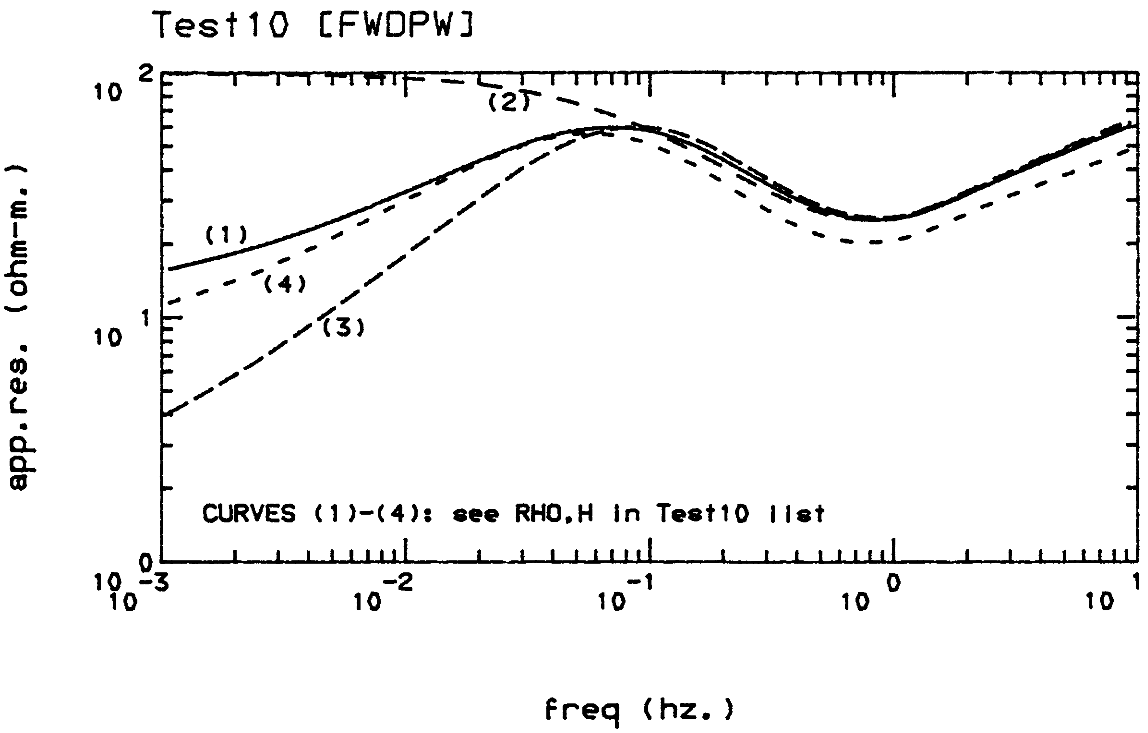
```
TEST7 [FWDTCI]
$FWD MM=3,SHIFT=1,MODE=1,SIG=.005,.05,.002,H=300,200,
X1=30E-6,NX=3,XM=160E-3,
IPLT=1,XT='TIME (sec.)',YT='APPARENT RESISTIVITY (ohm-m.)',$
$INIT MM=3,A=250,IOPT=1,EPS=.1E-9,NB=8,B0=.01,BM=100$
2:
$FWD H=200,300$$
$INIT EPS=.1E-6,NB=5$
3:
$FWD H=400,50$
$INIT $
4:
$FWD H=100,250$
$INIT $
```



```
TEST8 [FWDTCO]
$FWD MM=2,SHIFT=10000,MODE=1,SIG=.01,.2,H=200,
X1=.2E-3,NX=6,XM=.2,
IPLT=1,XT='TIME (sec.)',YT='scaled V/I (volts/amp)'$
$INIT MM=2,A=175,IOPT=0$
C 2:
$FWD $
$INIT A=300$
C 3:
$FWD $
$INIT A=1000$
```



```
TEST9 [FWDDCLAG]
$FWD MM=5,MODE=-1,NX=10,X1=3.05,XM=4000,
IPLT=1,XT='AB/2 (m.)',YT='APP.RES. (ohm-m.)',
RHO=    2220.0000     ,
     3497.6558    ,
     65202.355    ,
     1140.1653    ,
     30.000000    ,
H=      3.9603844     ,
     57.832748    ,
     20.665478    ,
     770.22137    $
$INIT MM=5$
CURVE 2
$FWD RHO(1)=4000,RHO(5)=100$
$INIT $
CURVE 3
$FWD RHO(1)=1000,RHO(5)=1$
$INIT $
```



```
Test10 [FWDPW]
$fwd mm=4,mode=-1,nx=3,x1=.001,xm=10, x2=1,
iplt=1,xt='freq (hz.)',yt='app.res. (ohm-m.)',
rho=300,20,1000,10,h=500,2500,10000$
$init mm=4,iob=5$
curve (2)
$fwd rho=1000,
rho(4)=100$
$init $
curve (3)
$fwd rho=500,
rho(4)=1$
$init $
the ans curve (4)
$fwd
rho=360,17,600,5.7, h=420,2400,.118e5$
$init $
```

Appendix 4.-- Source code availability and listing

Source Code Availability

The current version of the source code may be obtained by writing directly to the author*, and enclosing a magnetic tape to be copied and returned. This method of releasing the source code was selected in order to satisfy requests for the latest (e.g., possibly updated) version. Unless otherwise requested, the magnetic tape will be recorded in the following mode:

> Industry compatible: 9-track, standard ANSI-labeled, ASCII-mode, odd-parity, 800-bpi density, 80-character card-image records (blocked 50-card images, or 4000-characters, per‾physical block), and contained on one-file named "FWDSOL.VAX".

Note that the MAIN program for TEST1 (T1FWDSOL) and subprograms T1FCODE, T1SUBZ appear at the end of this file, following the double-precison subprogram DFWDSOL.

------

* present address is:

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

Source Listing

The attached subprograms are listed in the following order:

```
      SUBROUTINE FWDSOL(FCODE,SUBZ)                                     00000010
C                                                                       00000020
C <FWDSOL>: GENERAL FORWARD MODELING SOLUTION & PLOT FILE12  <12/13/83> 00000030
C          USING NLS (NONLINEAR LEAST SQUARES) INVERSION SUBPROGRAMS    00000040
C          FCODE AND SUBZ--EXACTLY AS REQUIRED AND CODED FOR THE        00000050
C          ADAPTIVE NONLINEAR LEAST-SQUARES ALGORITHM "NLSOL" (REF1&2)  00000060
C          BUT AS USED IN NLSOL2 (VAX NAMELIST VERSION).                00000070
C                                                                       00000080
C** THIS IS AN INTERFACE ROUTINE WRITTEN FOR THE VAX-11/780 BY          00000090
C   W.L.ANDERSON, U.S.GEOLOGICAL SURVEY, DENVER, COLORADO.              00000100
C                                                                       00000110
C   THE USER ONLY NEEDS TO WRITE SUBROUTINES FCODE AND SUBZ             00000120
C   EXACTLY AS USED IN SUBROUTINE "NLSOL" FOR INVERSE SOLUTIONS         00000130
C   (SEE DETAILS BELOW OR SEE REF2 BELOW).                              00000140
C                                                                       00000150
C** REF1:  DENNIS, J.E., ET AL, 1979, AN ADAPTIVE NONLINEAR LEAST-      00000160
C          SQUARES ALGORITHM, NTIS REPORT AD-A079-716.                  00000170
C                                                                       00000180
C   REF2:  ANDERSON, W.L., 1982, ADAPTIVE NONLINEAR LEAST-SQUARES       00000190
C          FOR CONSTRAINED OR UNCONSTRAINED MINIMIZATION PROBLEMS:      00000200
C          USGS OPEN-FILE REPT. 82-68, 65 P.                            00000210
C                                                                       00000220
C***********************************************************************00000230
C                                                                       00000240
C****  THE USER MUST DECLARE THE CALLING PARAMETERS AS EXTERNAL IN THE  00000250
C      CALLING PROGRAM (ANY DESIRED NAMES MAY BE USED).                 00000260
C E.G.,                                                                 00000270
C                                                                       00000280
C [MAIN]:                                                               00000290
C     EXTERNAL MY_FCODE,MY_SUBZ                                         00000300
C     CALL FWDSOL(MY_FCODE,MY_SUBZ)                                     00000310
C     STOP   !<OR USE>: CALL EXIT                                       00000320
C     END                                                               00000330
C [FCODE]:                                                              00000340
C     SUBROUTINE MY_FCODE(Y,X,B,W,F,IN,IDER)                            00000350
C     USER WRITTEN TO EVALUATE THE NONLINEAR OBJECTIVE FUNCTION (F)     00000360
C     USED IN NLSOL AS THE WEIGHTED SUM OF (Y(IN)-F)**2, WHERE          00000370
```

```
C     Y= OBSERVED DEPENDENT VARIABLE ARRAY (DIM. N, WHERE N IS
C      GIVEN IN $PARMS NAMELIST INPUT FOR NLSOL). [FOR FWDSOL,
C               N IS DETERMINED AUTOMATICALLY FROM $FWD X1,NX,XM,XNX.]
C     X= OBSERVED INDEPENDENT VARIABLE ARRAY (DIM. N,M, WHERE
C      M IS IN $PARMS INPUT FOR NLSOL). [FOR FWDSOL, M IS <=4
C               AND IS GIVEN BY $FWD X1,X2,X3,X4, AS REQUIRED.]
C     B= CURRENT PARAMETER ESTIMATES (DIM. K, WHERE
C      K IS IN $PARMS INPUT FOR NLSOL). [FOR FWDSOL, K IS THE
C               SAME AS $FWD MM.]
C     W= WORK ARRAY (DIM. 5)--MAY BE USED TO PASS DATA TO PCODE.
C     F= (OUTPUT) THE FUNCTION VALUE EVALUATED FOR THE GIVEN
C      Y,X, AND B ARRAYS AT THE OBSERVATION NO. 'IN'.
C     IN= (INPUT) OBSERVATION NO. TO EVALUATE F (1.LE.IN.LE.N),
C      WHICH IS CONTROLLED EXTERNALLY BY 'NLSOL'. USUALLY,
C      IN=1,2,...,N--BUT NOT ALWAYS.
C     IDER= 0 IF ANALYTICAL DERIVATIVES ARE USED (PCODE CALLED
C        AFTER FCODE).
C         = 1 IF ESTIMATED DERIVATIVES ARE USED (PCODE NOT CALLED
C        AFTER FCODE).
C                 [FOR FWDSOL, IDER=1 IS ALWAYS REQUIRED.]
C     DIMENSION Y(1),X(500,5),B(1),W(5)
C>>>>> INSERT USER CODE HERE TO EVALUATE F  <<<<<
C     END
C [SUBZ]:
C     SUBROUTINE MY_SUBZ(Y,X,B,W,NW,N,TITLE,IOUT)
C     USER WRITTEN INITIALIZATION ROUTINE (CALLED ONCE BY 'NLSOL').
C     SUBZ MAY BE USED TO CHECK Y(IN),X(IN,M) AFTER INPUT VIA
C     OBJECT (RUN)-TIME INPUT (SEE BELOW) ON UNIT IALT. ALSO, SUBZ
C     MAY BE USED TO READ ADDITIONAL $INIT PARAMETERS, AND TO LOAD
C      ANY COMMON BLOCKS IF NEEDED IN THE USERS FCODE,PCODE.
C     Y,X,B,W  ARE THE SAME AS USED IN FCODE (SEE ABOVE).
C     NW= USE ANY DUMMY INTEGER VARIABLE (THIS IS
C      TO MAINTAIN COMPATIBILITY WITH 'MARQRT' OR 'IMSLMQ').
C     N= NO. OF OBSERVATIONS IN Y(N),X(N,M) ARRAYS, WHERE
C      K.GE.N.LE.500 (N,M,K  ARE IN $PARMS INPUT FOR NLSOL).
C               [FOR FWDSOL, USE A DUMMY INTEGER VARIABLE NAME, NN.]
C     TITLE= (INPUT) 80-CHARACTER HEADING (SEE INPUT FOR005 BELOW).
C     IOUT= 1 IF TO WRITE OUTPUT ON BOTH FOR006 AND FOR016.
C         = 0 IF TO WRITE OUTPUT ONLY ON FOR006.
C     DIMENSION Y(1),X(500,5),B(1),W(5)
C     CHARACTER*80 TITLE
C>>>>> INSERT USER CODE HERE FOR ANY INITIALIZATION DESIRED  <<<<<
C
C>>>>>[NOTE: SUBZ MUST USE NAMELIST/INIT/ WITH READ(99,INIT) TO INPUT
C            ANY DESIRED PARAMETERS;  THIS IS COMPATIBLE WITH THE NEW
C            INVERSE SUBPROGRAM NLSOL2--WHICH USES CALL PRENAM(5,99),
C            WHERE FOR099 IS A SCRATCH FILE DURING RUN.]
C     END
C
C***********************************************************************
C
C** INPUT ORDER ON FOR005 (PARAMETER FILE LOGICAL NAME):
C
C 1.  TITLE (MAX. 80-CHARACTERS--ALWAYS READ BEFORE $FWD INPUT).
C 2.  $FWD ---$END (SEE DOCUMENTATION ON SUBR "FWDSOL")
```

```
C 3.  $INIT OPTIONAL NAMELIST USED FOR READING PROBLEM-DEPENDENT
C     PARAMETERS USED IN SUBROUTINE SUBZ (SEE ABOVE).
C 4.  OPTIONALLY, REPEAT STEPS 1-3 FOR EACH SUCCESSIVE FORWARD PROBLEM
C     DESIRED. (USE $FWD OVERRIDES, ETC.)
C
C OUTPUT IS GIVEN ON FOR006 (ON-LINE USUALLY) AND ON FOR016 (PRINT FILE)
C   ALSO, IF IPLT>0, PLOT12 OUTPUT ON FOR012 LOGICAL NAME.
C
C** TO RUN ON VAX (ELIMINATE <> DELIMITERS IN SUBSTITUTIONS):
C
C   $ASSIGN <PARAMETER FILE NAME> FOR005
C   [$ASSIGN <PLOT FILE NAME IF IPLT>0> FOR012]  !OPTIONAL
C   $RUN <MAIN NAME>
C
C [NOTE: FWDSOL USES SCRATCH UNITS FOR098 AND FOR099 DURING RUN.]
C
C***********************************************************************
C SPECIAL CASE TO INCLUDE FOR012.OBS (IF IT EXISTS) IN IPLT>0 OPTION.
C THIS OPTION IS AVAILABLE ONLY ON THE USGS VAX-11 (SEE W.ANDERSON).
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C$$ CHANGE THE FOLLOWING FORTRAN-77 PARAMETER STATEMENT ONLY IF
C$$ INCREASING THE DEFAULT DIMENSIONS FOR FWDSOL:
      PARAMETER (NDIM=500,MDIM=5,KDIM=20)
C$$ WHERE NDIM=MAX.OBS., MDIM=MAX.INDEP.VARS., KDIM=MAX.UNKNOWN PARMS.
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C
      CHARACTER*80 TITLE
      CHARACTER*40 TITLES(4),XT,YT
      CHARACTER*132 LINE
      DIMENSION B(KDIM),X(NDIM,MDIM),Y(NDIM),W(5),XNX(NDIM),
     1 SIG(10),RHO(10),H(9),
     2 YFWD(NDIM),XVAR(NDIM)
      EQUIVALENCE (TITLES(1),XT),(TITLES(2),YT)
C**
      NAMELIST/FWD/MM,MODE,SIG,RHO,H,SHIFT,B,
     1 X1,NX,XM,XNX,X2,X3,X4,IPLT,XT,YT
      NAMELIST/FWDO/MM,MODE,B,
     1 X1,NX,XM,XNX,X2,X3,X4,IPLT,XT,YT
      DATA MODE/1/,IPLT/0/,XT/'X'/,YT/'Y'/,SHIFT/1.0/,
     1 X2/1.0/,X3/1.0/,X4/1.0/
C**
      DO I=1,NDIM
        Y(I)=1.0
        X(I,1)=FLOAT(I)
      ENDDO
      CALL PRENAM(5,99)
C
C** READ FWDSOL TITLE LINE (RETURN FOR EACH SUCCESSIVE FWD PROBLEM)
C
10    READ(99,20,END=90) TITLE
20    FORMAT(A80)
C
C** READ $FWD, TEST, ETC.
C
      READ(99,FWD,END=900)
      M=IABS(MM)
      M2=M*2
      IF((M.LT.1.OR.M.GT.10).AND.MODE.NE.0)
     1 CALL ERRMSG('$FWD |MM|<1 OR >10 & MODE NOT 0',0,6,16)
      IF(NX.EQ.0)CALL ERRMSG('$FWD NX=0',0,6,16)
      IF(MODE.GT.0) THEN
        DO I=1,M
          IF(SIG(I).LE.0.0)CALL ERRMSG(
     1    '$FWD MODE>0 AND SIG(I)<=0 FOR SOME I<=|MM|',0,6,16)
          B(I)=SIG(I)
          RHO(I)=1.0/SIG(I)
          IF(I.LT.M) THEN
            IF(H(I).LE.0.0)CALL ERRMSG(
     1      '$FWD MODE>0 AND H(I)<=0 FOR SOME I<|MM|',0,6,16)
            B(M+I)=H(I)
          ENDIF
        ENDDO
      ELSE IF(MODE.LT.0) THEN
        DO I=1,M
          IF(RHO(I).LE.0.0)CALL ERRMSG(
     1    '$FWD MODE<0 AND RHO(I)<=0 FOR SOME I<=|MM|',0,6,16)
          SIG(I)=1.0/RHO(I)
          B(I)=RHO(I)
          IF(I.LT.M) THEN
            IF(H(I).LE.0.0)CALL ERRMSG(
     1      '$FWD MODE<0 AND H(I)<=0 FOR SOME I<|MM|',0,6,16)
            B(M+I)=H(I)
          ENDIF
        ENDDO
      ELSE  !MODE=0
        IF(M.LT.1.OR.M.GT.KDIM)CALL ERRMSG(
     1 '$FWD |MM|<1 OR >20 WHEN MODE=0',0,6,16)
      ENDIF  !END OF IF(MODE...) THEN ABOVE
      IF(MODE.NE.0) B(M2)=SHIFT
C
C  DETERMINE N (NO.OBS. FOR FWD SOLUTION)
C
      IF(NX.LT.0) THEN
        N=IABS(NX)
        IF(N.GT.NDIM)CALL ERRMSG(
     1 '$FWD NX<500',0,6,16)
        DO I=1,N
          X(I,1)=XNX(I)
        ENDDO
      ELSE  !NX>0
        DX=EXP(2.30258509/FLOAT(NX))
        XMTEST=0.5*(XM+XM*DX)
        N=0
        XX=X1/DX
30      XX=XX*DX
        IF(XX.GE.XMTEST) GO TO 40
        N=N+1
        X(N,1)=XX
        GO TO 30
      ENDIF
```

```
40    CALL NONBLANK(TITLE,NB)
      IF(NB.EQ.0) NB=1
C
C  FILL IN REST OF DATA MATRIX VIA X2,X3,X4
C
      DO I=1,N
        X(I,2)=X2
        X(I,3)=X3
        X(I,4)=X4
      ENDDO
C
C  INITIALIZE VIA CALL SUBZ (READ $INIT AND TEST, LOAD COMMON, ETC.)
C
      CALL SUBZ(Y,X,B,W,NPRNT,N,TITLE,1)
C     ******************************
C
C  WRITE $FWD ON FOR006 AND FOR016
C
C//   WRITE(6,42) TITLE
42    FORMAT(////' <FWDSOL>:',5X,A<NB>/2X,'-------'/)
      WRITE(16,42) TITLE
      OPEN(UNIT=98,STATUS='SCRATCH')
      IF(MODE.NE.0) THEN
        WRITE(98,FWD)
      ELSE
        WRITE(98,FWD0)
      ENDIF
C--REFMT WRITE(98,NAMELIST) TO UNIT=6 AND 16 TO BREAK OUT ARRAY LISTS
      REWIND 98
9910  READ(98,9920,END=9940) LINE
9920  FORMAT(A)
      I=INDEX(LINE,'$')
      IF(I.NE.0) GO TO 9930
      I=INDEX(LINE,'=')
      IF(I.NE.0) GO TO 9930
      LINE(11:)=LINE
      LINE(1:10)=' '
9930  CALL NONBLANK(LINE,I)
      IF(I.EQ.0) I=1
C//   WRITE(6,9920) LINE(1:I)
      WRITE(16,9920) LINE(1:I)
      GO TO 9910
9940  CLOSE(UNIT=98)
C//   WRITE(6,44) TITLE
44    FORMAT('1<FWDSOL>:',5X,A<NB>//3X,'I',4X,'X(I)',12X,'Y(I)'/)
      WRITE(16,44) TITLE
      IF(IPLT.GT.0) OPEN(UNIT=12,STATUS='NEW',CARRIAGECONTROL='LIST')
C
C  NOW GET THE FORWARD SOLUTION VIA FCODE FOR I=1,N
C
      DO I=1,N
          CALL FCODE(Y,X,B,W,F,I,1)
C         *************************
C//       WRITE(6,50) I,X(I,1),F
50        FORMAT(1X,I3,2E16.8)
```

```
          WRITE(16,50) I,X(I,1),F
          YFWD(I)=F
          XVAR(I)=X(I,1)
      ENDDO
      IF(IPLT.EQ.0) GO TO 10
C
C  PREPARE FOR012 FOR PLOT12 OUTPUT (IPLT>0)
C
      TITLES(3)=TITLE(1:40)
      TITLES(4)=TITLE(41:80)
      CALL NONBLANK(TITLES(4),I)
      IF(I.EQ.0) THEN
        NTITLES=3
      ELSE
        NTITLES=4
        IF(TITLES(3)(40:40).EQ.' ') THEN
          TITLES(3)(40:40)='-'
        ELSE
          DO I=39,2,-1
            IF(TITLES(3)(I:I).EQ.' ') THEN
              TITLES(4)=TITLE(I+1:)
              TITLES(3)(I:39)=' '
              TITLES(3)(40:40)='-'
              GO TO 60
            ENDIF
          ENDDO
        ENDIF
      ENDIF
60    WRITE(12,70) NTITLES,(TITLES(I),I=1,NTITLES)
70    FORMAT(I1/(A))
      WRITE(12,80) N,(XVAR(I),YFWD(I),I=1,N)
80    FORMAT(I/(2G16.8))
      GO TO 10
90    IF(IPLT.GT.0) THEN
        OPEN(UNIT=98,STATUS='OLD',FILE='FOR012.OBS',ERR=98)
C--SPECIAL CASE: "FOR012.OBS" EXISTS--COPY TO FOR012 AS "[OBS]"
        READ(98,70) NTITLES,(TITLES(I),I=1,NTITLES)
        READ(98,80) N,(XVAR(I),YFWD(I),I=1,N)
        CLOSE(UNIT=98)
        NTITLES=4
        TITLES(4)(36:40)='[OBS]'
        WRITE(12,70) NTITLES,(TITLES(I),I=1,NTITLES)
        WRITE(12,80) N,(XVAR(I),YFWD(I),I=1,N)
98      WRITE(12,100)
100     FORMAT('0'/'001'/' ')
      ENDIF
C
C** RETURN FROM FWDSOL (AFTER E.O.F ON FOR005)
C
      RETURN
900   CALL ERRMSG('E.O.F ON FOR005 READING $FWD INPUT ?',0,6,16)
C
C** END OF SUBROUTINE FWDSOL
C
      END
      SUBROUTINE NONBLANK(C,NB)
C--DETERMINE NON-BLANK CHAR LENGTH (=NB ON EXIT) OF C*(*)
C  NOTE THAT NB WILL BE IN [0,LEN(C)].
C
      CHARACTER*(*) C
      L=LEN(C)
      DO 10 I=L,1,-1
        NB=I
        IF(C(I:I).NE.' ') RETURN
10    CONTINUE
      NB=0
      RETURN
      END
      SUBROUTINE PRENAM(INUNIT,ITMP)
C--PRENAM CAN BE CALLED PRIOR TO READ(ITMP,NAME,...) TO SHIFT ALL
C  NAMELIST INPUT $NAME ... FROM COL.1 AND BEYOND ON INUNIT TO
C  NAMELIST INPUT $NAME ... FROM COL.2 AND BEYOND ON ITMP (UNIT=ITMP
C  IS DELETED AFTER CLOSING OR END OF PROCESS).  NOTE ITMP MAY BE
C  ANY UNIT NUMBER NOT BEING USED (BUT CANNOT BE INUNIT OR 6).
C
C--USAGE:
C
C     NAMELIST/ANYNAME/...
C     ...
C     CALL PRENAM(5,1)
C     ...
C     READ(1,ANYNAME,END=99,ERR=999)
C     ...
C
C--NOTE:  BECAUSE EARLIER VERSIONS (<3.0) OF VAX-11 FORTRAN-77
C     DID NOT HAVE NAMELIST AVAILABLE, A SIMULATED CALL NAMELIST WAS
C     USED BY MANY USERS.  IN PARTICULAR, W.L.ANDERSON USED A
C     SIMULATED CALL NAMELIST(INUNIT,'$ANYNAM',*99) SUBROUTINE WHICH
C     COULD CONTAIN $ANYNAM LISTS BEGINNING IN COL.1 TO 80;  BUT
C     SINCE VERSION 3.0 OF VAX-11 FORTRAN-77 REQUIRES THE INPUT
C     $ANYNAM LIST TO BEGIN IN COL.2 OR BEYOND, SUBROUTINE PRENAM
C     CAN BE USED ONCE TO MEET THIS REQUIREMENT, AND BECOMES
C     TRANSPARENT TO THE END USER'S INPUT FILE PREPARATION (COL.1-ON)
C
      CHARACTER*200 C
      IF(ITMP.EQ.6.OR.ITMP.EQ.INUNIT) CALL ERRMSG(
     1 '{PRENAM}: ITMP=6 OR ITMP=INUNIT VIOLATION',0,6,0)
      OPEN(UNIT=ITMP,STATUS='SCRATCH',FILE='PRENAM.TMP',ERR=999)
      NONAME=0
10    READ(INUNIT,20,END=99,ERR=888) C
20    FORMAT(A)
      CALL NONBLANK(C,NC)
      IF(NC.EQ.0) NC=1
      IF(NONAME.EQ.0) THEN
        I=INDEX(C,'$')
        IF(I.EQ.0) THEN
          WRITE(ITMP,30) C
30        FORMAT(A<NC>)
        ELSE
          NONAME=1
          WRITE(ITMP,40) C(I:NC)
40        FORMAT(1X,A)
          I=INDEX(C(I+1:NC),'$')
          IF(I.NE.0) NONAME=0
        ENDIF
      ELSE
        WRITE(ITMP,40) C(1:NC)
        I=INDEX(C,'$')
        IF(I.NE.0) NONAME=0
      ENDIF
      GO TO 10
99    REWIND ITMP
      RETURN
888   CALL ERRMSG('{PRENAM}: ERROR IN READING INUNIT',0,6,0)
999   CALL ERRMSG('{PRENAM}: CANNOT OPEN UNIT=ITMP',0,6,0)
      END
      SUBROUTINE ERRMSG(MSG,ISKIP,IUNIT1,IUNIT2)
C
C  GENERAL ERROR MESSAGE OUTPUT AND EXIT ON VAX-11/780
C
C  MSG*(*) = VARIABLE-LENGTH 'MESSAGE'
C  ISKIP = 0 FOR NO BLANK LINE BEFORE OUTPUT TO IUNIT1 & IUNIT2
C         > 0 FOR ONE BLANK LINE BEFORE.
C  IUNIT1 = 0 TO SUPPRESS OUTPUT ON IUNIT1 (>0 TO WRITE ON IUNIT1).
C  IUNIT2 = 0 TO SUPPRESS OUTPUT ON IUNIT2 (>0 TO WRITE ON IUNIT2).
C
C  MESSAGES ARE WRITTEN IN THE FORM:
C
C  {ERRMSG}: _MSG_HERE_
C
      CHARACTER*(*) MSG
      I=LEN(MSG)
      DO 1 J=1,2
        IF(J.EQ.1) THEN
       JUNIT=IUNIT1
          ELSE
       JUNIT=IUNIT2
        ENDIF
        IF(JUNIT.GT.0) THEN
          IF(ISKIP.EQ.0) THEN
            WRITE(JUNIT,2) MSG
            ELSE
       WRITE(JUNIT,3) MSG
          ENDIF
      ENDIF
1     CONTINUE
      CALL EXIT
2     FORMAT(1X,'{ERRMSG}: ',A<I>)
3     FORMAT(/1X,'{ERRMSG}: ',A<I>)
      END
      SUBROUTINE DFWDSOL(FCODE,SUBZ)
C
C <DFWDSOL>: GENERAL FORWARD MODELING SOLUTION & PLOT FILE12  <12/13/83>
C           USING NLS (NONLINEAR LEAST SQUARES) INVERSION SUBPROGRAMS
C           FCODE AND SUBZ--EXACTLY AS REQUIRED AND CODED FOR THE
```

```
C          ADAPTIVE NONLINEAR LEAST-SQUARES ALGORITHM "NLSOLX" (REF1&2) 00004230
C                                                                       00004240
C** THIS IS AN INTERFACE ROUTINE WRITTEN FOR THE VAX-11/780 BY          00004250
C   W.L.ANDERSON, U.S.GEOLOGICAL SURVEY, DENVER, COLORADO.              00004260
C                                                                       00004270
C   THE USER ONLY NEEDS TO WRITE SUBROUTINES FCODE AND SUBZ             00004280
C   EXACTLY AS USED IN SUBROUTINE "NLSOLX" FOR INVERSE SOLUTIONS        00004290
C   IN DOUBLE-PRECISION.                                                00004300
C   (SEE DETAILS BELOW OR SEE REF2 BELOW).                              00004310
C                                                                       00004320
C** REF1:   DENNIS, J.E., ET AL, 1979, AN ADAPTIVE NONLINEAR LEAST-     00004330
C          SQUARES ALGORITHM, NTIS REPORT AD-A079-716.                  00004340
C                                                                       00004350
C   REF2:  ANDERSON, W.L., 1982, ADAPTIVE NONLINEAR LEAST-SQUARES       00004360
C          FOR CONSTRAINED OR UNCONSTRAINED MINIMIZATION PROBLEMS:      00004370
C          USGS OPEN-FILE REPT. 82-68, 65 P.                            00004380
C                                                                       00004390
C***********************************************************************00004400
C                                                                       00004410
C****  THE USER MUST DECLARE THE CALLING PARAMETERS AS EXTERNAL IN THE  00004420
C      CALLING PROGRAM (ANY DESIRED NAMES MAY BE USED).                 00004430
C E.G.,                                                                 00004440
C                                                                       00004450
C [MAIN]:                                                               00004460
C     EXTERNAL MY_FCODE,MY_SUBZ                                         00004470
C     CALL DFWDSOL(MY_FCODE,MY_SUBZ)                                    00004480
C     STOP   !<OR USE>: CALL EXIT                                       00004490
C     END                                                               00004500
C [FCODE]:                                                              00004510
C     SUBROUTINE MY_FCODE(Y,X,B,W,F,IN,IDER)                            00004520
C     USER WRITTEN TO EVALUATE THE NONLINEAR OBJECTIVE FUNCTION (F)     00004530
C     USED IN NLSOL AS THE WEIGHTED SUM OF (Y(IN)-F)**2, WHERE          00004540
C     Y= OBSERVED DEPENDENT VARIABLE ARRAY (DIM. N, WHERE N IS          00004550
C      GIVEN IN $PARMS NAMELIST INPUT FOR NLSOLX). [FOR                 00004560
C               DFWDSOL, N IS DETERMINED AUTOMATICALLY FROM $FWD X1,    00004570
C             NX,XM,XNX.]                                               00004580
C     X= OBSERVED INDEPENDENT VARIABLE ARRAY (DIM. N,M, WHERE           00004590
C      M IS IN $PARMS INPUT FOR NLSOLX). [FOR DFWDSOL, M IS             00004600
C             <=4 AND IS GIVEN BY $FWD X1,X2,X3,X4, AS REQUIRED.]       00004610
C     B= CURRENT PARAMETER ESTIMATES (DIM. K, WHERE                     00004620
C      K IS IN $PARMS INPUT FOR NLSOLX). [FOR DFWDSOL, K IS             00004630
C               THE SAME AS $FWD MM.]                                   00004640
C     W= WORK.ARRAY (DIM. 5)--MAY BE USED TO PASS DATA TO PCODE.        00004650
C     F= (OUTPUT) THE FUNCTION VALUE EVALUATED FOR THE GIVEN            00004660
C      Y,X, AND B ARRAYS AT THE OBSERVATION NO. 'IN'.                   00004670
C     IN= (INPUT) OBSERVATION NO. TO EVALUATE F (1.LE.IN.LE.N),         00004680
C      WHICH IS CONTROLLED EXTERNALLY BY 'NLSOL'. USUALLY,              00004690
C      IN=1,2,...,N--BUT NOT ALWAYS.                                    00004700
C     IDER= 0 IF ANALYTICAL DERIVATIVES ARE USED (PCODE CALLED          00004710
C        AFTER FCODE).                                                  00004720
C         = 1 IF ESTIMATED DERIVATIVES ARE USED (PCODE NOT CALLED       00004730
C        AFTER FCODE).                                                  00004740
C               [FOR DFWDSOL, IDER=1 IS ALWAYS REQUIRED.]               00004750
C     IMPLICIT REAL*8 (A-H,O-Z)                                         00004760
C     DIMENSION Y(1),X(1000,5),B(1),W(5)                                00004770
```

```
C>>>>> INSERT USER CODE HERE TO EVALUATE F  <<<<<
C     END
C [SUBZ]:
C     SUBROUTINE MY_SUBZ(Y,X,B,W,NW,N,TITLE,IOUT)
C     USER WRITTEN INITIALIZATION ROUTINE (CALLED ONCE BY 'NLSOL').
C     SUBZ MAY BE USED TO CHECK Y(IN),X(IN,M) AFTER INPUT VIA
C     OBJECT (RUN)-TIME INPUT (SEE BELOW) ON UNIT IALT. ALSO, SUBZ
C     MAY BE USED TO READ ADDITIONAL $INIT PARAMETERS, AND TO LOAD
C      ANY COMMON BLOCKS IF NEEDED IN THE USERS FCODE,PCODE.
C     Y,X,B,W  ARE THE SAME AS USED IN FCODE (SEE ABOVE).
C     NW= USE ANY DUMMY INTEGER VARIABLE (THIS IS
C      TO MAINTAIN COMPATIBILITY WITH 'MARQRT' OR 'IMSLMQ').
C     N= NO. OF OBSERVATIONS IN Y(N),X(N,M) ARRAYS, WHERE
C      K.GE.N.LE.1000 (N,M,K  ARE IN $PARMS INPUT).
C               [FOR DFWDSOL, USE A DUMMY INTEGER VARIABLE NAME, NN.]
C     TITLE= (INPUT) 80-CHARACTER HEADING (SEE INPUT FOR005 BELOW).
C     IOUT= 1 IF TO WRITE OUTPUT ON BOTH FOR006 AND FOR016.
C         = 0 IF TO WRITE OUTPUT ONLY ON FOR006.
C     IMPLICIT REAL*8 (A-H,O-Z)
C     DIMENSION Y(1),X(1000,5),B(1),W(5)
C     CHARACTER*80 TITLE
C>>>>> INSERT USER CODE HERE FOR ANY INITIALIZATION DESIRED  <<<<<
C
C>>>>>[NOTE: SUBZ MUST USE NAMELIST/INIT/ WITH READ(99,INIT) TO INPUT
C            ANY DESIRED PARAMETERS;  THIS IS COMPATIBLE WITH THE NEW
C            INVERSE SUBPROGRAM NLSOLX2--WHICH USES CALL PRENAM(5,99),
C            WHERE FOR099 IS A SCRATCH FILE DURING RUN.]
C     END
C
C***********************************************************************
C
C** INPUT ORDER ON FOR010 (PARAMETER FILE LOGICAL NAME):
C
C 1.  TITLE (MAX. 80-CHARACTERS--ALWAYS READ BEFORE $FWD INPUT).
C 2.  $FWD ---$END (SEE DOCUMENTATION ON SUBR "FWDSOL=DFWDSOL")
C 3.  $INIT OPTIONAL NAMELIST USED FOR READING PROBLEM-DEPENDENT
C     PARAMETERS USED IN SUBROUTINE SUBZ (SEE ABOVE).
C 4.  OPTIONALLY, REPEAT STEPS 1-3 FOR EACH SUCCESSIVE FORWARD PROBLEM
C     DESIRED. (USE $FWD OVERRIDES, ETC.)
C
C OUTPUT IS GIVEN ON FOR006 (ON-LINE USUALLY) AND ON FOR016 (PRINT FILE)
C   ALSO, IF IPLT>0, PLOT12 OUTPUT ON FOR012 LOGICAL NAME.
C
C** TO RUN ON VAX (ELIMINATE <> DELIMITERS IN SUBSTITUTIONS):
C
C   $ASSIGN <PARAMETER FILE NAME> FOR005
C   [$ASSIGN <PLOT FILE NAME IF IPLT>0> FOR012]  !OPTIONAL
C   $RUN <MAIN NAME>
C
C [NOTE: DFWDSOL USES SCRATCH UNITS FOR098 AND FOR099 DURING RUN.]
C
C***********************************************************************
C SPECIAL CASE TO INCLUDE FOR012.OBS (IF IT EXISTS) IN IPLT>0 OPTION.
C THIS OPTION IS AVAILABLE ONLY ON THE USGS VAX-11 (SEE W.ANDERSON).
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
C$$ CHANGE THE FOLLOWING FORTRAN-77 PARAMETER STATEMENT ONLY IF
C$$ INCREASING THE DEFAULT DIMENSIONS FOR DFWDSOL:
      PARAMETER (NDIM=1000,MDIM=5,KDIM=100)
C$$ WHERE NDIM=MAX.OBS., MDIM=MAX.INDEP.VARS., KDIM=MAX.UNKNOWN PARMS.
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C
      IMPLICIT REAL*8 (A-H,O-Z)
      CHARACTER*80 TITLE
      CHARACTER*40 TITLES(4),XT,YT
      CHARACTER*132 LINE
      DIMENSION B(KDIM),X(NDIM,MDIM),Y(NDIM),W(5),XNX(NDIM),
     1 SIG(10),RHO(10),H(9),
     2 YFWD(NDIM),XVAR(NDIM)
      EQUIVALENCE (TITLES(1),XT),(TITLES(2),YT)
C**
      NAMELIST/FWD/MM,MODE,SIG,RHO,H,SHIFT,B,
     1 X1,NX,XM,XNX,X2,X3,X4,IPLT,XT,YT
      NAMELIST/FWDO/MM,MODE,B,
     1 X1,NX,XM,XNX,X2,X3,X4,IPLT,XT,YT
      DATA MODE/1/,IPLT/0/,XT/'X'/,YT/'Y'/,SHIFT/1.0D0/,
     1 X2/1.0D0/,X3/1.0D0/,X4/1.0D0/
C**
      DO I=1,NDIM
        Y(I)=1.0D0
        X(I,1)=DFLOAT(I)
      ENDDO
      CALL PRENAM(5,99)
C
C** READ DFWDSOL TITLE LINE (RETURN FOR EACH SUCCESSIVE FWD PROBLEM)
C
10    READ(99,20,END=90) TITLE
20    FORMAT(A80)
C
C** READ $FWD, TEST, ETC.
C
      READ(99,FWD,END=900)
      M=IABS(MM)
      M2=M*2
      IF((M.LT.1.OR.M.GT.10).AND.MODE.NE.0)
     1 CALL ERRMSG('$FWD |MM|<1 OR >10 & MODE NOT 0',0,6,16)
      IF(NX.EQ.0)CALL ERRMSG('$FWD NX=0',0,6,16)
      IF(MODE.GT.0) THEN
        DO I=1,M
          IF(SIG(I).LE.0.0D0)CALL ERRMSG(
     1   '$FWD MODE>0 AND SIG(I)<=0 FOR SOME I<=|MM|',0,6,16)
          B(I)=SIG(I)
          RHO(I)=1.0D0/SIG(I)
          IF(I.LT.M) THEN
            IF(H(I).LE.0.0D0)CALL ERRMSG(
     1       '$FWD MODE>0 AND H(I)<=0 FOR SOME I<|MM|',0,6,16)
            B(M+I)=H(I)
          ENDIF
        ENDDO
      ELSE IF(MODE.LT.0) THEN
        DO I=1,M
```

```
          IF(RHO(I).LE.0.0D0)CALL ERRMSG(
     1   '$FWD MODE<0 AND RHO(I)<=0 FOR SOME I<=|MM|',0,6,16)
          SIG(I)=1.0D0/RHO(I)
          B(I)=RHO(I)
          IF(I.LT.M) THEN
            IF(H(I).LE.0.0D0)CALL ERRMSG(
     1     '$FWD MODE<0 AND H(I)<=0 FOR SOME I<|MM|',0,6,16)
            B(M+I)=H(I)
          ENDIF
        ENDDO
      ELSE  !MODE=0
        IF(M.LT.1.OR.M.GT.KDIM)CALL ERRMSG(
     1 '$FWD |MM|<1 OR >100 WHEN MODE=0',0,6,16)
      ENDIF  !END OF IF(MODE...) THEN ABOVE
      IF(MODE.NE.0) B(M2)=SHIFT
C
C  DETERMINE N (NO.OBS. FOR FWD SOLUTION)
C
      IF(NX.LT.0) THEN
        N=IABS(NX)
        IF(N.GT.NDIM)CALL ERRMSG(
     1 '$FWD NX<1000',0,6,16)
        DO I=1,N
          X(I,1)=XNX(I)
        ENDDO
      ELSE  !NX>0
        DX=DEXP(2.302585092994046D0/DFLOAT(NX))
        XMTEST=0.5D0*(XM+XM*DX)
        N=0
        XX=X1/DX
30      XX=XX*DX
        IF(XX.GE.XMTEST) GO TO 40
        N=N+1
        X(N,1)=XX
        GO TO 30
      ENDIF
40    CALL NONBLANK(TITLE,NB)
      IF(NB.EQ.0) NB=1
C
C  FILL IN REST OF DATA MATRIX VIA X2,X3,X4
C
      DO I=1,N
        X(I,2)=X2
        X(I,3)=X3
        X(I,4)=X4
      ENDDO
C
C  INITIALIZE VIA CALL SUBZ (READ $INIT AND TEST, LOAD COMMON, ETC.)
C
      CALL SUBZ(Y,X,B,W,NPRNT,N,TITLE,1)
C     ******************************
C
C  WRITE $FWD ON FOR006 AND FOR016
C
C//   WRITE(6,42) TITLE
```

```
42    FORMAT(////' <DFWDSOL>:',5X,A<NB>/2X,'-------'/)
      WRITE(16,42) TITLE
      OPEN(UNIT=98,STATUS='SCRATCH')
      IF(MODE.NE.0) THEN
        WRITE(98,FWD)
      ELSE
        WRITE(98,FWD0)
      ENDIF
C--REFMT WRITE(98,NAMELIST) TO UNIT=6 AND 16 TO BREAK OUT ARRAY LISTS
      REWIND 98
9910  READ(98,9920,END=9940) LINE
9920  FORMAT(A)
      I=INDEX(LINE,'$')
      IF(I.NE.0) GO TO 9930
      I=INDEX(LINE,'=')
      IF(I.NE.0) GO TO 9930
      LINE(11:)=LINE
      LINE(1:10)=' '
9930  CALL NONBLANK(LINE,I)
      IF(I.EQ.0) I=1
C//   WRITE(6,9920) LINE(1:I)
      WRITE(16,9920) LINE(1:I)
      GO TO 9910
9940  CLOSE(UNIT=98)
C//   WRITE(6,44) TITLE
44    FORMAT('1<DFWDSOL>:',5X,A<NB>//3X,'I',4X,'X(I)',12X,'Y(I)'/)
      WRITE(16,44) TITLE
      IF(IPLT.GT.0) OPEN(UNIT=12,STATUS='NEW',CARRIAGECONTROL='LIST')
C
C  NOW GET THE FORWARD SOLUTION VIA FCODE FOR I=1,N
C
      DO I=1,N
          CALL FCODE(Y,X,B,W,F,I,1)
C         *************************
C//       WRITE(6,50) I,X(I,1),F
50        FORMAT(1X,I3,2D16.8)
          WRITE(16,50) I,X(I,1),F
          YFWD(I)=F
          XVAR(I)=X(I,1)
      ENDDO
      IF(IPLT.EQ.0) GO TO 10
C
C  PREPARE FOR012 FOR PLOT12 OUTPUT (IPLT>0)
C
      TITLES(3)=TITLE(1:40)
      TITLES(4)=TITLE(41:80)
      CALL NONBLANK(TITLES(4),I)
      IF(I.EQ.0) THEN
        NTITLES=3
      ELSE
        NTITLES=4
        IF(TITLES(3)(40:40).EQ.' ') THEN
          TITLES(3)(40:40)='-'
        ELSE
          DO I=39,2,-1
```

```
            IF(TITLES(3)(I:I).EQ.' ') THEN
              TITLES(4)=TITLE(I+1:)
              TITLES(3)(I:39)=' '
              TITLES(3)(40:40)='-'
              GO TO 60
            ENDIF
          ENDDO
        ENDIF
      ENDIF
60    WRITE(12,70) NTITLES,(TITLES(I),I=1,NTITLES)
70    FORMAT(I1/(A))
      WRITE(12,80) N,(XVAR(I),YFWD(I),I=1,N)
80    FORMAT(I/(2G16.8))
      GO TO 10
90    IF(IPLT.GT.0) THEN
        OPEN(UNIT=98,STATUS='OLD',FILE='FOR012.OBS',ERR=98)
C--SPECIAL CASE: "FOR012.OBS" EXISTS--COPY TO FOR012 AS "[OBS]"
        READ(98,70) NTITLES,(TITLES(I),I=1,NTITLES)
        READ(98,80) N,(XVAR(I),YFWD(I),I=1,N)
        CLOSE(UNIT=98)
        NTITLES=4
        TITLES(4)(36:40)='[OBS]'
        WRITE(12,70) NTITLES,(TITLES(I),I=1,NTITLES)
        WRITE(12,80) N,(XVAR(I),YFWD(I),I=1,N)
98      WRITE(12,100)
100     FORMAT('0'/'001'/' ')
      ENDIF
C
C** RETURN FROM DFWDSOL (AFTER E.O.F ON FOR005)
C
      RETURN
900   CALL ERRMSG('E.O.F ON FOR005 READING $FWD INPUT ?',0,6,16)
C
C** END OF SUBROUTINE DFWDSOL
C
      END
C <T1FWDSOL>: TEST1 FOR FWDSOL USING A 7-PARAMETER PROBLEM
C
      EXTERNAL T1FCODE,T1SUBZ
      CALL FWDSOL(T1FCODE,T1SUBZ)
      CALL EXIT
      END
      SUBROUTINE T1FCODE(Y,X,B,PASS,F,IN,IDER)
C--FUNCTION EVALUATION FOR 7-PARAMETER PROBLEM
      DIMENSION Y(1),X(500,5),B(1),PASS(5)
      PASS(1)=X(IN,1)
      T=PASS(1)
      F=B(1)+B(2)*T+B(3)*T**2+B(4)*SIN(B(5)*T)+B(6)*
     1 EXP(-B(7)*T)
      RETURN
      END
      SUBROUTINE T1SUBZ(Y,X,B,PASS,NPASS,N,TITLE,IOUT)
C--INITIALIZATION FOR 7-PARAMETER PROBLEM
C  ($INIT INPUT IS NULL HERE TO MAINTAIN COMPATIBILITY WITH
C   OTHER FWD-PROGRAMS AND NLS-PROGRAMS.)
```

```
      DIMENSION Y(1),X(500,5),B(1),PASS(5)
      CHARACTER*80 TITLE
      NAMELIST/INIT/NULL
      NPASS=1
      IF(IOUT.EQ.1) WRITE(16,10) TITLE
10    FORMAT('0<T1FWDSOL>:',5X,A)
      READ(99,INIT)  !SEE REASON FOR UNIT=99 IN APPENDIX 4 (SUBZ)
      RETURN
      END
```