ARI/R-89-576Z

ANTENNA PATTERN STUDY

TASK II

NAS8-37194

CONTRACT NUMBER: UI/ARI-88-1

FINAL REPORT

MARCH 29, 1989

PREPARED FOR

UWOHALI, INC.
3816 BOB WALLACE AVENUE, SW
HUNTSVILLE, ALABAMA 35805

PREPARED BY

APPLIED RESEARCH, INC.
5025 BRADFORD BOULEVARD
HUNTSVILLE, ALABAMA 35804

WARREN HARPER

PREFACE

The work described in this report was performed in compliance with the requirements of Contract UI/ARI-88-1, Antenna Pattern Study, Task II. Two electromagnetic scattering codes, NEC-BSC, Revision 2, and ESP4 were installed on a Compaq 386/20 computer, together with graphic display codes and auxiliary input data codes. The NEC-BSC and ESP4 codes were developed at the Electroscience Laboratory of Ohio State University. Several revisions were made on these codes by ARI to make them suitable for use on the Compaq computer.

The writer wishes to acknowledge the very valuable assistance of Mr. Mike Vinson of ARI in the revision and compilation of the software described in this report.

TABLE OF CONTENTS

LIST OF APPENDICES

LIST OF FIGURES

1.0 INTRODUCTION

Two electromagnetic scattering codes, NEC-BSC and ESP3, have been delivered under a previous contract (UI/ARI-87-3) and have been installed on a NASA VAX computer for use by George C. Marshall Space Flight Center antenna design personnel. The purpose of the work performed under this contract was to update the existing codes and certain supplementary software, to install the codes on a computer that will be delivered to the customer, to provide capability for graphic display of the data to be computed by use of the codes and to assist the customer in the solution of specific problems that demonstrate the use of the codes. With the exception of of one code revision, all of these tasks have been performed.

2.0 CODE REVISIONS

Tasks 1 through 3 of the Statement of Work of this contract call for acquisition and evaluation of the scattering code NEC-BSC, Revision 3. This revision was expected to be released by Ohio State University shortly after award of this contract. However, it has not yet been released, and recent communications with OSU indicate that it will not be released for several more months. Therefore, it was not possible to incorporate the revised code in this effort. Discussions between all concerned parties have led to an agreement that installation of the existing version of the code on the PC will be satisfactory.

No revision of the moment-method code ESP3 was required under the contract, inasmuch as the code was not expected to be revised. However, use of the code by ARI and by others revealed an that an error existed which degraded the quality of the computed data under certain conditions. The error was subsequently corrected by OSU and a revised version ESP4 was released. That version was immediately procured and was installed on the Compaq 386/20 computer.

Several revisions in the two codes were made by ARI and will be discussed in detail.

3.0 INSTALLATION OF NEC-BSC AND ESP4 ON 386/20 COMPUTER

The random-access memory requirement for NEC-BSC does not exceed that which can be accommodated by the 16-bit Microsoft compiler. The requirement is also essentially independent of the electrical size (size measured in wavelengths) of the scattering body or bodies for which the data are being computed. It depends only on the number of scattering elements being modeled, the dependency being somewhat linear. It is capable, in its present form, of solving scattering problems involving fairly complex structures, and is considered adequate for typical NASA uses in that respect.

The ESP4 code, like its ESP3 predecessor, has a very different type of memory requirement. The required amount of RAM depends on the electrical size of the scattering body. The moment method, on which the code is based, involves the formation and inversion of a square matrix, the size of which depends on the number of "patches" into which the scatterer surface is divided. The amount of memory required to perform the required computations increases exponentially with model size. An assessment of memory requirement for the 386 computer, taking into account the type of scattering problems that may be expected to be encountered by MSFC design personnel, as well as run time and cost, was made by ARI. A compromise was made between these factors and the computer RAM was increased to a value of 9 megabytes. This compromise value is considered adequate to accommodate all typical design problems concerning antenna

radiating elements and associated electrically conductive surfaces. It can be used for scattering bodies which are roughly three or four wavelengths (or less) across.

The large amount of memory required by the ESP4 code cannot be accessed by a 16-bit compiler, so that a 32-bit compiler had to be acquired. It was also decided that both codes should be compiled by use of the same compiler. The initial choice was the Lahey compiler. This compiler was procured and used to compile the NEC-BSC code, which was then tested on examples for which the correct output data were known. Initial attempts to do the compilation were unsuccessful, and it was discovered that the compiler would not accept certain statements as formatted in the source code. The format of these statements was changed and the compilation was accomplished. However, use of the executable code on test cases did not produce correct data. Further investigation revealed that certain statements in the source code, although expressed in good Fortran, could not be properly compiled. The vendor was apprised of this deficiency and very promptly made a correction to his software. A revised version of the compiler was then made available. This version correctly compiled the statements mentioned above, and the resulting executable code produced correct data when flat-plate scatterer models were used. However, when cylindrical scattering models were introduced, incorrect data were again generated.

The errors produced in the executable code were suspected to

be related to some optimization feature in the Lahey compiler. The source code, as received from OSU, would produce incorrect output data when compiled on the VAX in the OPTIMIZE mode. If compiled in the NOOPTIMIZE mode, the resulting executable code would produce good data. Ohio State personnel experienced the same results. It was found by ARI that the source code could be correctly compiled on the VAX in the OPTIMIZE mode if a section of the main program was removed and made into a subroutine which is called at the appropriate point by the main program. This change was made, and remains in the code as delivered. However, compilation on the Compaq PC, using the Lahey compiler, still did not produce a good executable code.

At this point, it was decided that it would not be wise to continue the effort to use the Lahey compiler, and that an alternate choice should be considered. Acting on that decision, the SVS FORTRAN-386 compiler was procured from Science International Corporation, Los Altos, California.

Initial efforts to compile the NEC-BSC source code with the SVS compiler were not successful. Again, errors occurred that seemed to be associated with optimization. An inquiry was made to the vendor and it was learned that the compiler contains five optional levels of optimization: 0 through 4. The default level is level 4 (the highest), and is the level that was being used. A change to level 3 was made and successful compilation was accomplished. After being compiled and linked, the resulting

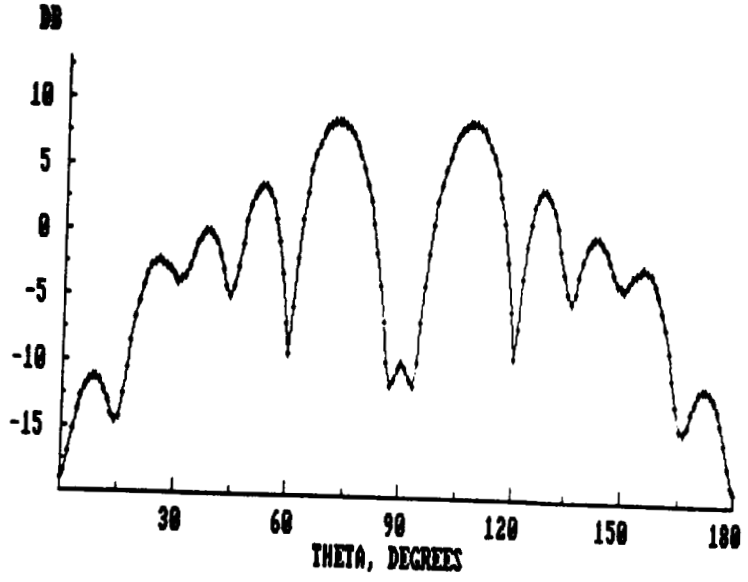
executable code was tested and found to generate valid data on all test cases on which it was tried.

After successful compilation and linking of NEC-BSC, the ESP4 source code was also successfully compiled and linked, using the SVS COMPILER and the Phar Lap linker. It also was found to produce valid output data on known test cases.

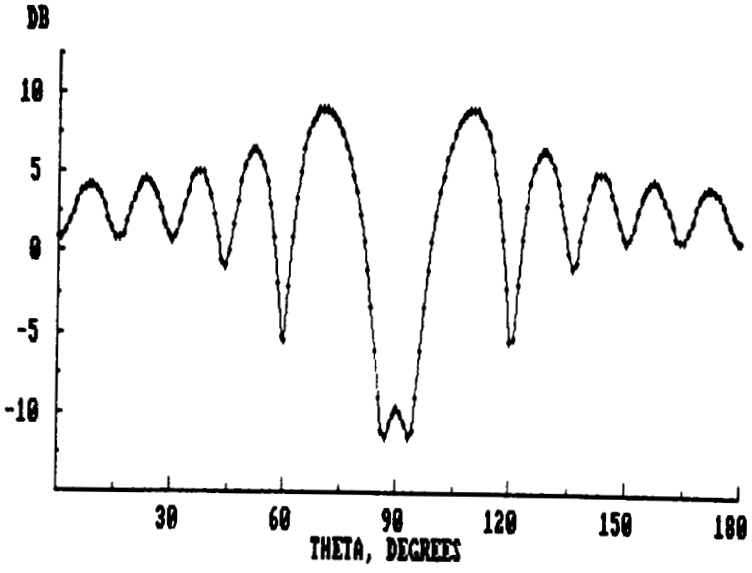
4.0 GRAPHICS CAPABILITY

A requirement exists for providing a graphic display of the data generated by the NEC-BSC and ESP4 codes. These codes produce output files which contain data relating to various components of the computed field, and for optional pattern planes. Thus, the desired graphics code should provide access to any polarization component that the user wishes to view and should do in convenient manner. It should also perform scaling functions which will make most effective use of the plotting area and will select the most appropriate increments of angle and amplitude for the data being displayed. The two graphic codes that are provided meet these requirements.

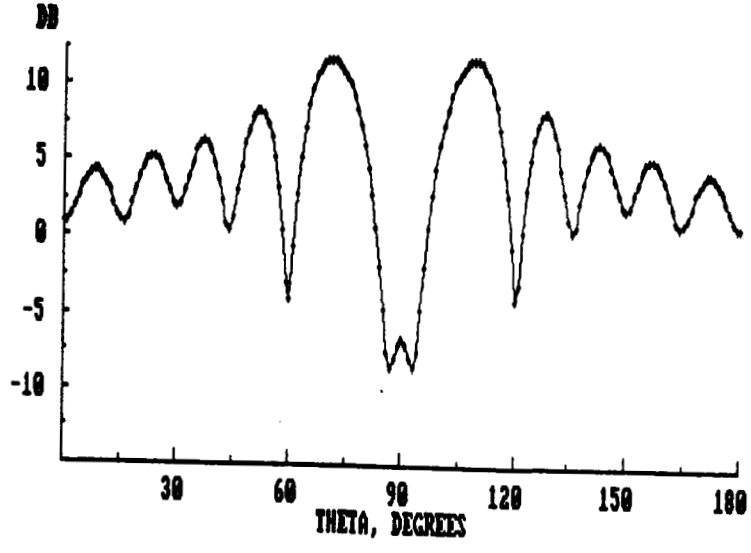
The first of the two codes, written for use with NEC-BSC, is called PLTGTD. The second, for use with ESP4, is PLTMM. These codes make use of a set of graphics routines called GRAFMATICS. Both codes request from the user the desired polarization component, which he then enters from the keyboard. The code then selects the appropriate columns of data from the output file, determines whether the pattern cut is of the great-circle or the conical type (for NEC-BSC), performs the scaling, sets up the grids and displays the data. The resulting plot can then be viewed and, if desired, transferred to the printer for hard copy. Examples are shown in Figure 1.



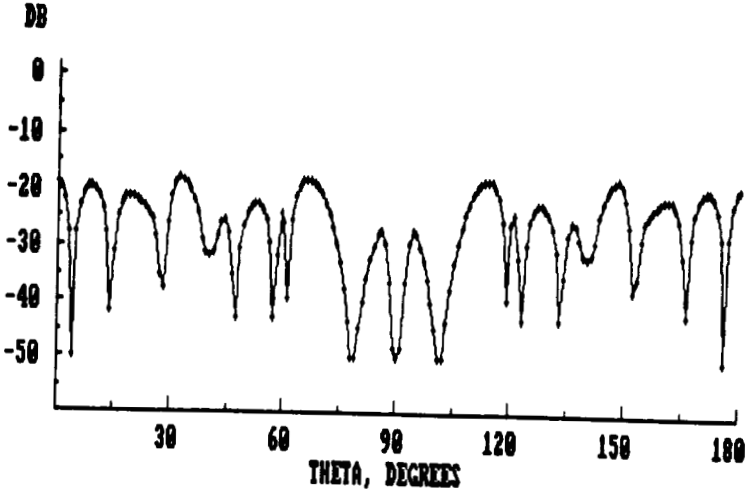
(a) E-Theta



(b) E-Phi



(c) Major axis of ellipse



(d) Minor axis of ellipse

Figure 1. Example of antenna pattern display.

5.0 USE OF THE SCATTERING CODES

The NEC-BSC and ESP4 codes are are stored on the hard disk in two subdirectories. NEC-BSC is stored in subdirectory GTD. ESP4 is stored in subdirectory MOMENT. They are conveniently executed and displayed by use of two batch files, GTD (in the GTD subdirectory) and MM (in the MOMENT subdirectory).

To run the NEC-BSC code, the GTD subdirectory is selected, after which the operator simply types the command GTD on the keyboard. This command sets up a sequence of actions. The first of these actions is to install the GRAPHICS capability. The second is to delete any previous plot file. The NEC-BSC executable code is then called and executed. In the course of execution, it will ask the user the name of the input data file to be used in the computation. This file must have already been prepared. The output data are then computed and placed in the output files. Figure 2 shows the messages that are output to the monitor as the code is being executed. One of the output files, OUTPUT.DAT, contains the complete output information, including input data and other descriptive information. It may be output to the printer at any time. The other file, PLOT.DAT, contains only numeric data to be accessed by the graphics code PLTGTD. This code is then called and requests that the user identify the polarization component that he wishes to display. After that piece of information has been entered from the keyboard, the graphics code selects and displays the requested data.

```
C:\GTD>GTD

C:\GTD>GRAPHICS

C:\GTD>DEL PLOT.DAT

C:\GTD>SCAT
  What is the name of the input file?
EXAMPLE1.DAT
ReCE: FAR ZONE PLATE TEST, EXAMPLE 1A.

ReUN: UNITS IN INCHES

ReFR: FREQUENCY IN GHZ.

RePD: PATTERN CUT

RePG: PLATE GEOMETRY

ReSG: SOURCE GEOMETRY

ReXQ: EXECUTE CODE

ReEN: END CODE


C:\GTD>PLTGTD
THE PATTERN IS READY TO BE PLOTTED


ENTER POLARIZATION COMPONENT DESIRED:
    1 -  E-THETA
    2 -  E-PHI
    3 -  MAJOR AXIS OF POLARIZATION ELLIPSE
    4 -  MINOR AXIS OF POLARIZATION ELLIPSE
    5 -  TOTAL AMPLITUDE
```

Figure 2. Monitor display for execution of NEC-BSC.

Execution of ESP4 is accomplished in essentially the same way. In this case, the user inputs the command MM from the keyboard, rather than GTD. The requests for input file name and polarization component appear as before, and the display is immediately presented on the monitor. The degree of completion is shown on the monitor screen as the impedance matrix is computed and subsequently, as the pattern is computed. An example of this display is shown in Figure 3.

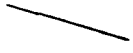
Certain changes were made by ARI in the NEC-BSC code to include the E-theta and E-phi components, together with the major and minor axes of the polarization ellipse and the total field, all in a single plot file. This change permits the plot code PLTGTD to access any selected component from the one file.

```
C:\MOMENT>MM

C:\MOMENT>GRAPHICS

C:\MOMENT>DEL PLOT.DAT

C:\MOMENT>ESP4
$Enter filename for input
EXAMPLE1.DAT
 Opening EXAMPLE1.DAT
 Beginning Z matrix computation
  [Z] MATRIX COL. 1 OF 15 FINISHED
  [Z] MATRIX COL. 2 OF 15 FINISHED
  [Z] MATRIX COL. 3 OF 15 FINISHED
  [Z] MATRIX COL. 4 OF 15 FINISHED
  [Z] MATRIX COL. 5 OF 15 FINISHED
  [Z] MATRIX COL. 6 OF 15 FINISHED
  [Z] MATRIX COL. 7 OF 15 FINISHED
  [Z] MATRIX COL. 8 OF 15 FINISHED
  [Z] MATRIX COL. 9 OF 15 FINISHED
  [Z] MATRIX COL. 10 OF 15 FINISHED
  [Z] MATRIX COL. 11 OF 15 FINISHED
  [Z] MATRIX COL. 12 OF 15 FINISHED
  [Z] MATRIX COL. 13 OF 15 FINISHED
  [Z] MATRIX COL. 14 OF 15 FINISHED
  [Z] MATRIX COL. 15 OF 15 FINISHED
 Computing radiation pattern
 Theta =     .0
 Theta =    3.0
 Theta =    6.0
 Theta =    9.0
 Theta =   12.0
 Theta =   15.0
 Theta =   18.0
 Theta =   21.0
 Theta =   24.0
 Theta =   27.0
 Theta =   30.0
 Theta =   33.0
```



```
 Theta =  348.0
 Theta =  351.0
 Theta =  354.0
 Theta =  357.0
 Theta =  360.0

   CPU RUN TIME FOR RUN    1 GEOMETRY    1 =        70.25 SECONDS


   TOTAL CPU RUN TIME =        70.53 SECONDS

 Programmed STOP

C:\MOMENT>PLTMM
THE PATTERN IS READY TO BE PLOTTED


ENTER POLARIZATION COMPONENT DESIRED:
    1 -  E-THETA
    2 -  E-PHI
```

Figure 3. Monitor Display for ESP4.

6.0 SPECIFIC STUDIES

6.1 SCATTERING STUDY FOR SHUTTLE EXTERNAL TANK

During the course of this contract, MSFC antenna design engineers expressed a concern with the type of scattering that will occur from the external fuel tank of the Shuttle when radar transponder antennas are located on the forward skirts of the solid-rocket boosters (SRB). The NEC-BSC code was viewed as a useful tool in predicting the composite radiation pattern that would result from the combination of direct radiation and the radiation that would be scattered by the external tank. However, a question existed regarding the validity of the data that would be derived by use of the code. The external tank would be modeled as a smooth, perfectly-conducting right circular cylinder. That model does not exactly agree with the actual shape of that part of the tank that lies alongside the antenna location. The surface of the tank in that region (the inter-tank section) does not have a smooth cylindrical surface. Additional strength is provided in the intertank structure by the use of external longitudinal stringers. These stringers produce a somewhat corrugated shape at the scattering surface of the tank.

A study was performed by ARI to assess the validity of the data that would be derived by use of the NEC-BSC code, treating the intertank skin as a smooth cylindrical surface. The work performed in that study and the conclusions that were drawn are described in Report ARI/R-89-573, ELECTROMAGNETIC SCATTERING BY SHUTTLE EXTERNAL TANK SURFACE, dated February 27, 1989.

6.2 OBSCURATION COMPUTATION FOR SRB RADAR ANTENNAS

An intermediate step in determining the adequacy of the radar transponder antenna patterns for two specified radar stations was computation of the optical shadow produced by the various bodies comprising the shuttle cluster in the launch configuration for a source located at an antenna location. This computation was performed by ARI, using the computer code SHADOW. In the absence of scale-model antenna pattern measurements, the results of the computation were used by NASA in preliminary assessment of the adequacy of the antenna arrangement and the radar station layout. A detailed description of the obscuration computation is given in report ARI/R-89-575, TRANSMISSION PATH OBSCURATION FOR RADAR TRANSPONDER ANTENNAS ON SHUTTLE SRB, dated March 24, 1989.

7.0 CONCLUSIONS

The two scattering codes NEC-BSC and ESP4 have been installed on a COMPAQ DESKPRO 386/20 computer, together with graphics software and auxiliary input data codes. All required revisions have been made and the codes are ready for use. It is believed that the codes, in their present form, will provide a valuable predictive capability for antenna patterns associated with large and complex scattering bodies with which NASA is concerned. The absence of the expected second revision of NEC-BSC code is regrettable, but the current revision, installed on the Compaq computer, with graphic display, is expected to be highly useful. It is recommended that the third revision be acquired and installed on the computer when it becomes available.

APPENDIX A

INPUT DATA CODES

```
      PROGRAM BSCINPUT
C
C     THIS PROGRAM PROVIDES AN INTERACTIVE MEANS FOR ENTERING INPUT
C     DATA INTO THE OHIO STATE UNIVERSITY NEC-BSC PROGRAM.  THE
C     REQUESTED PARAMETER VALUES ARE WRITTEN TO A DATA FILE
C     TO BE CALLED BY PROGRAM "SCAT".
C
C     THIS PROGRAM WAS WRITTEN BY:
C
C          J. WARREN HARPER
C          APPLIED RESEARCH, INC.
C          5025 BRADFORD BLVD.
C          HUNTSVILLE, ALABAMA 35805
C
c     Change section written by Mike Vinson
c
c     Revision date:  January 27, 1989
C
      Character*3    STRING
      Character*36   Temp
      Character*36   INPUT
      Character*2    NZERO
      Character*1    Answer
      Character*30   Description(33)
      Character*40   Line
      Character*40   Section(10)
C
      Description(1) = 'BP: Back or Bistatic Scatter'
      Description(2) = 'CE: Last or only Comment'
      Description(3) = 'CG: Cylinder Geometry'
      Description(4) = 'CM: Comment Card'
      Description(5) = 'EN: End program'
      Description(6) = 'FM: Swept Frequencies'
      Description(7) = 'FR: Frequency'
      Description(8) = 'GP: Infinite Ground Plane'
      Description(9) = 'LP: Line Printer Output'
      Description(10) = 'NC: Next set of Cylinders'
      Description(11) = 'NG: No Ground Plane'
      Description(12) = 'NP: Next Set of Plates'
      Description(13) = 'NR: Next Set of Receivers'
      Description(14) = 'NS: Next Set of Sources'
      Description(15) = 'NX: Next Problem'
      Description(16) = 'PD: Far Zone Pattern Cut'
      Description(17) = 'PG: Plate Geometry'
      Description(18) = 'PN: Near Zone Pattern Cut'
      Description(19) = 'PP: Plotter output'
      Description(20) = 'PR: Gain or Coupling Factors'
      Description(21) = 'RA: Receiver Array Geometry'
      Description(22) = 'RD: Far Zone Range'
      Description(23) = 'RG: Receiver Geometry'
      Description(24) = 'RM: NEC - MM Receiver Input'
      Description(25) = 'RT: Rotate-Translate Geometry'
      Description(26) = 'SA: Source Array Geometry'
      Description(27) = 'SG: Source Geometry'
      Description(28) = 'SM: NEC - MM Source Input'
      Description(29) = 'TO: Test Options'
      Description(30) = 'UF: Model Scale Factor'
      Description(31) = 'UN: Units of Geometry'
      Description(32) = 'US: Units of Source Size'
      Description(33) = 'XQ: Execute Code'
```

```
c
      Write(*,300)
 300  Format(1H$,'Would you like to edit the existing file? ')
      Read(*,310) Answer
 310  Format(A1)
 320  Format(A40)
c
c     See if the user wants to edit the current file
c
      If( answer .eq. 'Y' .or. answer .eq. 'y') then
c
c        Enter the change section
c
         Open( Unit = 1, File = 'Input.bak', Status = 'Old')
         Open( Unit = 2, File = 'Input.dat', Status = 'Unknown')
         Read(1,320,End=999) Line
 400     Call Search(Line,Description,k)
         Section(1) = Line
         n = 0
         icount = 1
c         Do While (n .eq. 0)
405      Continue
            Read(1,320,End=999) Line
            Call Search(Line,Description,n)
            icount = icount + 1
            Section(icount) = Line
          If( n .eq. 0) goto 405
c         End do
         icount = icount - 1
         Call Change(Section,k,Description,icount)
         goto 400
 999     Continue
         Call Change(Section,k,Description,icount)
         Close(2)
      Else

      OPEN(1,FILE='FOR005.DAT',STATUS='NEW')
C
      STRING = 'CE:'
      WRITE(*,1)
    1 FORMAT(1H$,'Enter Comment (max 36 chars)   ')
      READ(*,2)INPUT
    2 FORMAT(A36)
      WRITE(1,3)STRING,INPUT
    3 FORMAT(A3,A36)
C
C     Get units for this run from the user.
C
 1000 WRITE(*,200)
  200 FORMAT(1X,'Enter number indicating units to be used -'/15X,
     .'1 = Meters'/15X,'2 = Feet'/15X,'3 = Inches'/)
      READ(*,2) INPUT

      If( Input .gt. '3' .or. Input .lt. '1' ) Then
         Write(*,*) 'Your selection was out of range.  Please try agair
         goto 1000
      End if
```

```
C
      STRING='UN:'
      WRITE(1,9) STRING
      WRITE(1,2) INPUT
C
C     Get frequency from user.
C
      WRITE(*,5)
    5 FORMAT(1X,'FREQUENCY IN GHZ.?')
      READ(*,2) INPUT
      STRING='FR:'
      WRITE(1,9) STRING
      WRITE(1,2) INPUT
    7 FORMAT(I2)
    8 FORMAT(I1,1X,A1)
    9 FORMAT(A3)
C
      WRITE(*,10)
   10 FORMAT(1X,'    *** Orientation of Pattern Axes ***'/)
      STRING='PD:'
      WRITE(*,15)
   15 FORMAT(1X,'THETA, PHI FOR Z AXIS, THETA, PHI FOR X AXIS (1 LINE)'
      READ(*,2) INPUT
      WRITE(1,9) STRING
      WRITE(1,2) INPUT
C
C                    *** TYPE OF PATTERN CUT ***
C
      WRITE(*,210)
  210 FORMAT(1X,'TYPE OF PATTERN CUT DESIRED (GREAT CIRCLE OR CONICAL)'
     1/'           T = CONICAL CUT (CONSTANT THETA)'/'           F = GRE
     2 CIRCLE CUT (CONSTANT PHI)'//' ENTER T OR F, FOLLOWED BY VALUE O
     3THE FIXED ANGLE, ON ONE LINE.')
      READ(*,2) INPUT
      WRITE(1,2) INPUT
C
C                    *** ANGULAR RANGE DESIRED ***
C
      WRITE(*,220)
  220 FORMAT(1X,'ANGULAR RANGE DESIRED FOR PATTERN: INITIAL ANGLE, FIN
     1 ANGLE, ANGULAR INCREMENT')
      READ(*,2) INPUT
      WRITE(1,2) INPUT
C
C                        *** PLATE GEOMETRY ***
C
      WRITE(*,40)
   40 FORMAT(1X'HOW MANY PLATES?')
      READ(*,*) NOPLT
      IF(NOPLT.LT.1) GOTO 82
C
      STRING='PG:'
      DO 80 MP=1,NOPLT
      WRITE(1,9) STRING
```

```
C
      WRITE(*,50) MP
   50 FORMAT(1X,'PLATE NUMBER ',I2,/' HOW MANY CORNERS?')
      READ(*,7) NUM
      NZERO='0'
      WRITE(1,8) NUM,NZERO
C
      DO 70 ME=1,NUM
      WRITE(*,60) ME
   60 FORMAT(1X,'X,Y,Z POSITION OF CORNER NUMBER ',I2)
      READ(*,2) INPUT
      WRITE(1,2) INPUT
   70 CONTINUE
   80 CONTINUE
C
C                *** CYLINDER GEOMETRY ***
C
   82 WRITE(*,84)
   84 FORMAT(1X,'HOW MANY CYLINDERS?')
      READ(*,*) NCYL
      IF(NCYL.LT.1) GOTO 99
      DO 94 N=1,NCYL
      STRING='CG:'
      WRITE(1,9) STRING
      WRITE(*,86) N
   86 FORMAT(1X,'LOCATION (X,Y,Z) OF ORIGIN, CYLINDER NO. ',I1)
      READ(*,2) INPUT
      WRITE(1,2) INPUT
C
      WRITE(*,88) N
   88 FORMAT(1X,'CYLINDER NO. ',I1,' ORIENTATION'/1X,'THETA-Z, PHI-Z,
     1ETA-X, PHI-X')
      READ(*,2) INPUT
      WRITE(1,2) INPUT
C
      WRITE(*,90)
   90 FORMAT(1X,'CYLINDER RADII:  RX, RY')
      READ(*,2) INPUT
      WRITE(1,2) INPUT
C
      WRITE(*,92)
   92 FORMAT(1X,'POSITIONS & ANGLES OF END CAPS'/1X,'POSITION & ANGLE
     1R NEGATIVE END, POSITION & ANGLE FOR POSITIVE END.')
      READ(*,2) INPUT
      WRITE(1,2) INPUT
   94 CONTINUE

C
C                  *** SOURCE GEOMETRY ***
C
   99 WRITE(*,100)
  100 FORMAT(1X,'       *** SOURCE GEOMETRY ***'/)
      WRITE(*,110)
  110 FORMAT(1X,'HOW MANY SOURCE ELEMENTS?')
      READ(*,*) NWIRES
      STRING='SG:'
```

```
C
      DO 180 MS=1,NWIRES
      WRITE(1,9) STRING
      WRITE(*,120) MS
  120 FORMAT(1X,'SOURCE ELEMENT NUMBER',I2/)
      WRITE(*,130)
  130 FORMAT(1X,'X,Y,Z POSITION OF ELEMENT CENTER?')
      READ(*,2) INPUT
      WRITE(1,2) INPUT
C
      WRITE(*,140)
  140 FORMAT(1X,'THETA & PHI ANGLES FOR LENGTH & WIDTH VECTORS'/'
     1(THETA-L, PHI-L, THETA-W, PHI-W')
      READ(*,2) INPUT
      WRITE(1,2) INPUT
C
      WRITE(*,160) MS
  160 FORMAT(1X,'TYPE, LENGTH AND WIDTH OF ELEMENT',I2/'          TYPE:'/
     1             -1    UNIFORM CURRENT DISTRIBUTION'/'            -2
     2PIECEWISE SINUSOIDAL DISTRIBUTION'/)
      READ(*,2) INPUT
      WRITE(1,2) INPUT
C
      WRITE(*,170)  S
  170 FORMAT(1X,'EXCITATION (MAGNITUDE, PHASE) FOR ELEMENT',I2)
      READ(*,2) INPUT
      WRITE(1,2) INPUT
  180 CONTINUE
      STRING='XQ:'
      WRITE(1,9) STRING
      STRING='EN:'
      WRITE(1,9) STRING
      End if
      CLOSE (1)
      END
c
c
      Subroutine Search(String,Array,n)
c
      Character*80 String
      Character*30 Array(33)
c
c     Search for the first two letters of the string in the first two
c     letters of the elements of the array.
c
      Do 10 i = 1,33
         If( string(1:2) .eq. Array(i)(1:2) ) then
            n = i
            Return
         End if
   10 Continue
      n = 0
      Return
      End
c
```

```
c
      Subroutine Change(Section,k,Description,icount)
c
      Character*40   Section(10)
      Character*40   Line
      Character*30   Description(33)
      Character*1    Answer
c
      Integer*4      k
      Integer*4      Icount
      Integer*4      Offset
      Integer*4      joffset
c
      Character*1    Btemp(2)
      Integer*2      Itemp
c
      Equivalence ( Itemp, Btemp(1) )
c
      Write(*,*)
      Write(*,*)
      Write(*,*) 'Command card found:  ',Description(k)(1:3)
      Write(*,*) 'This is described as:  ',Description(k)(4:30)
      Write(*,*)
      Write(*,*) 'These are the cards in this set:'
      Do 10 i = 1,Icount
         Write(*,*) Section(i)
 10   Continue
      Write(*,*)
      Write(*,90)
 90   Format(1H$,'Do you want to change any of the cards in this SET?
      Read(*,110) Answer
      Write(*,*)
      Write(*,*)
c
      If( Answer .eq. 'Y' .or. Answer .eq. 'y') Then
         Do 20 i = 1,Icount
 300     Write(*,*) Section(i)
         Write(*,80)
 80      Format(/1X,'Note: To insert a new card',/
     .                 '      before this card, just',/
     .                 '      type INSERT  To delete',/
     .                 '      a card, type DELETE.'/)
         Write(*,100)
 100     Format(1H$,'Do you want to Change this card? ')
         Read(*,120) Line
         Answer = Line(1:1)
c
c     To test all possible combinations of the word insert in both
c     upper and lower case, convert each of the first six letters
c     to upper case.
c
         Do 500 k = 1,6
            Btemp(1) = Line(k:k)
            if( itemp .ge. 97 .and. itemp .le. 122 ) then
               Itemp = Itemp - 32
               Line(k:k) = Btemp(1)
            End if
 500     Continue
```

```
c
         If( Line(1:6) .eq. 'INSERT' ) then
            Call Insert
            Goto 300
         End if
         if( Line(1:6) .eq. 'DELETE' ) Goto 20
         If( Answer .eq. 'Y' .or. Answer .eq. 'y') Then
            Write(*,*)
            Write(*,70)
70          Format(//1X,'Note: Type ^ to begin and end inserted data',/
     .                  '       for this card.  Type # to delete',/
     .                  '       characters.'/ )
            Write(*,*) 'Enter the New card below (Space = no change):'
            Write(*,*) Section(i)
            Read(*,120) Line
            Ins_flag = 0
            Offset = 0
            Do 200 j = 1,40
               If( Line(j:j) .eq. '^') then
                  Ins_flag = 1 - Ins_flag
                  Offset = Offset + 1
                  Goto 200
               End if
               If( Line(j:j) .eq. '#') then
                  Do 125 m = j-Offset,39
                     section(i)(m:m) = section(i)(m+1:m+1)
125               Continue
                  Offset = Offset + 1
                  Goto 200
               End if
               If( Line(j:j) .ne. ' ' .and. Ins_flag .eq. 0) then
                  joffset = j - Offset
                  Section(i)(joffset:joffset) = line(j:j)
               Else if( ins_flag .eq. 1 ) then
                  Do 130 m = 39,j-Offset,-1
                     Section(i)(m+1:m+1) = Section(i)(m:m)
130               Continue
                  joffset = j - Offset
                  Section(i)(joffset:joffset) = line(j:j)
               End if
200         Continue
         End if
         Write(*,*)
         write(2,120) Section(i)
 20      Continue
      Else
         Do 30 i = 1,icount
            write(2,120) Section(i)
30       Continue
      End if
110   Format(A1)
120   Format(A40)
      Return
      End
```

```
c
c
      Subroutine Insert
c
      Character*40   Line
      Character*1    Answer
c
  1   Write(*,10)
 10   Format(/' Type the card to be inserted:')
      Read(*,100) Line
      Write(*,200) Line
      Write(*,300)
      Read(*,90) Answer
      If( Answer .eq. 'Y' .or. answer .eq. 'y') then
         write(2,100) Line
      Else
         Goto 1
      End if
 90   Format(A1)
100   Format(A40)
200   Format(//1X,A40/)
300   Format(1H$,'Is this card correct? ')
      Return
      End
```

```
      PROGRAM INDAT4
C
C     This program provides an interactive means for entering input
C     data into the ohio state university esp4 program.  The requested
C     parameter values are written to a data file, specified by the
C     user, in the proper order to be called by esp4.
C
C     This program was written by:
C
C           J. WARREN HARPER
C           APPLIED RESEARCH, INC.
C           5025 BRADFORD AVE.
C           HUNTSVILLE, ALABAMA   35805
c
c           Modified by: Mike Vinson
c
C
      DIMENSION NCNRS(10)
      DIMENSION X(10),Y(10),Z(10)
      DIMENSION NPLA(10),BDSK(10),VGA(10),ZLDA(10)
      DIMENSION PCN(3,10,10),IA(10),IB(10),SEGM(10)
      DIMENSION IREC(10),IPN(10),IGS(10),NAS(10)
      DIMENSION NSA(10),VG(10),ZLD(10),IFM(10)
      DIMENSION IABFP(10),IABAP(10)
      DIMENSION VLG(10),ZL(10)
C
      CHARACTER *1 USEWRS,GEOPRT,DFLT,FARZN,USEPLT,MCOUPL,WRTIMP
      CHARACTER *1 CHFREQ,CHCPAT,CHGRAD,CHGCON,Change
      Character*40 Filename, Variable
      Character*1 Answer
C
      COMPLEX VLG
      COMPLEX ZL
      COMPLEX VGA
      COMPLEX ZLDA
      Complex*8   Zsht(30)
C
      WRITE(*,10)
      WRITE(*,20)
      WRITE(*,30)
   10 FORMAT(' This program (ESP4) is based on the method of moments')
   20 FORMAT('                ************************')
   30 FORMAT(' ')
C
      WRITE(*,2000)
 2000 FORMAT(1H$,'Do you wish to change an existing input file? (Y/N) ')

      READ(*,50) CHGFIL
      IF( CHGFIL .EQ. 'N' .OR. CHGFIL .EQ. 'n' ) GOTO 8

      Write(*,2011)
 2011 Format(1H$,'Enter the filename to edit:  ')

      Read(*,431) Filename
431   format(A40)

      Open( Unit=10, File=Filename, Status='old')
      ICHG=0
C
C     READ IN VALUES FROM EXISTING DATA FILE
```

```
C
      READ(10,*) NGO, NPRINT, NRUNS, NWGS, IWR, IWRZT, INT, INTP, INTD,
                 INWR, IRGM, IFIL, RF, INDZI
      If( Indzi .ne. 0) Then
         Read(10,*) Fmc1, Fmc2, Dfzi, Dff, Irs12, Thrd, Phrd,
                    Thinc, Phinc
      End if
      If( Indzi .eq. 0) Then
         READ(10,*) IFE, IPFE, FNDFE, PHFE
         READ(10,*) IFA, IPFA, FNDFA, THFA
         READ(10,*) ISE, IPSE, FNDSE, PHSE, THIN, PHIN
         READ(10,*) ISA, IPSA, FNDSA, THSA
      End if

      READ(10,*) FMC,CMM,A
C     IF(USEPLT .ne. 'N' .OR. USEPLT .ne. 'n') Then
      READ(10,*) NPLTS
      If( Nplts .ne. 0) Then
         DO 1520 NPL=1,NPLTS
            READ(10,*) NCNRS(NPL), SEGM(NPL), IREC(NPL), IPN(NPL),
                       IGS(NPL), Zsht(Npl)
            DO 1510 NCNR=1,NCNRS(NPL)
               READ(10,*) PCN(1,NCNR,NPL), PCN(2,NCNR,NPL),
                          PCN(3,NCNR,NPL)
 1510       CONTINUE
 1520    CONTINUE
      End if
C 1530 IF(USEWRS.EQ.'N'.OR.USEWRS.EQ.'n') GOTO 810
      READ(10,*) IWRZM,IRDZM
      If( Jnwr .ne. 0 .and. Irgm .ne. 0) Then
         READ(10,*) NM,NP,NAT,NFPT,NFS1,NFS2
C
         DO 1550 I=1,NP
           READ(10,*) X(I),Y(I),Z(I)
 1550    CONTINUE
C
         DO 1560 I=1,NM
            READ(10,*) IA(I),IB(I)
        write(*,*) ia(i), ib(i)
 1560    CONTINUE
C
         DO 1570 I=1,NFPT
            READ(10,*) IFM(I),IABFP(I),VLG(I),ZL(I)
        write(*,*) IFM(I),IABFP(I),VLG(I),ZL(I)
 1570    CONTINUE
C
         DO 1580 I=1,NAT
            READ(10,*) NAS(I),IABAP(I),NPLA(I),VGA(I),ZLDA(I),BDSK(I)
        write(*,*) NAS(I),IABAP(I),NPLA(I),VGA(I),ZLDA(I),BDSK(I)
 1580    CONTINUE
      End if
      REWIND 10
C
C     END OF READING OF DATA FILE
C
      Variable = 'Frequency'
      Call Rchange(Variable, FMC, 0)
 3200 FORMAT(/)
C
C 2002 IF(USEWRS.EQ.'N'.OR.USEWRS.EQ.'n') GOTO 2006
```

```
      If( Nwgs .gt. 0) Then

         Variable = 'Wire Radius'
         Call Rchange( Variable, A, 0)
C
         Variable = 'Wire Conductivity'
         Call Rchange( Variable, Cmm, 0)
C
      End if
C
      If( INDZI .ne. 0) then
5020     Write(*,*) 'The current values for Frequency Sweep are: '
         write(*,5010) Fmc1, Fmc2, Dfzi, Dff, Irs12, Thrd, Phrd,
     .                 Thri, Phri
5010     Format(5x,'1 -Begin of sweep (MHz)',F8.2,/
     .     5x,'2 -End of sweep (Mhz)',F8.2,/
     .     5x,'3 -Step size for Imdepance Matrix (Mhz)',F8.2,/
     .     5x,'4 -Step size for MM computation (Mhz)',F8.2,/
     .     5x,'5 -Indicator of radiation or scattering problem',I2,/
     .     10x,' 1 - Implies radiation problem',/
     .     10x,' 2 - Implies scattering problem',/
     .     10x,'-2 - Implies scattering including incident wave',/
     .     5x,'6 -Angle for radiated or scattered feild (Degrees)',/
     .     10x,'Theta = ',F8.2,5x,'Phi = ',F8.2,/
     .     5x,'7 -Angle of incident for scattering problem (Degrees)',/
     .     10x,'Theta = ',F8.2,5x,'Phi = ',F8.2//)
         Write(*,6050)
         Read(*,50) Answer
         If (Answer .eq. 'y' .or. Answer .eq. 'Y') Then
            Write(*,2035)
            Read(*,*) I
            If( I .eq. 1 ) then
                Variable = 'Beginning of Sweep (MHz)'
                Call Rchange( Variable, Fmc1, 0)
            Else if(I .eq. 2) then
                Variable = 'End of Sweep (MHz)'
                Call Rchange( Variable, Fmc2, 0)
            Else if(I .eq. 3) then
                Variable = 'Step Size for Z matrix (MHz)'
                Call Rchange( Variable, Dfzi, 0)
            Else if( I .eq. 4) then
                Variable = 'Step Size for MM Computation (Mhz)'
                Call Rchange( Variable, DFF, 0)
            Else if( I .eq. 5) then
5025            Variable = 'Radiation or Scattering Problem'
                Call Ichange( Variable, IRS12, 0)
                j = Irs12
                if( j.ne.1 .or. j.ne.2 .or. j.ne.-2 ) then
                   Write(*,*) 'Input out of range.  Try again'
                   Goto 5025
                End if
            Else if( I.eq. 6) then
                Variable = 'Theta'
                Call Rchange( Variable, Thrd, 0)
                Variable = 'Phi'
                Call Rchange( Variable, Phrd, 0)
            Else if( I.eq. 7) then
                Variable = 'Theta'
                Call Rchange( Variable, Thri, 0)
                Variable = 'Phi'
```

```
                Call Rchange( Variable, Phri, 0)
             End if
             Goto 5020
          End if
       Else
          Write(*,6020) Ife, Ipfe, Fndfe, Phfe
 6020     Format(' The current values of Pattern Data are: ',//
     .      5x,'1 -Compute near or far zone elevation radiation pattern'
     .      ,I2/,5x,'2 -Write elevational pattern to plot file',I2,/
     .      5x,'3 -Elevation angle increment (Deg) ',F8.2,/
     .      5x,'4 -Constant phi elevational angle (Deg)',F8.2/)
C
          Write(*,6030) Ifa, Ipfa, Fndfa, Thfa
 6030 Format(5x,'5 -Compute near or far zone azimuth radiation pattern'
     .   ,I2/,5x,'6 -Write azimuthal pattern to plot file',I2/,
     .        5x,'7 -Azimuth angle increment (Deg) ',F8.2/,
     .        5x,'8 -Constant theta azimuthal angle (Deg)',F8.2/)
C
C
          Write(*,6050)
 6050     Format(1H$,'Would you like to change any of these? ')
          Read(*,50) Answer
C
          Write(*,6060)
 6060     Format(//' Scattering Section:'/)
          Write(*,6070) Ise, Ipse, Fndse, Phse
 6070     Format(5x,'1 -Type of Scattering in Elevation Plane: ',I2,/10x,
     .                '0 -Do not compute backscatter',/10x,
     .                '1 -Compute Backscatter pattern',/10x,
     .                '2 -Compute Bistatic scattering pattern',/10x,
     .                '3 -Computer Forward scattering pattern',/
     .             5x,'2 -Write scattering pattern to Plot file',I2/,
     .             5x,'3 -Angle increment for scattering ',F8.2,/
     .             5x,'4 -Constant Phi angle (Deg) for scattering',F8.2/)
C
          Write(*,6080) Isa, Ipsa, Fndsa, Thsa
 6080     Format(5x,'5 -Type of Scattering in Azimuthal Plane: ',I2,/
     .             5x,'6 -Write scattering pattern to Plot file',I2/,
     .             5x,'7 -Angle increment for scattering ',F8.2,/
     .             5x,'8 -Constant Theta angle (Deg) for scattering',F8.2/)
C
          Write(*,6040) Thin, Phin
 6040     Format(5x,'9 -Theta angle (deg) of incident wave in Bistatic',
     .          ' Scattering',F8.2,/,
     .          4x,'10 -Phi angle (deg) of incident wave in Bistatic',
     .          ' Scattering',F8.2//)
          Write(*,6050)
          Read(*,50) Answer
       End if
C
 2009 WRITE(*,2010)
 2010 FORMAT(1H$,'Would you like to change the Plate Data? (Y/N) ')
      READ(*,50) Answer
      IF( Answer .eq. 'Y' .OR. Answer .eq. 'y') Then
          Write(*,*) '       Option Menu'
          Write(*,*) '          1 - Add a plate'
          Write(*,*) '          2 - Change a plate'
          write(*,*)
          Write(*,2720)
 2720     Format(1H$,'Your choice: ')
```

```
         Read(*,*) I
         If( I .eq. 1) Then
            NPLTS = NPLTS + 1
            NPL = NPLTS
            Ichg = 1
            CALL PLATE( NPL, NCNRS, SEGM, IREC, IPN, IGS, PCN, ICHG,
                        Zsht)
         Else
            Write(*,2030) Nplts
 2030       Format(1H$'There are now', I2,' plates.   ')
            Write(*,2035)
 2035       Format(1H$'Which would you like to change? ')
            READ(*,*) NPL
            Ichg = 0
            CALL PLATE( NPL, NCNRS, SEGM, IREC, IPN, IGS, PCN, ICHG,
                        Zsht)
         End if
      End if
C

 2055 WRITE(*,2060)
 2060 FORMAT(1H$,'Would you like to change the Wire Data? (Y/N) ')
      READ(*,50) Answer
      IF( Answer .eq. 'Y' .OR. Answer .eq. 'y') Then
         Ichg = 0
         CALL WIRE(NM,NP,NAT,NFPT,NSF1,NSF2,ICHG)
      End if
      GOTO 910
C
C       ********** END OF CHANGE SECTION; BEGIN ORIGINAL INPUT **********
C
    8 ICHG=1
      Write(*,310)
  310 Format(1H$,'Enter the filename to write:   ')
      Read(*,431) Filename
      Open( Unit=10, File=Filename, Status='new')

      WRITE(*,12)
   12 FORMAT(1H$,'Do you wish to specify any of the following',
     .           ' parameters? (Y/N) '/)

      WRITE(*,14)
   14 FORMAT(10X,'* Number of runs to be made'/10X,
     .           '* Print of modal currents'/10X,
     .           '* Print of impedance matrix'/10X,
     .           '* Define frequency sweep'/10X,
     .           '* Number of Simpsons-rule intervals for:'/15X,
     .              'Wire-to-wire impedances'/15X,
     .              'Surface-patch monopoles'/15X,
     .              'Disk monopoles'//)
C
      READ(*,50)DFLT
C
      IF(DFLT.EQ.'Y'.OR.DFLT.EQ.'y') Then
         Variable = 'Number of Runs'
         Call Ichange( Variable, Nruns, Ichg)
C
         Variable = 'Number of Wire Geometries per Run'
         Call Ichange( Variable, Nwgs, Ichg)
C
```

```
          WRITE(*,1025)
 1025     FORMAT(/1X,'MODAL CURRENT PRINTOUT: '/6X,
     .                '0  NO MODAL CURRENT PRINTOUT'/6X,
     .                '1  MODAL CURRENTS PLUS WIRE/PLATE GEOMETRY PRINTED')
          Variable = 'Modal Current Printout'
          Call Ichange( Variable, Iwr, Ichg)
C
          WRITE(*,1030)
 1030     FORMAT(/1X,'WRITE IMPEDANCE MATRIX IN OUTPUT FILE? (Y/N)')
          READ(*,50)WRTIMP
C

          IF(WRTIMP.EQ.'Y'.OR.WRTIMP.EQ.'y') Then
 1032        IWRZT=1
          Else
             IWRZT=0
          End if
C
          WRITE(*,1040)
 1040     FORMAT(/1X,'NUMBER OF SIMPSONS-RULE INTEGRATION INTERVALS: '//
     .                1X,'FOR WIRE-TO-WIRE IMPEDANCES?')
          READ(*,*) INT
          WRITE(*,1050)
 1050     FORMAT(/1X,'FOR SURFACE-PATCH MODULES?')
          READ(*,*) INTP
          WRITE(*,1060)
 1060     FORMAT(/1X,'FOR DISK MONOPOLES?')
          READ(*,*) INTD
          WRITE(*,1070)
 1070     FORMAT(//1X,'HOW IS THE WIRE GEOMETRY DEFINED?'/10X,
     .                '0  BY SUBROUTINE WGEOM'/10X,
     .                '1  DEFINED VIA THE INPUT FILE')
          READ(*,*) IRGM
       Else
C
C        --- APPLY DEFAULT VALUES ---
C    READ 1:
          NRUNS=1
          NWGS=1
          IWR=0
          IWRZT=1
          INT=4
          INTP=6
          INTD=18
          IRGM=1
       End if
C
C
C            --- INPUT NON-DEFAULT INFORMATION ---
C
    35 WRITE(*,40)
    40 FORMAT(/1X,'DOES THE MODEL CONTAIN WIRES? (Y/N)')
       READ(*,50) USEWRS
       WRITE(*,45)
    45 FORMAT(/1X,'DOES THE MODEL CONTAIN PLATES? (Y/N)')
       READ(*,50) USEPLT
C
    50 FORMAT(A1)
       IF(USEWRS.EQ.'Y'.OR.USEWRS.EQ.'y') INWR=1
C
```

```
      WRITE(*,60)
   60 FORMAT(/1X,'SPECIFY REQUIRED OUTPUT:'/10X,
     .           '1  PRINT WIRE & PLATE GEOMETRY'/10X,
     .           '2  PRINT INPUT PARAMETERS & WIRE/PLATE GEOMETRY'/10X,
     .           '3  PRINT NOTHING')
C
      READ(*,*) NPRINT
C
      WRITE(*,70)
   70 FORMAT(/1X,'DO YOU WANT TO PRINT INPUT DATA BEFORE COMPUTING?')
      READ(*,50) IFIRST
      IF(IFIRST.EQ.'Y'.OR.IFIRST.EQ.'y') THEN
          NGO=0
          ELSE
          NGO=1
      ENDIF
C
      Write(*,710)
  710 Format(1H$,'Do you want a Frequency Sweep? ')
      Read(*,50) Answer
      If( Answer .eq. 'Y' .or. Answer .eq. 'y') then
         Write(*,720)
  720    Format(' 1 -Frequency sweep with impedance matrix ',
     .          'interpolation method',/' 2 -Frequency sweep with ',
     .          '"improved" impedance matrix interpolation method')
         Write(*,725)
  725    Format(1H$,'Which would you prefer? ')
         Read(*,*) INDZI
c
         Variable = 'Beginning of Sweep (MHz)'
         Call Rchange( Variable, Fmc1, 1)
c
         Variable = 'End of Sweep (MHz)'
         Call Rchange( Variable, Fmc2, 1)
c
         Variable = 'Step Size for Z matrix (MHz)'
         Call Rchange( Variable, Dfzi, 1)
c
         Variable = 'Step Size for MM Computation (Mhz)'
         Call Rchange( Variable, DFF, 1)
c
  730    Write(*,735)
  735    Format(' Indicator of radiation or scattering problem',/
     .      10x,' 1 - Implies radiation problem',/
     .      10x,' 2 - Implies scattering problem',/
     .      10x,'-2 - Implies scattering including incident wave')
         Variable = 'Radiation or Scattering Problem'
         Call Ichange( Variable, IRS12, 0)
         j = Irs12
         if( j.ne.1 .or. j.ne.2 .or. j.ne.-2 ) then
            Write(*,*) 'Input out of range.  Try again'
            Goto 730
         End if
c
         Write(*,740)
  740    Format(' Angle for Computing Radiated or Scattered Field ',
     .          '(Deg)')
         Variable = 'Theta'
         Call Rchange( Variable, Thrd, 1)
         Variable = 'Phi'
```

```
            Call Rchange( Variable, Phrd, 1)
C
            Write(*,*) 'Angle of Indicent Plane Wave of Scattering (Deg)'
            Variable = 'Theta'
            Call Rchange( Variable, Thri, 1)
            Variable = 'Phi'
            Call Rchange( Variable, Phri, 1)

         Else
            INDZI = 0
            WRITE(*,80)
   80       FORMAT(/1X,'SPECIFY TYPE OF PATTERN CUT'/10X,
                       '1  GREAT-CIRCLE CUT (VARIABLE THETA)'/10X,
                       '2  CONICAL CUT (VARIABLE PHI)'/)
            READ(*,*) ICUT
            WRITE(*,90)
   90       FORMAT(/1X,'FAR-ZONE ANTENNA PATTERN? (Y/N)')
            READ(*,50) FARZN
            WRITE(*,110)
  110       FORMAT(/1X,'VALUE OF CONSTANT ANGLE: ')
            READ(*,*) CONANG
            WRITE(*,130)
  130       FORMAT(/1X,'ANGLE INCREMENT?')
            READ(*,*) ANGINC
            WRITE(*,120)
  120       FORMAT(/1X,'DO YOU WANT AN OUTPUT FOR PLOTTING?  (Y/N)')
            READ(*,50) PLTOUT
C
            IF( FARZN .EQ. 'N' .OR. FARZN .EQ. 'n') Then
C
C                 PARAMETER VALUES SET FOR FAR-ZONE RADIATION
C                    PATTERN FOR BOTH TYPES OF PATTERN CUT
C
               Variable = 'Radius for Field Point (meters)'
               Call RChange( Variable, RF, 1)
               IFE=0
               IPFE=0
               FNDFE=1
               PHFE=1.0
               IFA=0
               IPFA=0
               FNDFA=1
               THFA=1.0
C
C        --- INPUT MATRIX DOES NOT EXIST FOR THIS CASE ---
               IWRZT=0
C
C           ---  PLATES ARE MODELED AS PERFECT CONDUCTORS ---
               CMM=-1.0
C
C              --- "WIRE RADIUS" SET TO .001 FOR PLATE ---
               A=.001
C
C            --- INFORMATION INPUT FOR PLANE-WAVE SCATTER ---
C
               WRITE(*,160)
  160          FORMAT(/1X,'SPECIFY TYPE OF SCATTER COMPUTATION: '/10X,
                          '1  BACKSCATTER'/10X,
                          '2  BISTATIC SCATTER'/10X,'3  FORWARD SCATTER')
C
```

```
            READ(*,*) ITYPE
            IF(ICUT.EQ.1) THEN
               ISE=ITYPE
               ISA=0
            ELSE
               ISE=0
               ISA=ITYPE
            ENDIF
C
            IF(PLTOUT.EQ.'Y'.OR.PLTOUT.EQ.'y') THEN
               IPSE=1
               IPSA=1
            ELSE
               IPSE=0
               IPSA=0
            ENDIF
C
C       -- INPUT ANGLE INCREMENTS FOR PLANE-WAVE SCATTER ---
C
            FNDSE=ANGINC
            FNDSA=ANGINC
C
C       --- INPUT CONSTANT ANGLE FOR PLANE-WAVE SCATTER ---
C
            PHSE=CONANG
            THSA=CONANG
C
C              --- INPUT ANGLE OF INCIDENCE ---
C
            WRITE(*,170)
  170       FORMAT(/1X,'WHAT IS THE ANGLE OF INCIDENCE? (THETA, PHI)')
            READ(*,*) THIN,PHIN

         Else
C
C         --- PARAMETERS SET FOR PLANE-WAVE SCATTER CASE ---
C
            RF = -1
            ISE=0
            IPSE=0
            FNDSE=1
            PHSE=1.0
            THIN=1.0
            PHIN=1.0
            ISA=0
            IPSA=0
            FNDSA=1
            THSA=1
C
C      --- CONSTANT ANGLE VALUE IS SET FOR EITHER PATTERN CUT ---
C
            PHFE=CONANG
            THFA=CONANG
C
C              --- TYPE OF PATTERN CUT IS SET ---
C
            IF(ICUT.EQ.1) THEN
               IFE=1
               IFA=0
            ELSE
```

```
                IFE=0
                IFA=1
            ENDIF
C
C       --- OUTPUT DESIGNATION & ANGLE INCREMENT VALUE ---
C
            IF(PLTOUT.EQ.'Y'.OR.PLTOUT.EQ.'y') THEN
                IPFE=1
                IPFA=1
            ELSE
                IPFE=0
                IPFA=0
            ENDIF
C
            FNDFE=ANGINC
            FNDFA=ANGINC
C
        End if
      End if
C
C               --- INPUT FREQUENCY ---
C
  175 WRITE(*,180)
  180 FORMAT(1X,'FREQUENCY IN MEGAHERTZ?')
      READ(*,*) FMC
C
C       --- INPUT WIRE CONDUCTIVITY & RADIUS IF WIRES USED ---
C
      IF(USEWRS.EQ.'N'.OR.USEWRS.EQ.'n') then
C
C
C       --- SUPPLY FICTITIOUS NUMBERS FOR WIRE PARAMETERS ---
C
  210    CMM=-1.0
         A=0.001
C
      Else
         WRITE(*,190)
  190    FORMAT(1X,'WIRE CONDUCTIVITY IN MEGAMHOS/METER?')
         READ(*,*) CMM
         WRITE(*,200)
  200    FORMAT(1X,'WIRE RADIUS IN METERS?')
         READ(*,*) A
      End if
C
  215 IF( USEPLT .EQ. 'Y' .OR. USEPLT .EQ. 'y') Then
C
C               --- INPUT PLATE INFORMATION ---
C
         WRITE(*,220)
  220    FORMAT(/1X,'HOW MANY PLATES?')
         READ(*,*) NPLTS
C
         WRITE(*,225)
  225    FORMAT(/1X,'INPUT TYPE OF PLATE TEST MODE:'/10X,
                '0  FULL-SURFACE PATCH MODE'/10X,
                '1 FILAMENTARY TEST MODE')
         READ(*,*) IFIL
C
         WRITE(*,230)
```

```
  230    FORMAT(/1X,'PLATE INFORMATION --'/)
C
         DO 420 NPL=1,NPLTS
            CALL PLATE( NPL, NCNRS, SEGM, IREC, IPN, IGS, PCN, ICHG,
                        Zsht)
  420    CONTINUE
       End if
C
C                --- REUSE OF IMPEDANCE MATRIX ---
C      Disk storage of impedance is not anticipated; therefore:
       IWRZM=0
       IRDZM=0
C
       IF( USEWRS .EQ. 'Y' .OR. USEWRS .EQ. 'y') Then
C
C                  --- INPUT WIRE INFORMATION ---
          CALL WIRE(NM,NP,NAT,NFPT,NSF1,NSF2,ICHG)
          CALL WRPTS(I,NP,X,Y,Z,ICHG)
          CALL ENDPT(NM,IA,IB,ICHG)
          CALL FPPLT(NFPT,IFM,IABFP,VLG,ZL,ICHG)
C
          DO 700 I=1,NAT
             CALL FPNPL(I,NAS,IABAP,NPLA,VGA,ZLDA,BDSK,ICHG)
  700     CONTINUE
       End if
C
C           --- PARAMETER VALUES WRITTEN TO DATA FILE ---
C
  810  WRITE(10,100) NGO, NPRINT, NRUNS, NWGS, IWR, IWRZT, INT, INTP,
                     INTD, INWR, IRGM, IFIL, Rf, Indzi
  100  FORMAT(2I2,2I4,2I2,3I4,3I2,F6.2,I4)
       If( Indzi .ne. 0) Then
          Write(10,*) Fmc1, Fmc2, Dfzi, Dff, Irs12, Thrd, Phrd,
                      Thinc, Phinc
       End if
       If( Indzi .eq. 0) Then
          WRITE(10,910) IFE, IPFE, FNDFE, PHFE
          WRITE(10,910) IFA, IPFA, FNDFA, THFA
          WRITE(10,910) ISE, IPSE, FNDSE, PHSE, THIN, PHIN
          WRITE(10,910) ISA, IPSA, FNDSA, THSA
          WRITE(10,920) FMC, CMM, A
       End if
  910  Format(2I2,4F6.2)
  920  Format(2F8.2,F8.4)
C
       IF(USEPLT.EQ.'N'.OR.USEPLT.EQ.'n') Then
C
          WRITE(10,*) 0
          WRITE(10,*) 4, 0.2, 1, 3, 0
          DO I=1,4
             WRITE(10,*) 1.0, 1.0, 1.0
          End do
       Else
          WRITE(10,*) NPLTS
C
          DO NPL=1,NPLTS
             WRITE(10,930) NCNRS(NPL), SEGM(NPL), IREC(NPL), IPN(NPL),
                           IGS(NPL), zsht(Npl)
  930        Format(I3,F6.3,3I3,' (',F6.2,',',F6.2,') ')
             DO NCNR=1,NCNRS(NPL)
```

```
                  WRITE(10,940) PCN(1,NCNR,NPL), PCN(2,NCNR,NPL),
                                PCN(3,NCNR,NPL)
 940              Format(3F7.2)
               End do
            End do
         End if
C
C        --- INPUT FILLER NUMBERS FOR NO-PLATES CASE ---
C
C
         WRITE(10,*) IWRZM,IRDZM
C
C     --- DO NOT WRITE WIRE INFORMATION TO FILE FOR NO-WIRES CASE ---
C
         If( Inwr .ne. 0 .and. Irgm .ne. 0) Then
C
            WRITE(10,950) NM,NP,NAT,NFPT,NFS1,NFS2
 950        Format(6I3)
C
            DO I=1,NP
               WRITE(10,960) X(I),Y(I),Z(I)
            End do
 960        Format(3F8.3)
C
            DO I=1,NM
               WRITE(10,950) IA(I),IB(I)
            End do
C
            IF( NFPT .GT. 0) Then
               DO I=1,NFPT
                  WRITE(10,970) IFM(I),IABFP(I),VLG(I),ZL(I)
               End do
            Else
               WRITE(10,970) 1,0,(1.,0.),(1.,0.)
            End if
970         Format(2I3,' (',F6.2,',',F6.2,') ',' (',F6.2,',',F6.2,') ')
C
C
            DO I=1,NAT
               WRITE(10,980) NAS(I),IABAP(I),NPLA(I),VGA(I),ZLDA(I),BDSK(I)
            End do
980         Format(3I3,' (',F6.2,',',F6.2,') ','(',F6.2,',',F6.2,') ',F8.3)
C
         End if
C
         END
c
c
         Subroutine Rchange ( Variable, Value, New)
c
         Character*1     Change
         Character*40    Variable
         Real*4          Value
         Integer*4       New
c
         n = 1
         Do while( Variable (n:n+1) .ne. '  ')
            n = n + 1
         End do
         n = n - 1
```

```
C
      if( New .eq. 0) then
         Write(*,10) Variable(1:n), Value
 10      Format(1H$,'The current value of ',A<n>,' is ',F8.2)
C
         Write(*,30)
 30      Format(1H$,'Would you like to change this value? ' )

         Read(*,50) Change
      Else
         Change = 'y'
      End if
C
      If( Change .eq. 'y' .or. Change .eq. 'Y') Then
         Write(*,40) Variable(1:n)
 40      Format(1H$,'Please enter the new value for ',A<n>,' ')
         READ(*,*) Value
      End if

      Write(*,100)
 50   Format(A1)
100   Format(/)
      Return
      End
C
C
      Subroutine Cchange ( Variable, Value, New)
C
      Character*1     Change
      Character*40    Variable
      Complex*8       Value
      Integer*4       New
C
      n = 1
      Do while( Variable (n:n+1) .ne. '  ')
         n = n + 1
      End do
      n = n - 1
C
      if( New .eq. 0) then
         Write(*,10) Variable(1:n), Value
 10      Format(1H$,'The current value of ',A<n>,' is ',
                '(',F6.2,',',F6.2,')')
C
         Write(*,30)
 30      Format(1H$,'Would you like to change this value? ' )

         Read(*,50) Change
      Else
         Change = 'y'
      End if
C
      If( Change .eq. 'y' .or. Change .eq. 'Y') Then
         Write(*,40) Variable(1:n)
 40      Format(1H$,'Please enter the new value for ',A<n>,' ')
         READ(*,*) Value
      End if

      Write(*,100)
 50   Format(A1)
```

```
100   Format(/)
      Return
      End
C
C
      Subroutine Ichange ( Variable, Ivalue, New)
C
      Character*1     Change
      Character*40    Variable
      Integer*4       New
      Integer*4       Ivalue
C
      n = 1
      Do while( Variable (n:n+1) .ne. '  ' .and. n.lt.40)
         n = n + 1
      End do
      n = n - 1
C
      if( New .eq. 0) then
         Write(*,10) Variable(1:n), Ivalue
 10      Format(1H$,'The current value of ',A<n>,' is ',I5)
C
         Write(*,30)
 30      Format(1H$,'Would you like to change this value? ' )

         Read(*,50) Change
      Else
         Change = 'y'
      End if
C
      If( Change .eq. 'y' .or. Change .eq. 'Y') Then
         Write(*,40) Variable(1:n)
 40      Format(1H$,'Please enter the new value for ',A<n>,' ')
         READ(*,*) Ivalue
      End if

      Write(*,100)
 50   Format(A1)
100   Format(/)
      Return
      End
C
C
      SUBROUTINE PLATE( NPL, NCNRS, SEGM, IREC, IPN, IGS, PCN, ICHG,
     .                  Zsht)
C
      DIMENSION NCNRS(10),SEGM(10),IREC(10),IGS(10),IPN(10),PCN(3,10,10)
      Complex*8 Zsht(10)
      Character*40 Variable
C
      WRITE(*,240) NPL
  240 FORMAT(1X,'For PLATE NUMBER ',I3)
C
      Variable = 'Number of Corners'
      CALL ICHaNGE( Variable, NCNRS(NPL), ICHG)

      Variable = 'Size of Patch'
      CALL RCHaNGE( Variable, SEGM(NPL), ICHG)

      WRITE(*,254)
```

```
  254 FORMAT(1X,'IDENTIFY SHAPE OF PLATE -'/
     .       10X,'0  POLYGONAL'/
     .       10X,'1  RECTANGULAR'/)
      Variable = 'Shape of Plate'
      CALL ICHaNGE( Variable, IREC(NPL), ICHG)

C
C       *** POLARIZATION SELECTION IS SET TO 3 FOR ALL CASES ***
      IPN(NPL)=3
C
      WRITE(*,256)
  256 FORMAT(1X,'WHICH IS THE GENERATING SIDE?')
      Variable = 'The Generating Side'
      CALL ICHaNGE( Variable, IGS(NPL), ICHG)
C
      Write(*,500)
  500 Format(1X,'The complex Sheet Impedance must be set to zero ',
     .          'if the plate contains'/,' a wire/plate attachment')
      Variable = 'Sheet Impedance'
      Call CChange( Variable, Zsht(Npl), Ichg)
c
      WRITE(*,260)
  260 FORMAT(/1X,'INPUT POSITION OF EACH CORNER (X, Y, Z)')
C
      DO 275 NCNR=1,NCNRS(NPL)
         WRITE(*,270) NCNR
  270    FORMAT(1X,'NO.  ',I3)
         Variable = 'X coordinate'
         CALL RCHaNGE( Variable, PCN(1,NCNR,NPL), Ichg)
         Variable = 'Y coordinate'
         Call Rchange( Variable, PCN(2,NCNR,NPL), Ichg)
         Variable = 'Z coordinate'
         Call Rchange( Variable, PCN(3,NCNR,NPL), ICHG)
  275 CONTINUE
      RETURN
      END
c
c
C
      SUBROUTINE WIRE(NM,NP,NAT,NFPT,NSF1,NSF2,ICHG)
c
      Character*40 Variable
      Character*1 Mcoupl
c
      Variable = 'the Number of Wire Segments'
      CALL ICHaNGE(Variable, NM, ICHG)
c
      Variable = 'Total No. of Points in Wire Structure'
      CALL ICHaNGE(Variable, NP, ICHG)
c
      Variable = 'Number of Wire-to-Plate attachment pts.'
      CALL ICHaNGE(Variable, NAT, ICHG)
c
      Variable = 'Number of Feed Pts. in Wire Structure'
      CALL ICHaNGE(Variable, NFPT, ICHG)
c
      WRITE(*,470)
C
  470 FORMAT(1H$,'IS MUTUAL COUPLING TO BE COMPUTED?')
      READ(*,50) MCOUPL
```

```
   50 FORMAT(A1)
      IF( MCOUPL .EQ. 'Y' .OR. MCOUPL .EQ. 'y') Then
         Variable = 'location of First Feed Port'
         CALL ICHaNGE(Variable, NSF1, ICHG)
C
         Variable = 'location of Second Feed Port'
         CALL ICHaNGE(Variable, NSF2, ICHG)
      Else
         NSF1=0
         NSF2=0
      End if
      RETURN
      END
C
C
C
      SUBROUTINE WRPTS(I,NP,X,Y,Z,ICHG)
      DIMENSION X(10),Y(10),Z(10)
C
      Character*40 Variable
C
      WRITE(*,520)
  520 FORMAT(/1X,'INPUT LOCATIONS (X,Y,Z) OF WIRE POINTS:')
      DO 530 I=1,NP
         WRITE(*,525) I
  525    FORMAT(1X,'For POINT ',I3)
         Variable = 'X coordinate'
         Call Rchange( Variable, X(i), ichg)
         Variable = 'Y coordinate'
         Call Rchange( Variable, Y(i), ichg)
         Variable = 'Z coordinate'
         CALL RCHaNGE( Variable, Z(I), ICHG)
  530 CONTINUE
      RETURN
      END
C
C
C
      SUBROUTINE ENDPT(NM,IA,IB,ICHG)
      DIMENSION IA(10),IB(10)
C
      Character*40 Variable
C
      WRITE(*,535)
  535 FORMAT(/1X,'INPUT ENDPOINT A & ENDPOINT B OF EACH WIRE SEGMENT:'/)
      DO 550 J=1,NM
         WRITE(*,540) J
  540    FORMAT(1X,'For SEGMENT ',I3)
         Variable = 'Endpoint A'
         CALL ICHaNGE( Variable, IA(J), ICHG)
         Variable = 'Endpoint B'
         Call Ichange( Variable, IB(j), Ichg)
  550 CONTINUE
      RETURN
      END
C
C
C
      SUBROUTINE FPPLT(NFPT,IFM,IABFP,VLG,ZL,ICHG)
      DIMENSION IFM(10),IABFP(10),VLG(10),ZL(10)
```

```
      COMPLEX VLG
      COMPLEX ZL
C
      Character*40 Variable
C
      WRITE(*,560)
  560 FORMAT(/1X,'INPUT LOCATIONS OF FEED POINTS:'/)
      DO 650 I=1,NFPT
         WRITE(*,570) I
  570    FORMAT(1X,'FEED POINT NO.  ',I3,' is in which Wire Segment?')
         Variable = 'Wire Segment'
         CALL ICHaNGE( Variable, IFM(I), ICHG)
         WRITE(*,580)
  580    FORMAT(1X,'AT WHICH END OF SEGMENT?'/10X,
     .             '0  ENDPOINT A'/10X,
     .             '1  ENDPOINT B'/)
         Variable = 'Endpoint'
         CALL ICHaNGE( Variable, IABFP(I), ICHG)
C
         Variable = 'Generator Complex Voltage at Feed Point'
         Call Cchange( Variable, VLG(I), Ichg)
C
         Variable = 'Complex Load Impedance (ohms)'
         Call Cchange( Variable, ZL(I), Ichg)
C
  650 CONTINUE
      RETURN
      END
C
C
C
      SUBROUTINE FPNPL(NAT,NAS,IABAP,NPLA,VGA,ZLDA,BDSK,ICHG)
      DIMENSION NAS(10),IABAP(10),NPLA(10),BDSK(10)
C
      Complex Zlda(1), Vga(1)
      Character*40 Variable
C
      IF(NAT.EQ.0) Return
C
      DO 700 I=1,NAT
         Variable = 'No. of wire segments attached to plate'
         CALL ICHaNGE( Variable, NAS(I), ICHG)
C
         WRITE(*,660)
  660    FORMAT(/1X,'ENDPOINT ATTACHED TO PLATE:'/10X,
     .             '0  POINT A'/10X,
     .             '1  POINT B')
         Variable = 'Endpoint'
         CALL ICHaNGE( Variable, IABAP(I), ICHG)
C
         Variable = 'Plate Number'
         CALL ICHaNGE( Variable, NPLA(I), ICHG)
C
         Variable = 'Attachment Pt Complex Generator Voltage'
         Call Cchange( Variable, VGA(I), Ichg)
C
         Variable = 'Attachment Point Complex Load Impedance'
         Call Cchange( Variable, ZLDA(I), Ichg)
C
         Variable = 'Disk Radius (meters)'
```

```
        Call Rchange( Variable, Bdsk(i), Ichg)
700 CONTINUE
    RETURN
    END
```

APPENDIX B

GRAPHICS CODES

```
      PROGRAM PLTGTD
      IMPLICIT INTEGER*2(I-N)
      DIMENSION ANG(600)
      DIMENSION AMP(600)
      DIMENSION ASTEP(5)
      DIMENSION XPLT(361,7)
      CHARACTER*1 Z,CHCUT,HOLDIT,NEXTP
C
      OPEN(2,FILE='PRN')
C
      WRITE(*,70)
   70 FORMAT(1X,'THE PATTERN IS READY TO BE PLOTTED'//)
C
C     INPUT THE TYPE OF POLARIZATION COMPONENT DESIRED
C
 1000 WRITE(*,100)
  100 FORMAT(1X,'ENTER POLARIZATION COMPONENT DESIRED:'/5X,'1 -  E-THETA
     .'/5X,'2 -  E-PHI',/5X,'3 -  MAJOR AXIS OF POLARIZATION ELLIPSE'/5X
     .,'4 -  MINOR AXIS OF POLARIZATION ELLIPSE'/5X,'5 -  TOTAL AMPLITUD
     .E')
      READ(*,*) IPOL
C
C     ASSOCIATE DATA COLUMNS WITH POLARIZATION DESIGNATIONS
C
      IF(IPOL.EQ.1) KPOL=6
      IF(IPOL.EQ.2) KPOL=7
      IF(IPOL.EQ.3) KPOL=3
      IF(IPOL.EQ.4) KPOL=4
      IF(IPOL.EQ.5) KPOL=5
C
      JCL1=68
      JCL2=566
      JROW1=30
      JROW2=150
C
      MINOR=1
      LABEL=1
      NDEC=0
      IOPT=0
C
C     READ DATA FROM FILE PLOT.DAT TO ARRAY XPLT
C
      NUMPNT=0
      OPEN(UNIT=1,FILE='PLOT.DAT',STATUS='OLD')
      DO 150 I=1,361
      READ(1,120,END=200) XPLT(I,1),XPLT(I,2),XPLT(I,3),XPLT(I,4),XPLT(I
     2,5),XPLT(I,6),XPLT(I,7)
  120 FORMAT(1X,5(3X,F7.2),2(6X,F7.2))
C
C     COUNT POINTS AS READING PROCEEDS
      NUMPNT=NUMPNT+1
C
  150 CONTINUE
  200 CONTINUE
C
```

```
C     DETERMINE TYPE OF PATTERN CUT (GREAT CIRCLE OR CONICAL)
C        CHCUT='Y' FOR GREAT-CIRCLE CUT; CHCUT='N' FOR CONICAL CUT
C
      VAL1=XPLT(2,1)
      VAL2=XPLT(1,1)
      CHCUT='N'
      IF(ABS(VAL2-VAL1).GT.0.01) CHCUT='Y'
C
C     SET MINIMUM AND MAXIMUM VALUES OF ANGLE
C
      IF(CHCUT.EQ.'Y') THEN
      AFST=XPLT(1,1)
      ALST=(XPLT(3,1)-XPLT(2,1))*(NUMPNT-1)+XPLT(1,1)
      NCUT=1
      ENDIF
C
      IF(CHCUT.EQ.'N') THEN
      AFST=XPLT(1,2)
      ALST=XPLT(NUMPNT,2)
      NCUT=2
      ENDIF
C
      ANGDIF=XPLT(2,NCUT)-XPLT(1,NCUT)
C
C     TRANSFER DATA FROM XPLT ARRAY TO ANG AND AMP ARRAYS
C
      ANG(0)=0.0
C
      DO 80 I=1,NUMPNT
      ANG(I)=XPLT(I,NCUT)
      IF(ANG(I).LT.ANG(I-1)) THEN
      ANG(I)=ANG(I-1)+ANGDIF
      ENDIF
      AMP(I)=XPLT(I,KPOL)
      IF(AMP(I).LT.-50.) AMP(I)=-50.
   80 CONTINUE
C
C     FIND AMPMIN AND AMPMAX VALUES -
C
      AMPMIN=-10.
      AMPMAX=10.
C
      DO 20 I=1,NUMPNT
      IF(AMP(I).GT.AMPMAX) AMPMAX=AMP(I)
      IF(AMP(I).LT.AMPMIN) AMPMIN=AMP(I)
   20 CONTINUE
C
   25 ASPECT=(ALST-AFST)/(AMPMAX-AMPMIN)
C
C     ROUND OFF AMPMIN & AMPMAX VALUES -
C
      AMPMIN=INT(AMPMIN)
      AMPMAX=INT(AMPMAX)
C
C     DETERMINE BEST ANGLE AND AMPLITUDE INCREMENTS TO USE -
```

```
C
      XINC=(ALST-AFST)/10.
      YINC=(AMPMAX-AMPMIN)/10.
C
      ASTEP(1)=1.
      ASTEP(2)=2.
      ASTEP(3)=5.
      ASTEP(4)=10.
      ASTEP(5)=30.
C
      DO 30 I=1,5
      XSTEP=ASTEP(I)
      IF(XSTEP.GT.XINC) GOTO 40
   30 CONTINUE
   40 XMAJOR=XSTEP
C
      DO 50 I=1,4
      YSTEP=ASTEP(I)
      IF(YSTEP.GT.YINC) GOTO 55
   50 CONTINUE
   55 YMAJOR=YSTEP
      ISTEP=INT(YSTEP)
C
      NSTEP=AMPMIN/ISTEP
      IF(AMPMIN.LT.-50.) GOTO 58
      AMPMIN=NSTEP*ISTEP-ISTEP
      GOTO 350
   58 AMPMIN=NSTEP*ISTEP
C
  350 CALL QSETUP(0,3,-2,0)
C
   60 CALL QPLOT(JCL1,JCL2,JROW1,JROW2,AFST,ALST,AMPMIN,AMPMAX,AFST,AMPM
     1IN,IOPT,1.,ASPECT)
      CALL QSMODE(6)
C
      YMAX=AMPMAX+0.1*(AMPMAX-AMPMIN)
      CALL QXAXIS(AFST,ALST,XMAJOR,MINOR,LABEL,NDEC)
      CALL QYAXIS(AMPMIN,YMAX,YMAJOR,MINOR,LABEL,NDEC)
      IF(CHCUT.EQ.'Y') GOTO 300
      CALL QPTXTA(12,'PHI, DEGREES',2)
      GOTO 400
  300 CALL QPTXTA(14,'THETA, DEGREES',2)
  400 CALL QPTXTC(4,'DB',2)
C
      J1=1
C
      CALL QTABL(J1,NUMPNT,ANG,AMP)
C
      CLOSE(1)
      CLOSE(2)
C
      READ(*,600) HOLDIT
  600 FORMAT(A1)
C
      WRITE(*,700)
```

```
700 FORMAT(////////////////////)
    WRITE(*,75)
 75 FORMAT(1X,'DO YOU WANT TO MAKE ANOTHER PLOT?  Y OR N')
    READ(*,600) NEXTP
    IF(NEXTP.EQ.'Y'.OR. NEXTP.EQ.'y') GOTO 1000
    STOP
    END
```

```
      PROGRAM PLTMM
      IMPLICIT INTEGER*2(I-N)
      DIMENSION ANG(600)
      DIMENSION AMP(600)
      DIMENSION ASTEP(5)
      DIMENSION XPLT(361,7)
      CHARACTER*1 Z,HOLDIT,NEXTP
C
      OPEN(2,FILE='PRN')
C
      WRITE(*,70)
   70 FORMAT(1X,'THE PATTERN IS READY TO BE PLOTTED'//)
C
C     INPUT THE TYPE OF POLARIZATION COMPONENT DESIRED
C
 1000 WRITE(*,100)
  100 FORMAT(1X,'ENTER POLARIZATION COMPONENT DESIRED:'/5X,'1 -  E-THETA
     .'/5X,'2 -  E-PHI'/)
      READ(*,*) IPOL
C
C     ASSOCIATE DATA COLUMNS WITH POLARIZATION DESIGNATIONS
C
      IF(IPOL.EQ.1) KPOL=2
      IF(IPOL.EQ.2) KPOL=4
C
      JCL1=68
      JCL2=566
      JROW1=30
      JROW2=150
C
      MINOR=1
      LABEL=1
      NDEC=0
      IOPT=0
C
C     READ DATA FROM FILE PLOT.DAT TO ARRAY XPLT
C
      NUMPNT=0
      OPEN(UNIT=1,FILE='PLOT.DAT',STATUS='OLD')
      DO 150 I=1,361
      READ(1,120,END=200) JANG,XPLT(I,2),XPLT(I,3),XPLT(I,4),XPLT(I,5)
  120 FORMAT(I4,3(1X,F5.1,1X,F6.1))
      XPLT(I,1)=REAL(JANG)
C
C     COUNT POINTS AS READING PROCEEDS
      NUMPNT=NUMPNT+1
C
  150 CONTINUE
  200 CONTINUE
C
C     SET MINIMUM AND MAXIMUM VALUES OF ANGLE
C
      AFST=XPLT(1,1)
      ALST=(XPLT(3,1)-XPLT(2,1))*REAL(NUMPNT-1)+XPLT(1,1)
      NCUT=1
C
```

```
      ADIF=XPLT(2,1)-XPLT(1,1)
C
C     TRANSFER DATA FROM XPLT ARRAY TO ANG AND AMP ARRAYS
C
      ANG(0)=0.0
C
      DO 80 I=1,NUMPNT
      ANG(I)=XPLT(I,1)
      IF(ANG(I).LT.ANG(I-1)) THEN
      ANG(I)=ANG(I-1)+ADIF
      ENDIF
      AMP(I)=XPLT(I,KPOL)
      IF(AMP(I).LT.-50.) AMP(I)=-50.
   80 CONTINUE
C
C     FIND AMPMIN AND AMPMAX VALUES -
C
      AMPMIN=-10.
      AMPMAX=10.
C
      DO 20 I=1,NUMPNT
      IF(AMP(I).GT.AMPMAX) AMPMAX=AMP(I)
      IF(AMP(I).LT.AMPMIN) AMPMIN=AMP(I)
   20 CONTINUE
C
   25 ASPECT=(ALST-AFST)/(AMPMAX-AMPMIN)
C
C     ROUND OFF AMPMIN & AMPMAX VALUES -
C
      AMPMIN=INT(AMPMIN)
      AMPMAX=INT(AMPMAX)
C
C     DETERMINE BEST ANGLE AND AMPLITUDE INCREMENTS TO USE -
C
      XINC=(ALST-AFST)/10.
      YINC=(AMPMAX-AMPMIN)/10.
C
      ASTEP(1)=1.
      ASTEP(2)=2.
      ASTEP(3)=5.
      ASTEP(4)=10.
      ASTEP(5)=30.
C
      DO 30 I=1,5
      XSTEP=ASTEP(I)
      IF(XSTEP.GT.XINC) GOTO 40
   30 CONTINUE
   40 XMAJOR=XSTEP
C
      DO 50 I=1,4
      YSTEP=ASTEP(I)
      IF(YSTEP.GT.YINC) GOTO 55
   50 CONTINUE
   55 YMAJOR=YSTEP
      ISTEP=INT(YSTEP)
```

```
C
      NSTEP=AMPMIN/ISTEP
      IF(AMPMIN.LT.-50.) GOTO 58
      AMPMIN=NSTEP*ISTEP-ISTEP
      GOTO 350
   58 AMPMIN=NSTEP*ISTEP
C
  350 CALL QSETUP(0,3,-2,0)
C
   60 CALL QPLOT(JCL1,JCL2,JROW1,JROW2,AFST,ALST,AMPMIN,AMPMAX,AFST,AMPM
     1IN,IOPT,1.,ASPECT)
      CALL QSMODE(6)
C
      YMAX=AMPMAX+0.1*(AMPMAX-AMPMIN)
      CALL QXAXIS(AFST,ALST,XMAJOR,MINOR,LABEL,NDEC)
      CALL QYAXIS(AMPMIN,YMAX,YMAJOR,MINOR,LABEL,NDEC)
  300 CALL QPTXTA(14,'ANGLE, DEGREES',2)
  400 CALL QPTXTC(4,'DB',2)
C
      J1=1
C
      CALL QTABL(J1,NUMPNT,ANG,AMP)
C
      CLOSE(1)
      CLOSE(2)
C
      READ(*,600) HOLDIT
  600 FORMAT(A1)
C
      WRITE(*,700)
  700 FORMAT(////////////////////)
      WRITE(*,75)
   75 FORMAT(1X,'DO YOU WANT TO MAKE ANOTHER PLOT?  Y OR N')
      READ(*,600) NEXTP
      IF(NEXTP.EQ.'Y') GOTO 1000
      END
```