

AD-A200 315

JUNCTION CODE USER'S MANUAL ELECTROMAGNETIC SCATTERING
AND RADIATION BY G. (U) HOUSTON UNIV ELECTROMAGNETICS

1/3

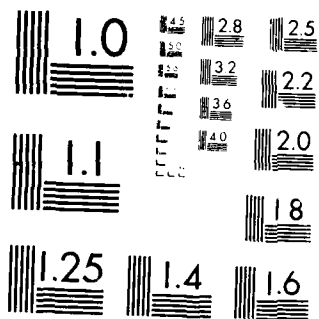
UNCLASSIFIED

LAB D R MILTON ET AL. AUG 88 TR-87-18 NOSC-ID-1324

F/G 20/14

NL

NS6001-85-D-0203



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD-A200 315

Technical Document 1324
August 1988

Junction Code User's Manual

Electromagnetic Scattering and
Radiation by Arbitrary
Configurations of Conducting
Bodies and Wires

D. R. Wilton
S. U. Hwu

Applied Electromagnetics Laboratory
Department of Electrical Engineering
University of Houston

DTIC
ELECTE
OCT 07 1988
S D
H

Approved for public release; distribution is unlimited.

The views and conclusions contained in this report are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Naval Ocean Systems Center or the U.S. Government.

38 10 0 000

NAVAL OCEAN SYSTEMS CENTER
San Diego, California 92152-5000

E. G. SCHWEIZER, CAPT, USN
Commander

R. M. HILLYER
Technical Director

ADMINISTRATIVE INFORMATION

This report was prepared by Applied Electromagnetics Laboratory, Department of Electrical Engineering, University of Houston, for Code 822 of the Naval Ocean Systems Center.

Released by
I. C. Olson, Head
Antenna and RF Systems
Integration Branch

Under authority of
G. E. Ereckson, Head
Shipboard Systems
Division

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED		1b. RESTRICTIVE MARKINGS	
2a. SECURITY CLASSIFICATION AUTHORITY		3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.	
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE			
4. PERFORMING ORGANIZATION REPORT NUMBER(S) Technical Report Number 87-18		5. MONITORING ORGANIZATION REPORT NUMBER(S) NOSC TD 1324	
6a. NAME OF PERFORMING ORGANIZATION Applied Electromagnetics Laboratory	6b. OFFICE SYMBOL (if applicable)	7a. NAME OF MONITORING ORGANIZATION Naval Ocean Systems Center	
6c. ADDRESS (City, State and ZIP Code) Department of Electrical Engineering University of Houston		7b. ADDRESS (City, State and ZIP Code) San Diego, CA 92152-5000	
8a. NAME OF FUNDING/SPONSORING ORGANIZATION David Taylor Research Center	8b. OFFICE SYMBOL (if applicable) DTNSRDC	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N66001-85-D-0203	
6c. ADDRESS (City, State and ZIP Code) Bethesda, MD 20084		10. SOL I-ICE OF FUNDING NUMBERS	
		PROGRAM ELEMENT NO. 62121N	PROJECT NO. CM41
11. TITLE (include Security Classification) JUNCTION CODE USER'S MANUAL Electromagnetic Scattering and Radiation by Arbitrary Configurations of Conducting Bodies and Wires			
12. PERSONAL AUTHOR(S) D. R. Wilton, S. U. Hwu			
13a. TYPE OF REPORT Final	13b. TIME COVERED FROM Oct 1987 TO Dec 1987	14. DATE OF REPORT (Year, Month, Day) August 1988	15. PAGE COUNT 205
16. SUPPLEMENTARY NOTATION			
17. COSATI CODES		18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)	
FIELD	GROUP		
			electromagnetic environment effects electromagnetic interference
19. ABSTRACT (Continue on reverse if necessary and identify by block number) This report gives a brief description of the computer program JUNCTION which results from application of numerical procedures.			
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT <input type="checkbox"/> UNCLASSIFIED/UNLIMITED <input checked="" type="checkbox"/> SAME AS RPT <input type="checkbox"/> DTIC USERS		21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a. NAME OF RESPONSIBLE PERSON J. C. Logan		22b. TELEPHONE (include Area Code) (619) 553-3780	22c. OFFICE SYMBOL Code 822

Contents



Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

1	Introduction	6
1.1	JUNCTION Code	6
1.2	Program Structure	8
2	Data Generation	14
2.1	Triangular Patch and Wire Modeling	14
2.2	Automatic Triangulation and Segmentation of Models	14
2.3	Specification of Input Data	16
2.4	Modeling Guidelines	20
2.5	Current Reference Directions	22
3	Example Problem	24
3.1	Geometry	24
3.2	Input Data	26
3.2.1	FOR002.DAT	26
3.2.2	FOR008.DAT	27
3.3	Output Data	28
3.3.1	NEC.DAT	28
3.3.2	IGU.NEC	29
3.3.3	FOR003.DAT	30
3.3.4	FOR009.DAT	34
3.3.5	FOR004.DAT	36
3.3.6	FOR010.DAT	37
3.3.7	FOR011.DAT	38
A	Program Listings	40
A.1	JUNCTION	41
A.2	DATGEN	48

A.3 READIN	50
A.4 BQUAD	50
A.5 CYLIND	52
A.6 CONE	55
A.7 DISK	57
A.8 SPHERE	59
A.9 KNIT	63
A.10 ATTACH	66
A.11 QUERY	68
A.12 CHANGE	69
A.13 PACKIT	72
A.14 WRIOUT	74
A.15 TSTDAT	75
A.16 XPROD	78
A.17 UNTVEC	78
A.18 FAEMUL	78
A.19 BFACVT	80
A.20 EOFCLR	81
A.21 WIRDAT	81
A.22 NECDAT	82
A.23 BODIN	85
A.24 WIRIN	89
A.25 FACMUL	92
A.26 ORTFAC	93
A.27 CLSBOD	96
A.28 FACOUT	97
A.29 ADBMUL	98
A.30 BODPAR	99
A.31 EDGFAC	102
A.32 SOLTN	103
A.33 WIRMUL	109
A.34 EDGMUL	111
A.35 BCUMUL	112
A.36 FACEDG	112
A.37 FACVTX	112
A.38 VTXCRD	113
A.39 WIRMAN	113

A.40 MTXWIR	115
A.41 ZWW	116
A.42 ZBW	123
A.43 ZBB	128
A.44 ZWB	135
A.45 PATTEN	141
A.46 BPATTN	143
A.47 WPATTN	147
A.48 CHARGE	150
A.49 QBODY	151
A.50 QWIRE	154
A.51 CPATT	157
A.52 SGQADS	158
A.53 SEGQAD	160
A.54 WCUMUL	161
A.55 NODMUL	162
A.56 WIROUT	163
A.57 FNDJUN	164
A.58 JUNFAC	165
A.59 JANGLE	166
A.60 JUNPAR	167
A.61 FACPAR	169
A.62 ADWMUL	171
A.63 POTBOD	172
A.64 PD3FAC	173
A.65 BIMAGE	176
A.66 BSWTCH	177
A.67 FACQAD	178
A.68 INTGRN	180
A.69 PS3FAC	182
A.70 CAS	185
A.71 JKFPAT	185
A.72 INTJUN	187
A.73 POTWIR	190
A.74 WIMAGE	191
A.75 WSWTCH	192
A.76 PDWSEG	193

A.77 CKTOT	195
A.78 CKSELF	196
A.79 CKRED	197
A.80 ELIC1K	198

List of Figures

1.1	Typical conducting wire and body configuration.	7
1.2	Structure of controlling program JUNCTION.	9
1.3	Subroutines called by DATGEN.	10
1.4	Subroutines called by SOLTN.	11
1.5	Subroutines called by POTBOD and POTWIR.	12
1.6	Block diagram depicting relationships between JUNCTION and NEEDS. . .	13
-		
2.1	Incident field geometrical parameters.	19
2.2	Examples of triangles embedded within triangles.	21
2.3	The relationship between face and edge orientations and current reference direction.	23
3.1	Geometry of example problem.	25

Chapter 1

Introduction

1.1 JUNCTION Code

This report gives a brief description of the computer program JUNCTION which results from the application of numerical procedures described in [1]. The program invokes the method of moments to solve a coupled electric field integral equation for the currents induced on an arbitrary configuration of perfectly conducting bodies and wires. An important feature of the code is its ability to treat wire-to-wire, surface-to-surface, and wire-to-surface junctions. Wires may be connected to surfaces at essentially arbitrary angles and may be attached to surface edges or vertices. Results obtained using this algorithm are in the form of electric current and charge densities and far field patterns.

Fig.1.1 depicts a typical conducting wire and body configuration which might be treated by JUNCTION. The theory leading to a numerical algorithm for treating such structures is described in [1]. Here we are mostly concerned with describing the format of the input data for specifying the problem geometry and excitation to JUNCTION. This data may be generated by any number of means - by using an outboard program specially written for a given geometry, by running an interactive geometry generation program such as IGUANA[2], or by entering data via a digitizing tablet. JUNCTION only requires that two input files exist which contain the following problem specifications:

1. Planar triangular patch model of surfaces. A completely specified surface model merely consists of a collection of numbered *vertices* (corners of the triangular patches) and their coordinates, and an *edge connection list* specifying which pair of vertices each numbered edge connects.
2. Piecewise linear segment model of wires. A completely specified wire model merely consists of a collection of numbered *nodes* and their coordinates, and a *segment connection list* specifying

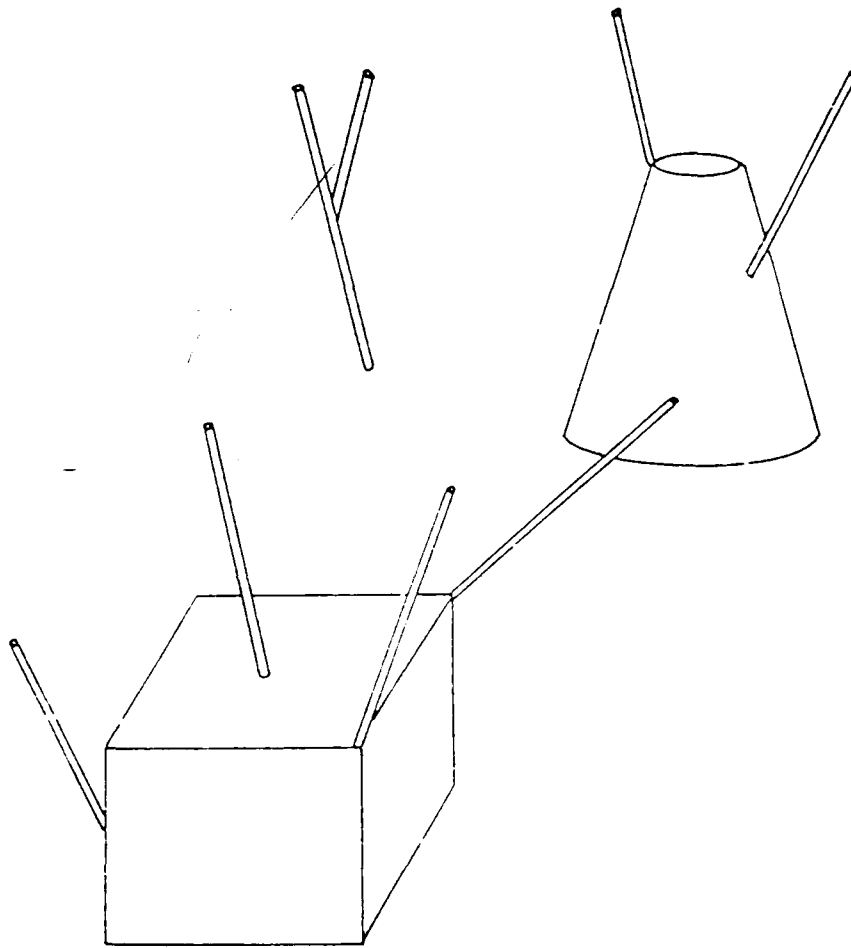


Figure 1.1: Typical conducting wire and body configuration.

which pair of nodes each numbered segment connects.

3. Excitation information such as frequency, angle of arrival, and polarization of incident plane waves, and location and magnitudes of voltage sources.
4. Ground plane and/or symmetry information requiring the specification locations and types of image planes.
5. Output control parameters.

A detailed description of the format of the input data may be found in Section 2.3.

1.2 Program Structure

Figs. 1.2-1.5 illustrates in block diagram form the dependence of the subroutines on their calling routines in JUNCTION. The figures group the subroutines roughly according to their function. For example, Fig. 1.2 shows the subroutines called by the main controller program, JUNCTION. The subroutines shown in Fig. 1.3 are all called by the input data generation subroutine, DATGEN. Subroutines shown in Fig. 1.4 are called by the subroutine SOLTN, which controls the matrix element computation and matrix solving processes. The main task in the matrix element computation step is the computation of potential integrals. This task is controlled by the subroutines POTBOD and POTWIR shown in Fig. 1.5. Listings for all subroutines in JUNCTION except CGESL and CGEFA (called by SOLTN) may be found in Appendix A. These latter two subroutines are contained in the LINPACK library [5] and may be replaced by any linear equation solving package if desired.

Fig. 1.6 shows the relationship between JUNCTION and the software package NEEDS (Numerical Electromagnetics Engineering Design System). It should be understood that JUNCTION is a stand-alone program, but that translators have been added to it to allow it to interface with NEEDS. Thus JUNCTION can read a file formatted in NEC (Numerical Electromagnetic Code) [3] format and generated by NEEDS. It can also translate JUNCTION formatted data to NEC formatted data for convenient display using the IGUANA (Interactive Graphics Utility for Automated NEC Analysis) software contained in NEEDS.

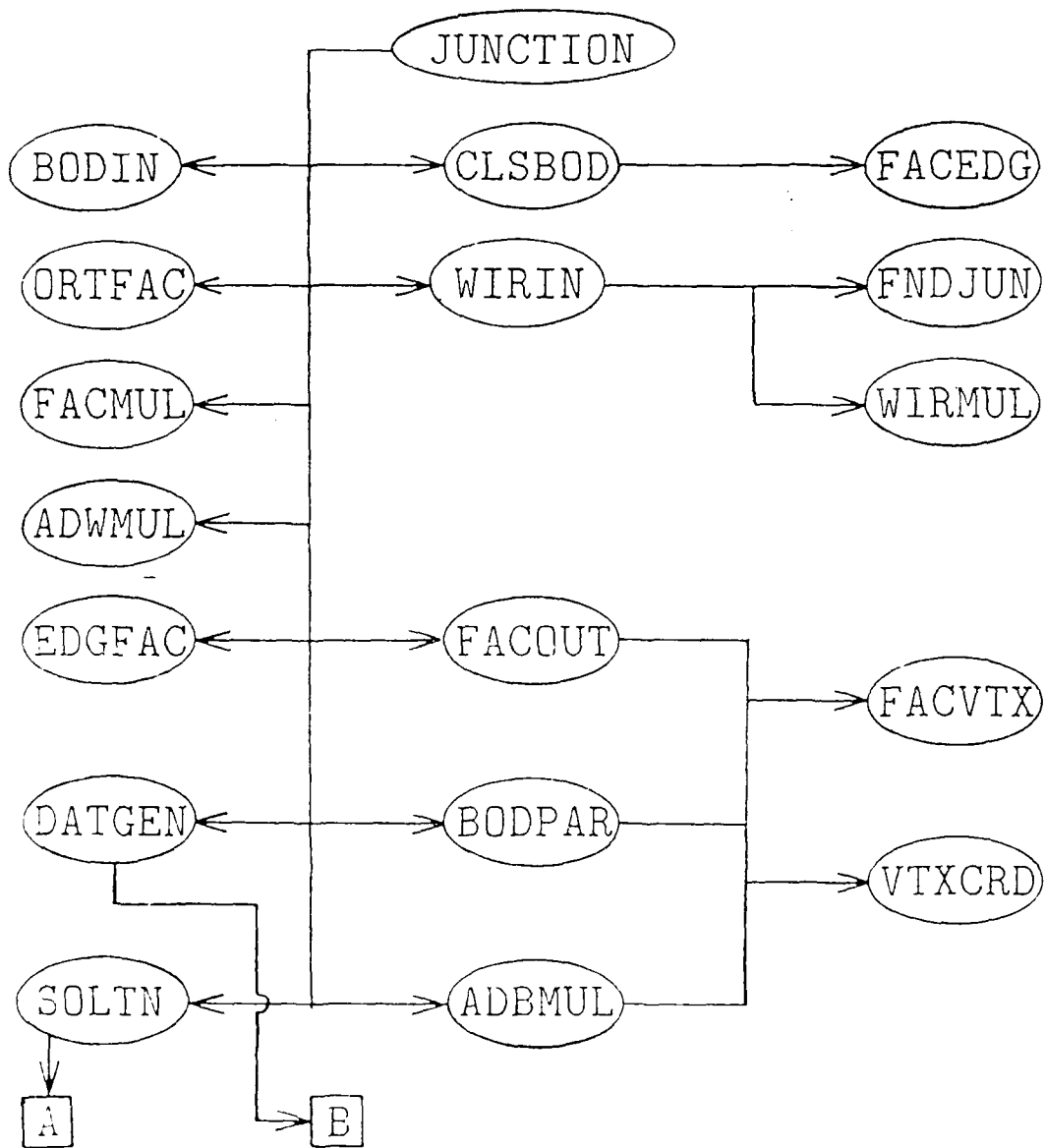


Figure 1.2: Structure of controlling program JUNCTION.

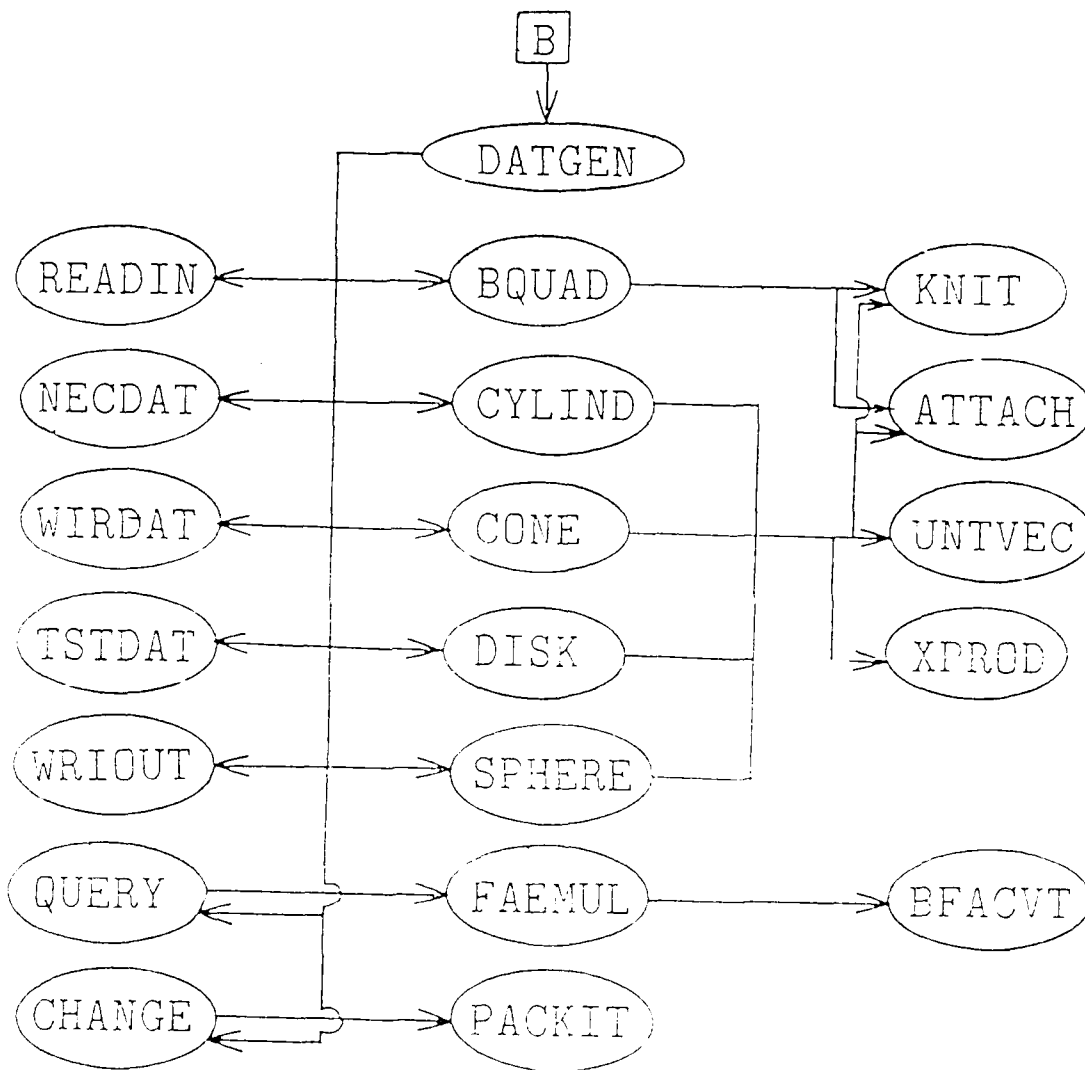


Figure 1.3: Subroutines called by DATGEN.

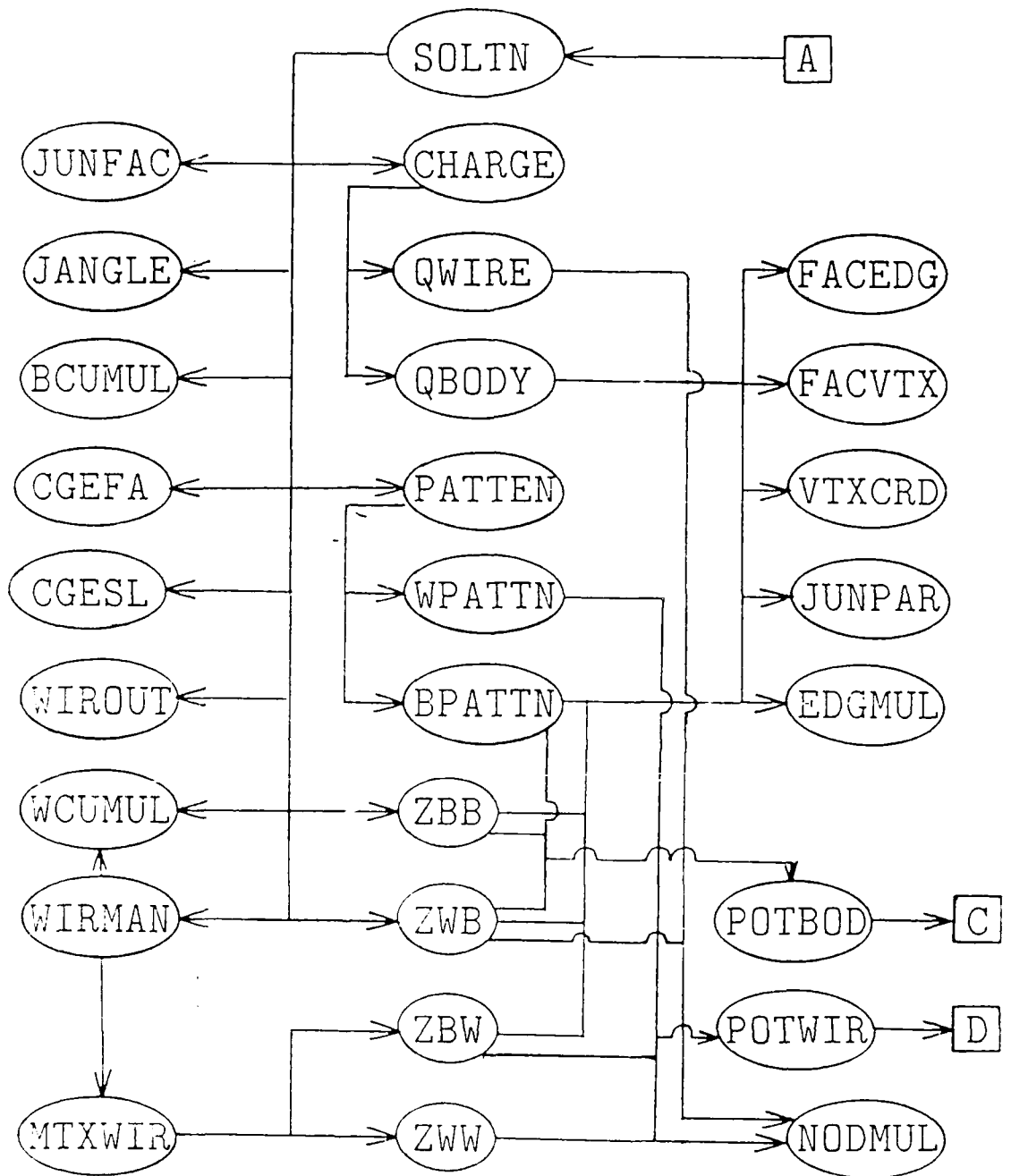


Figure 1.4: Subroutines called by SOLTN.

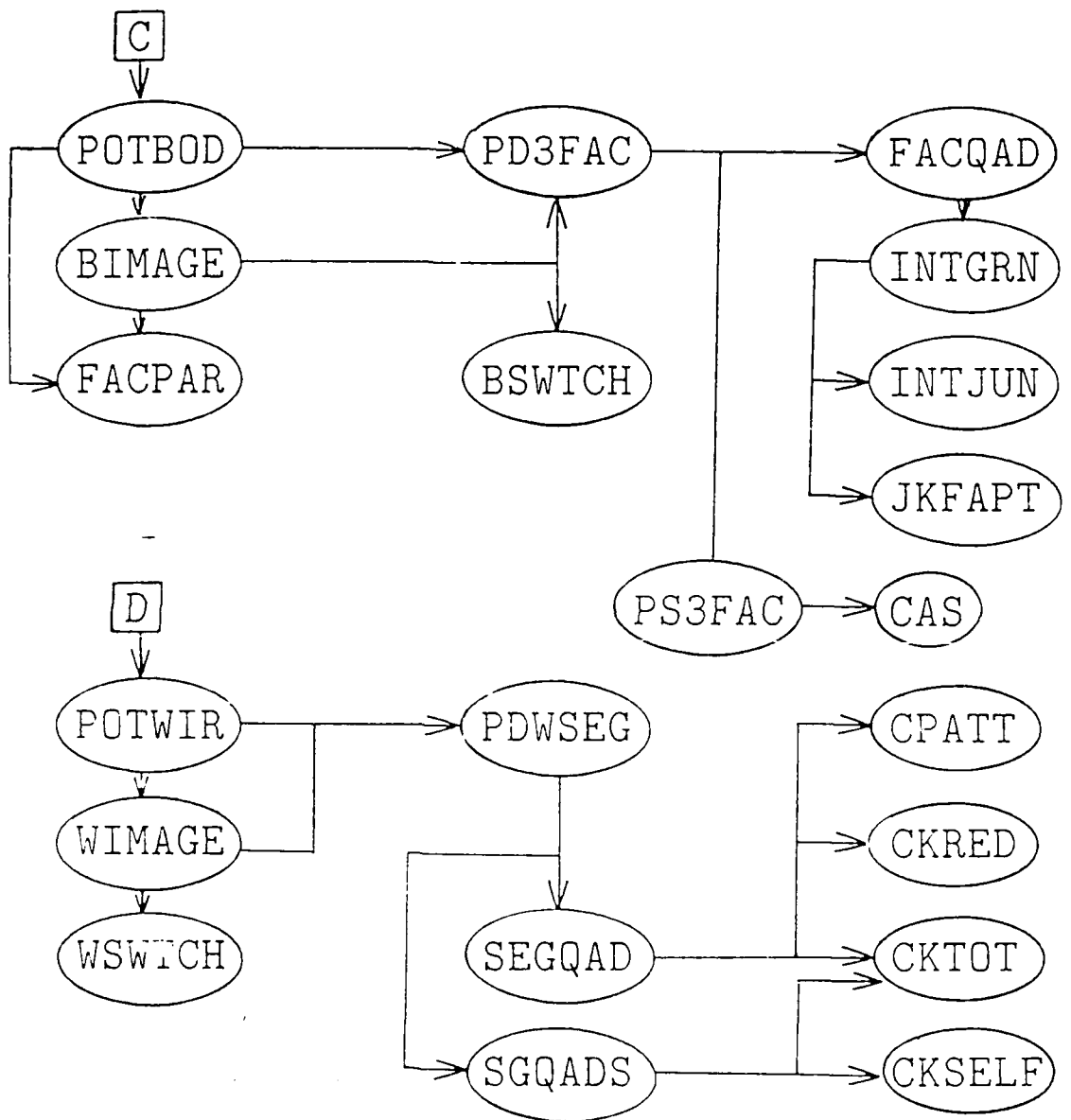


Figure 1.5: Subroutines called by POTBOD and POTWIR.

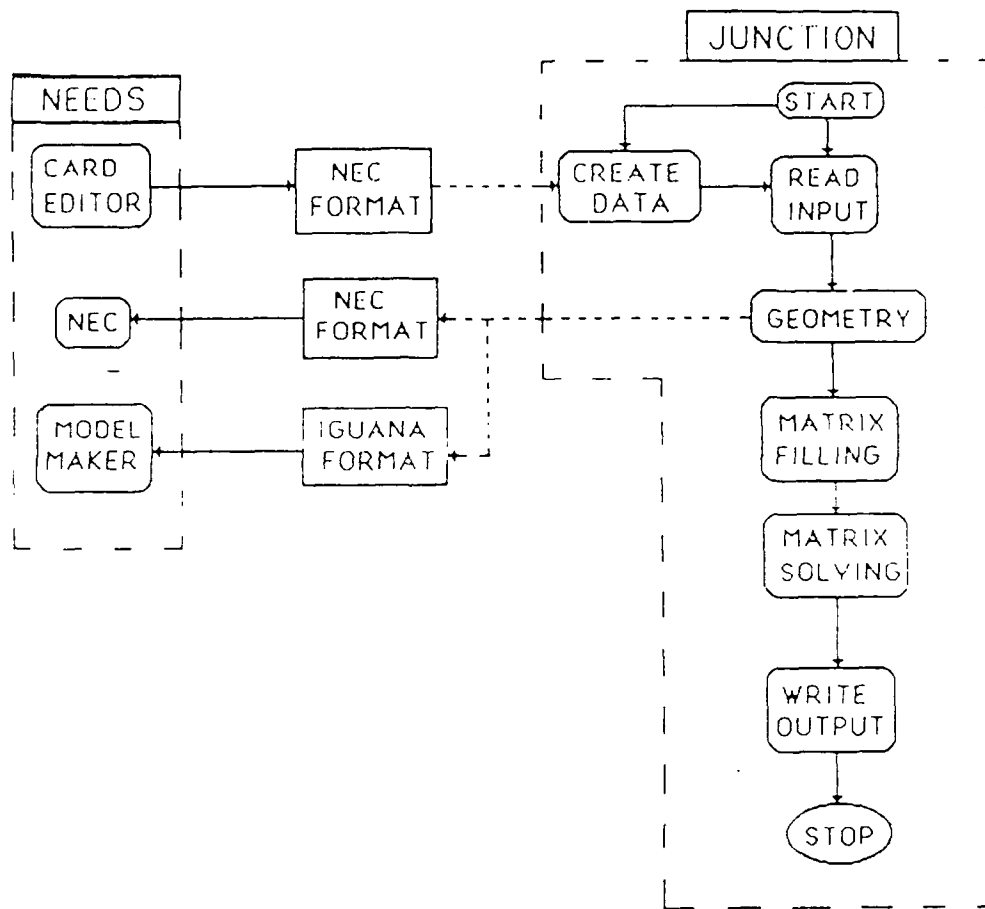


Figure 1.6: Block diagram depicting relationships between JUNCTION and NEEDS.

Chapter 2

Data Generation

In the following sections we discuss several practical aspects related to modeling a structure and to using the JUNCTION code. User aspects include generation of the geometry, and specification of input parameters.

2.1 Triangular Patch and Wire Modeling

The first step in the numerical solution of any practical electromagnetics problem is to accurately model the geometry and to represent it in some form which can be easily handled by the computer. In JUNCTION, we wish to model surfaces and wires by means of discrete triangular patches and segments, respectively. Triangulation and segmentation schemes are non-unique, and finding a suitable one may at first appear to require a certain amount of experience or intuition. However, there are several guidelines which can be followed in order to effectively model a structure. The cardinal rule, however, is simply this—*the density of the triangulation or segmentation should be sufficient to model both the local variations in the surface or wire geometry and in the surface current density.*

2.2 Automatic Triangulation and Segmentation of Models

The generation of the geometry for input to JUNCTION is often greatly simplified by using a geometry preprocessing program called DATGEN. This program is an extension of the program BUILD developed at Sandia laboratories [4] and has been incorporated into JUNCTION. This menu-driven subprogram allows the user to build up a collection of arbitrary bodies by joining together certain canonical three-dimensional triangulated shapes (primitives) to form objects. The

extension contained in JUNCTION also allows the user to add segmented linear wires to form composite wire-and-conducting-body problem geometries. All surface and wire intersections which form junctions are automatically found by JUNCTION by comparing vertex and node coordinates.

DATGEN is an interactive program which is called by JUNCTION and which prompts the user to choose from its catalog of canonical surface shapes or to edit existing geometry files. In creating bodies or wires, the user also provides information to allow DATGEN to automatically subdivide these geometrical elements into triangular patches or segments as required by JUNCTION's input data file. The options available in DATGEN's geometry menu and the required input necessary to define each shape is listed below:

1. **NEC Data**—Input data files are formed by translating a NEC formatted data file which includes GW, SP and SC geometry cards.
2. **Quadrilateral**—A quadrilateral is formed by specifying the coordinates for the four corners in order of progression around the boundary, then specifying the number of edges desired along the side formed by the first two corners, and the number of edges desired along an adjacent side.
3. **Cylinder**—A cylinder is formed by first specifying the center points of the two end plates of the cylinder. Next a point on the circumference of each end plate is defined. These points need not be equidistant from the center points; if they are not, a section of a conical surface is formed. Either end of the cylinder may be open or closed. The number of edges around the circumference of the cylinder, along its axis, and radially along the endcaps (if present) are also specified. A slotted cylinder may also be generated by specifying a beginning and ending angle at each end plate. If the points specified on the end plates are rotated about the axis with respect to one another, then the triangulation scheme and the slot, if present, is similarly twisted about the cylindrical surface.
4. **Cone**—An open-ended, finite-length cone is defined by first specifying the coordinates of the apex. Next, the coordinates of points at the center and on the circumference of the base are given. Finally, the number of edges around the circumference and the number of edges from the apex to the base are specified.
5. **Disk**—A disk is generated by specifying the coordinates of a point at the center, a point on the circumference, and a point on a line perpendicular to and passing through the center of the disk. The triangulation scheme is specified by entering the number of circumferential and radial edges.
6. **Sphere**—The sphere is specified by entering three points: a point at the center, a point at the north pole, and a point on the equator corresponding to zero degrees longitude. It is possible to specify only a sector of a spherical surface. The beginning and ending angles of

the longitudes and latitudes of the sector boundaries are specified, followed by the number of edges along the corresponding directions.

7. **Wire**—A straight wire is formed by specifying the coordinates of the two ends, then specifying the number of subsegments desired and the radius of the wire.

The above geometries are automatically joined by DATGEN along any completely coincident edges by eliminating redundant edges and vertices from the model. The output of DATGEN consists of matrices characterizing the model wire and surface geometries plus a list of so-called "test parameters" which specify the excitation, symmetries present, and quantities to be computed by JUNCTION. The test data are generated from interactive input and are required input for JUNCTION. The geometry data for bodies (wires) consists of two set of matrices. The first matrix is a vertex (node) list matrix in which the row index corresponds to a vertex (node) number and the associated elements are its three-dimensional coordinates with respect to a global origin. The second matrix is an edge (segment) matrix in which the row index corresponds to an edge (a segment) number and the associated elements are the numbers of the two vertices (nodes) to which it is connected. -

2.3 Specification of Input Data

Table 2.1 and 2.2 provide the format of and a brief description of the input data files required by JUNCTION. These files can either be generated by the user, or generated interactively through DATGEN. The following notes apply to Tables 2.1 and 2.2 and more fully explain the function of each parameter in the two input data files.

Notes on Tables 2.1 and 2.2:

1. NJCT specifies the type of wire/body configuration. A value of -1 indicates that a conducting body is present, but no wires; 1 indicates there are bodies, wires and wire-to-surface junctions; 0 indicates there are bodies and wires but no wire-to-surface junctions present in this geometry configuration.
2. The number of ground planes, NGNDP, specifies the number of image planes (electric or magnetic) to be used on symmetrical structures. If this parameter is non-zero, a value must be given for the type of image plane at $x = 0$, $y = 0$, $z = 0$ in the IGNDP parameter line. If NGNDP is zero, then zero can be specified for each of the image planes.
3. IGNDP specifies the type of image plane to be placed in the $x = 0$, $y = 0$, and $z = 0$ planes, respectively. A value of -1 indicates a perfect magnetic conductor, a +1 indicates a perfect electric conductor, and a 0 indicates no image plane present.

Table 2.1: DESCRIPTION OF INPUT DATA IN FOR002.DAT

NJCT MNJUN	Junction flag, number of junctions.
NNODES NEDGES	Number of nodes, edges.
NODE X Y Z	Vertex number and rectangular coordinates.
:	
NE NF NT	Edge number, "from" node and "to" node.
:	
NGNDP	Number of ground planes.
IGNDP(1) IGNDP(2) IGNDP(3)	Ground plane type for $x = 0, y = 0, z = 0$.
IPAT	Pattern flag.
THETA1 THETA2 NTHETA PHI1	Start and end angles for pattern.
PHI2 NPHI	
ICHRGE	Charge density flag.
ITYPE	Type of excitation—plane wave and/or voltage sources.
THETA PHI RTH AITH RPH AIPH	Incident plane wave and polarization information.
NWVLT	Number of voltage sources.
NODE RV XV	Wire node number, voltage (Re, Im)
:	
FREQ	Frequency of operation.
-1	End of the input.

Table 2.2: DESCRIPTION OF INPUT DATA IN FOR008.DAT

NWNOD NWSEG	Number of nodes and wire segments.
NODE X Y Z	Node number and rectangular coordinates.
:	
NSEG NF NT RAD	Segment number, "from" node, "to" node, radius of wire segment.
:	

4. IPAT is a flag which should be set to 1 or 2 if a pattern computation is desired or to 0 if it is not. If IPAT is 0 then the data line following it should be omitted. A value of 1 indicates a 3 point quadrature is used; a 2 indicates a 1 point quadrature is used.
5. The line that begins with THETA1 defines the pattern cuts desired. THETA1 and THETA2 specify the starting and ending angles for the pattern computation, and NTHETA specifies the number of evenly spaced angles between THETA1 and THETA2 for the pattern calculation.
6. ICHARGE is a flag which should be set to 1 if charge density output is desired or to 0 if it is not.
7. ITYPE specifies the type of excitation. P indicates a plane wave excitation, V indicates a voltage source excitation.
8. The line beginning with THETA specifies the incident angle and polarization for plane wave scattering. Refer to Fig. 8. This line should always be present. If a scattering problem is not desired, then these parameters may be set to zero.
9. The line beginning with NODE contains information about the location of voltage sources impressed at wire nodes. NODE denotes the wire node (may be a wire-to-surface junction node) number to which a source is to be added. RV and XV specify the values of real and imaginary parts of the voltage source. This line is repeated for each voltage source. The reference direction for voltage sources is the same as the current reference direction (c.f. Section 2.5).

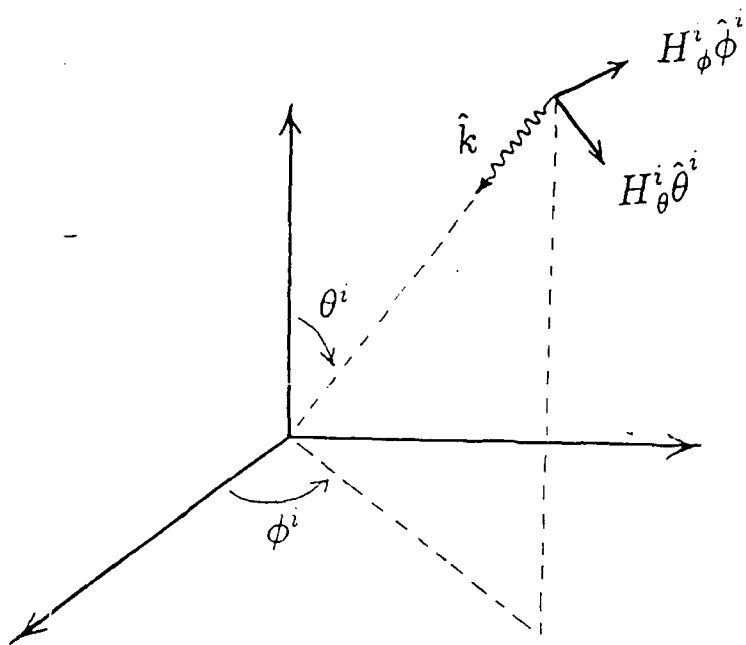


Figure 2.1: Incident field geometrical parameters.

2.4 Modeling Guidelines

1. Triangles may not be imbedded within triangles. Fig. 2.1 shows examples of violations of this rule.
2. For convex bodies, specifying vertices to be on the surface of the original body always results in a triangulated model with reduced surface area and volume as compared to the object being modeled. Thus the solution may appear to correspond to that of an electrically smaller object. This difficulty usually can be substantially corrected by scaling up the object so as to maintain the correct volume or surface area. For this purpose, the surface area and volume of the model are available as output of JUNCTION.
3. At geometrical discontinuities such as edges, corners, wire attachment points, etc., the surface current density usually varies quite rapidly. In such regions, wavelength may have little bearing on the maximum edge length that will yield an accurate solution. Convergence can often be greatly accelerated for such problems by using concentrated mesh schemes in regions where the current varies most rapidly.
4. When modeling closely spaced, almost parallel bodies or wires, the triangulation and segmentation schemes for the parallel surfaces should be made commensurate. Otherwise, modeling-induced discretization of the near field of one surface coupling to the other can produce severe errors in the solution. This problem can also be relieved by choosing edge and segment lengths that are much smaller than the separation distances of the surfaces involved.
5. Segment and triangle edge lengths on wires or in regions on smooth surfaces that are not near surface edges or other geometrical discontinuities, should typically be no longer than 1/5 to 1/10 of a wavelength. This guideline should not, however, be followed blindly. For high-Q resonant structures, such as a resonant length of wire or a narrow tape, for example, it is often found that convergence of the solution as the number of unknowns is increased is very slow and more segments or edges must be used to obtain a solution. Also, very accurate solutions for currents are often required to accurately compute patterns in regions where sidelobe levels are low because the cancellation of the fields in such regions reduces the number of significant figures available. In many practical problems, it is necessary to repeat calculations for several maximum edge or segment lengths in order to check the convergence of the solution.
6. The wire model neglects circumferentially directed currents and any circumferential variation of the axial current; all wire radii must be much smaller than a wavelength at the frequency of excitation for this assumption to remain justified.
7. For wire-to-surface junctions, the wire radii are assumed small compared to the edge lengths of triangles to which they are attached.

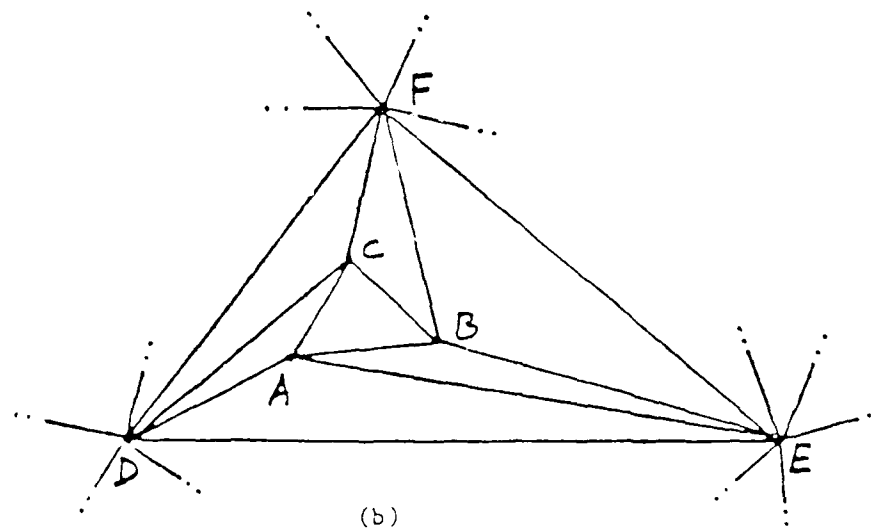
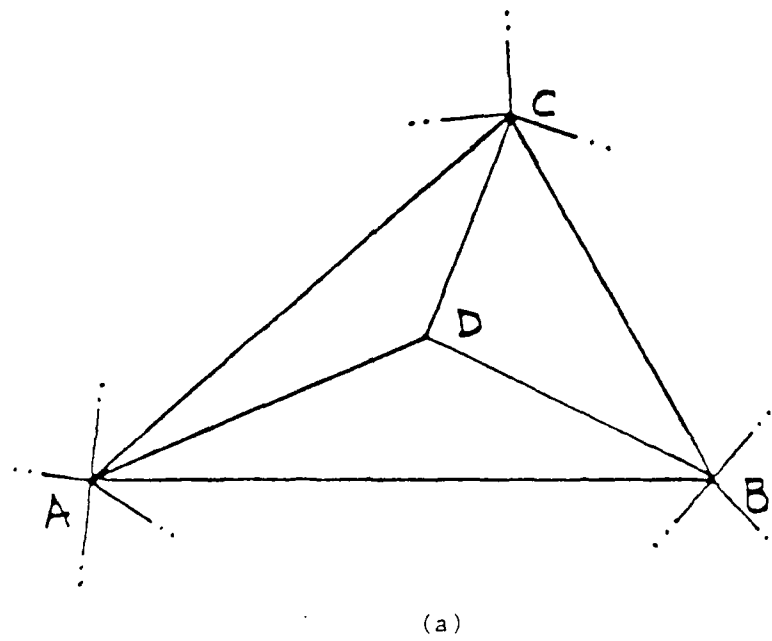


Figure 2.2: Examples of triangles embedded within triangles.

8. If a wire is nearly parallel to a surface at the attachment point, it is assumed that the angle is sufficiently large that the wire-to-surface junction region remains small compared to edge lengths of the attached triangle.
9. For a wire nearly parallel to a surface, the triangular patch density may need to be increased on the surface in the neighborhood of the wire. At wire-to-surface junctions, the junction vertex angles of the junction triangles should be kept small.

2.5 Current Reference Directions

1. The reference current direction on a wire segment is assumed to be from lower numbered to higher numbered segments.
2. The assumed reference direction for the currents across a given edge is determined by the cross product of 1) the edge orientation vector (determined by the "from" and "to" designation in the input data) with 2) the surface normal, taken in that order. The surface normal for a given face is determined from the orientation of the triangle's boundary, as specified by the order in which its edges and/or vertices appear in the output face list. The triangle's normal is merely related to this orientation by the right hand rule. The relationship between face and edge orientations and current reference direction is illustrated in Fig. 2
3. The current reference direction at wire-to-surface junction is into the wire from the surface.
4. For a *closed* surface, the outward normal is automatically chosen by the program. For an *open* surface the following procedure is used to define the normal for one triangular face. Once the normal has been chosen for this face, the program automatically orients the normals for the remaining faces.
 - (a) The lowest numbered edge which is connected to edge number 1 and which also forms a triangular face with edge number 1 is found.
 - (b) The two edges of the previous step are temporarily treated as vectors directed away from their common vertex. Note that for this purpose, the "from" and "to" vertex designations of the edges are ignored.
 - (c) The surface normal is taken to be in the direction of the cross product of these two temporary vectors with edge 1 as the second argument of the cross product. The direction of this normal is then propagated to the adjacent faces and hence over the entire structure.
5. In order to prevent patches from becoming incoherently oriented on intersecting surfaces, it may be necessary to reorient the normals of the intersecting patches manually.

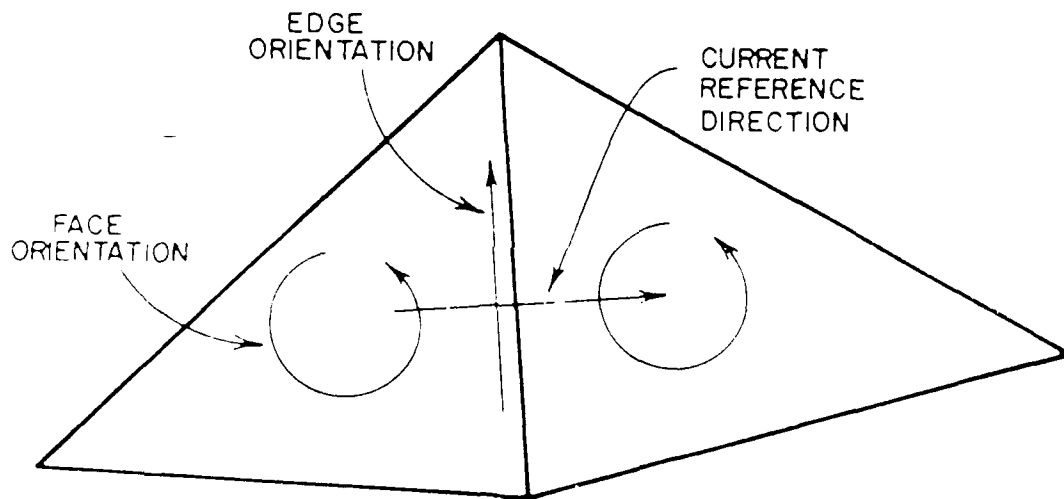


Figure 2.3: The relationship between face and edge orientations and current reference direction.

Chapter 3

Example Problem

The following files were generated by JUNCTION for the simple example problem shown in Fig. 3.1.

1. FOR002.DAT is an input data file associated with the plate.
2. FOR008.DAT is an input data file associated with the wire.
3. NEC.DAT is a NEC formatted geometry data file.
4. IGUANA.NEC is an IGUANA formatted geometry data file.
5. FOR003.DAT is an output data file associated with the plate.
6. FOR009.DAT is an output data file associated with the wire.
7. FOR004.DAT is an output data file for the surface current
8. FOR010.DAT is an output data file for a far field pattern.
9. FOR011.DAT is an output data file for the charge density.

3.1 Geometry

Fig. 3.1 shows the geometry of the example problem. The geometry consists of a 0.15λ length monopole with 0.001λ radius mounted on the center of a $0.2\lambda \times 0.2\lambda$ square plate. The wire was divided into 3 segments and the plate was divided into 8 triangular patches. This is not a sufficient number of wire segments or triangles to accurately model the geometry, but it is sufficient to illustrate the format of the input and output of the program.

Wire Radius: $a = 0.001\lambda$
 Wire Length: 0.15λ

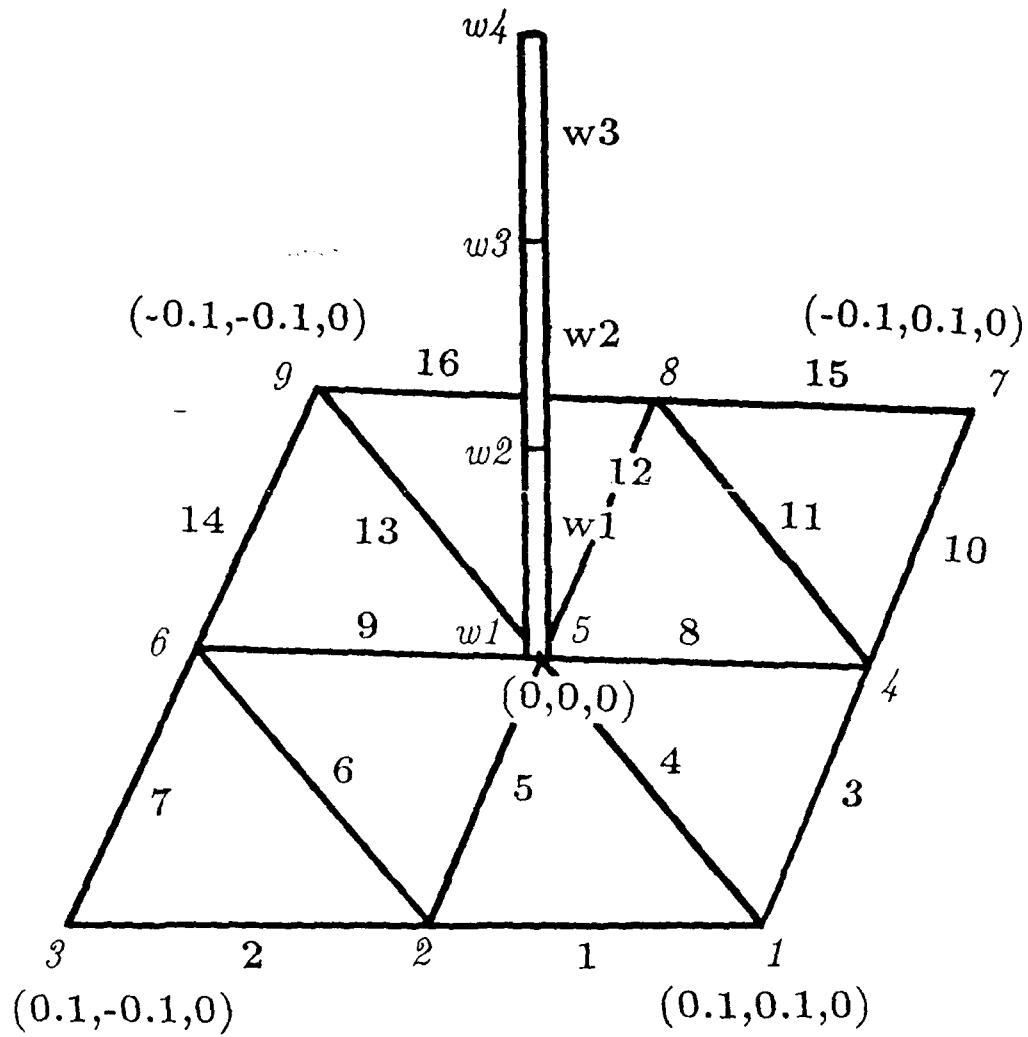


Figure 3.1: Geometry of example problem.

3.2 Input Data

3.2.1 FOR002.DAT

```
C
C FOR002.DAT:INPUT DATA ASSOCIATED WITH BODIES
C
1 1
9 16
1 1.00000001490E-01 1.00000001490E-01 0.00000000000E+00
2 1.00000001490E-01 0.00000000000E+00 0.00000000000E+00
3 1.00000001490E-01 -1.00000001490E-01 0.00000000000E+00
4 0.00000000000E+00 1.00000001490E-01 0.00000000000E+00
5 0.00000000000E+00 0.00000000000E+00 0.00000000000E+00
6 0.00000000000E+00 -1.00000001490E-01 0.00000000000E+00
7 -1.00000001490E-01 1.00000001490E-01 0.00000000000E+00
8 -1.00000001490E-01 0.00000000000E+00 0.00000000000E+00
9 -1.00000001490E-01 -1.00000001490E-01 0.00000000000E+00
  1 1 2
  2 2 3
  3 1 4
  4 1 5
  5 2 5
  6 2 6
  7 3 6
  8 4 5
  9 5 6
 10 4 7
 11 4 8
 12 5 8
 13 5 9
 14 6 9
 15 7 8
 16 8 9
      0
      0 0 0
      2
0.0000000E+00 0.0000000E+00 1 0.0000000E+00 180.0000
19
1
```

V

```
      1
      1 1.000000      0.000000E+00
3.000000E+08
-1.000000
```

3.2.2 FOR008.DAT

C

C FOR008.DAT : INPUT DATA ASSOCIATED WITH WIRE

C

```
      4          3
      1 0.000000E+00 0.000000E+00 0.000000E+00
      2 0.000000E+00 0.000000E+00 5.000000E-02
      3 0.000000E+00 0.000000E+00 0.100000
      4 0.000000E+00 0.000000E+00 0.150000
      1          1          2 1.000000E-03
      2          2          3 1.000000E-03
      3          3          4 1.000000E-03
```

V

```
      1
      1 1.000000      0.000000E+00
```


3.3 Output Data

3.3.1 NEC.DAT

C

C NEC.DAT : GEOMETRY DATA IN NEC FORMAT

C

CM INPUT DATA IN NEC FORMAT

CE

GW	0	3	0.0000	0.0000	0.0000	0.0000	0.0000	0.1500	0.0010
SP		2	0.0000	0.0000	0.0000	0.1000	0.0000	0.0000	
SC		2	0.1000	0.1000	0.0000				
SP		2	0.0000	-0.1000	0.0000	0.1000	-0.1000	0.0000	
SC		2	0.1000	0.0000	0.0000				
SP		2	0.0000	0.1000	0.0000	0.0000	0.0000	0.0000	
SC		2	0.1000	0.1000	0.0000				
SP		2	0.0000	0.0000	0.0000	0.0000	-0.1000	0.0000	
SC		2	0.1000	0.0000	0.0000				
SP		2	-0.1000	0.0000	0.0000	0.0000	0.0000	0.0000	
SC		2	0.0000	0.1000	0.0000				
SP		2	-0.1000	-0.1000	0.0000	0.0000	-0.1000	0.0000	
SC		2	0.0000	0.0000	0.0000				
SP		2	-0.1000	0.1000	0.0000	-0.1000	0.0000	0.0000	
SC		2	0.0000	0.1000	0.0000				
SP		2	-0.1000	0.0000	0.0000	-0.1000	-0.1000	0.0000	
SC		2	0.0000	0.0000	0.0000				
GE									
EN									

3.3.2 IGU.NEC

C

C IGU.NEC : GEOMETRY DATA IN IGUANA FORMAT

C

CM, INPUT DATA IN IGUANA FORMAT

CE,

GW, 1,1,	0.100,	0.100,	0.000,	0.100,	0.000,	0.000,0.1
GW, 2,1,	0.100,	0.000,	0.000,	0.100,	-0.100,	0.000,0.1
GW, 3,1,	0.100,	0.100,	0.000,	0.000,	0.100,	0.000,0.1
GW, 4,1,	0.100,	0.100,	0.000,	0.000,	0.000,	0.000,0.1
GW, 5,1,	0.100,	0.000,	0.000,	0.000,	0.000,	0.000,0.1
GW, 6,1,	0.100,	0.000,	0.000,	0.000,	-0.100,	0.000,0.1
GW, 7,1,	0.100,	-0.100,	0.000,	0.000,	-0.100,	0.000,0.1
GW, 8,1,	0.000,	0.100,	0.000,	0.000,	0.000,	0.000,0.1
GW, 9,1,	0.000,	0.000,	0.000,	0.000,	-0.100,	0.000,0.1
GW, 10,1,	-0.000,	0.100,	0.000,	-0.100,	0.100,	0.000,0.1
GW, 11,1,	0.000,	0.100,	0.000,	-0.100,	0.000,	0.000,0.1
GW, 12,1,	0.000,	0.000,	0.000,	-0.100,	0.000,	0.000,0.1
GW, 13,1,	0.000,	0.000,	0.000,	-0.100,	-0.100,	0.000,0.1
GW, 14,1,	0.000,	-0.100,	0.000,	-0.100,	-0.100,	0.000,0.1
GW, 15,1,	-0.100,	0.100,	0.000,	-0.100,	0.000,	0.000,0.1
GW, 16,1,	-0.100,	0.000,	0.000,	-0.100,	-0.100,	0.000,0.1
GW,999, 3,	0.00,	0.00,	0.00,	0.00,	0.00,	0.15, 0.0010

GE,

EN,

3.3.3 FOR003.DAT

C
C FOR003.DAT: OUTPUT DATA ASSOCIATED WITH BODIES
C

NUMBER OF JUNCTION= 1
ON WIRE NODE 1

NUMBER OF IMAGE PLANES= 0
IMAGE PLANE NOTATION:
0=NO GROUND PLANE
1=A P.M.C. GROUND PLANE
-1=A P.E.C. GROUND PLANE
0 IN THE X=0 PLANE
0 IN THE Y=0 PLANE
0 IN THE Z=0 PLANE

NUMBER OF VOLTAGE SOURCE = 1
V=(1.0,0.0) VOLTS ON WIRE NODE 1

VERTEX COORDINATE LIST
ALL DIMENSIONS ARE IN METERS

VERTEX NUMBER	X-COORDINATE	Y-COORDINATE	Z-COORDINATE
1	0.10000E+00	0.10000E+00	0.00000E+00
2	0.10000E+00	0.00000E+00	0.00000E+00
3	0.10000E+00	-0.10000E+00	0.00000E+00
4	0.00000E+00	0.10000E+00	0.00000E+00
5	0.00000E+00	0.00000E+00	0.00000E+00
6	0.00000E+00	-0.10000E+00	0.00000E+00
7	-0.10000E+00	0.10000E+00	0.00000E+00
8	-0.10000E+00	0.00000E+00	0.00000E+00
9	-0.10000E+00	-0.10000E+00	0.00000E+00

IPAT= 2
IF IPAT.GT.0 FAR FIELD PATTERNS ARE COMPUTED
PATTERN PARAMETERS:
PHI1, PHI2, NPHI, THETA1, THETA2, NTHETA

0.0000 0.0000 1 0.0000 180.0000 19

FOR BODY NUMBER: 1

FACE	1	HAS EDGES	1	4	5	WITH VERTICES	5	2	1
FACE	2	HAS EDGES	2	6	7	WITH VERTICES	6	3	2
FACE	3	HAS EDGES	4	3	8	WITH VERTICES	4	5	1
FACE	4	HAS EDGES	6	5	9	WITH VERTICES	5	6	2
FACE	5	HAS EDGES	8	11	12	WITH VERTICES	8	5	4
FACE	6	HAS EDGES	9	13	14	WITH VERTICES	9	6	5
FACE	7	HAS EDGES	11	10	15	WITH VERTICES	7	8	4
FACE	8	HAS EDGES	13	12	16	WITH VERTICES	8	9	5

EDGE-VERTEX CONNECTION LIST

EDGE	1	GOES FROM VERTEX	1	TO VERTEX	2	MULT=	0
EDGE	2	GOES FROM VERTEX	2	TO VERTEX	3	MULT=	0
EDGE	3	GOES FROM VERTEX	1	TO VERTEX	4	MULT=	0
EDGE	4	GOES FROM VERTEX	1	TO VERTEX	5	MULT=	1
EDGE	5	GOES FROM VERTEX	2	TO VERTEX	5	MULT=	1
EDGE	6	GOES FROM VERTEX	2	TO VERTEX	6	MULT=	1
EDGE	7	GOES FROM VERTEX	3	TO VERTEX	6	MULT=	0
EDGE	8	GOES FROM VERTEX	4	TO VERTEX	5	MULT=	1
EDGE	9	GOES FROM VERTEX	5	TO VERTEX	6	MULT=	1
EDGE	10	GOES FROM VERTEX	4	TO VERTEX	7	MULT=	0
EDGE	11	GOES FROM VERTEX	4	TO VERTEX	8	MULT=	1
EDGE	12	GOES FROM VERTEX	5	TO VERTEX	8	MULT=	1
EDGE	13	GOES FROM VERTEX	5	TO VERTEX	9	MULT=	1
EDGE	14	GOES FROM VERTEX	6	TO VERTEX	9	MULT=	0
EDGE	15	GOES FROM VERTEX	7	TO VERTEX	8	MULT=	0
EDGE	16	GOES FROM VERTEX	8	TO VERTEX	9	MULT=	0

BODY PARAMETER LIST

NUMBER OF VERTICES= 9
NUMBER OF EDGES= 16
NUMBER OF FACES= 8

NUMBER OF EDGES INCLUDING MULTIPLICITY= 8

MODELING PARAMETER LIST (METERS)

SURFACE AREA OF THE SCATTERER= 0.40000E-01 SQ.METERS
AVERAGE EDGE LENGTH= 0.12071E+00 METERS
MAXIMUM EDGE LENGTH(EDGE NO. 4)= 0.14142E+00 METERS
MINIMUM EDGE LENGTH(EDGE NO. 1)= 0.10000E+00 METERS
AVERAGE FACE AREA = 0.50000E-02 SQ.METERS
MAXIMUM FACE AREA (FACE NO. 1)= 0.50000E-02 SQ.METERS
MINIMUM FACE AREA (FACE NO. 1)= 0.50000E-02 SQ.METERS
MINIMUM FACE HEIGHT TO BASE RATIO (FACE NO. 1)=0.50000E+00

EDGE 1 IS ATTACHED TO FACES
1
EDGE 2 IS ATTACHED TO FACES
2
EDGE 3 IS ATTACHED TO FACES
3
EDGE 4 IS ATTACHED TO FACES
1 3
EDGE 5 IS ATTACHED TO FACES
1 4
EDGE 6 IS ATTACHED TO FACES
2 4
EDGE 7 IS ATTACHED TO FACES
2
EDGE 8 IS ATTACHED TO FACES
3 5
EDGE 9 IS ATTACHED TO FACES
4 6
EDGE 10 IS ATTACHED TO FACES
7
EDGE 11 IS ATTACHED TO FACES
5 7
EDGE 12 IS ATTACHED TO FACES
5 8
EDGE 13 IS ATTACHED TO FACES
6 8

EDGE 14 IS ATTACHED TO FACES

6

EDGE 15 IS ATTACHED TO FACES

7

EDGE 16 IS ATTACHED TO FACES

8

FREQ= 3.00000E+08

SURFACE CURRENTS

EDGE NUMBER	CURRENT DENSITY (AMPS/METER)			PHASE(DEG)
	REAL	IMAGINARY	MAGNITUDE	
1	0.00000E+00	0.00000E+00	0.00000E+00	
2	0.00000E+00	0.00000E+00	0.00000E+00	
3	0.00000E+00	0.00000E+00	0.00000E+00	
4	-0.13488E-09	-0.64974E-08	0.64988E-08	90.000
5	0.60283E-05	0.51320E-03	0.51324E-03	89.327
6	0.13376E-03	0.42825E-02	0.42846E-02	88.211
7	0.00000E+00	0.00000E+00	0.00000E+00	
8	-0.60286E-05	-0.51321E-03	0.51325E-03	-90.673
9	0.60279E-05	0.51321E-03	0.51324E-03	89.327
10	0.00000E+00	0.00000E+00	0.00000E+00	
11	-0.13376E-03	-0.42825E-02	0.42846E-02	-91.789
12	-0.60277E-05	-0.51320E-03	0.51323E-03	-90.673
13	0.24486E-09	0.83496E-08	0.83532E-08	90.000
14	0.00000E+00	0.00000E+00	0.00000E+00	
15	0.00000E+00	0.00000E+00	0.00000E+00	
16	0.00000E+00	0.00000E+00	0.00000E+00	

3.3.4 FOR009.DAT

C

C FOR009.DAT : OUTPUT DATA ASSOCIATED WITH WIRE

C

NODES	SEGMENTS		
4	3		

NODE #	X	Y	Z
1	0.0000000E+00	0.0000000E+00	0.0000000E+00
2	0.0000000E+00	0.0000000E+00	5.0000001E-02
3	0.0000000E+00	0.0000000E+00	0.1000000
4	0.0000000E+00	0.0000000E+00	0.1500000

SEG. #	FROM	TO	RADIUS
1	1	2	1.0000000E-03
2	2	3	1.0000000E-03
3	3	4	1.0000000E-03

SEG. #	CENTER POINT COORDINATES		
1	0.0000000E+00	0.0000000E+00	2.5000000E-02
2	0.0000000E+00	0.0000000E+00	7.5000003E-02
3	0.0000000E+00	0.0000000E+00	0.1250000

TOTAL UNKNOWN NUMBER = 3

NODE #	MULTIPLICITY
1	1
2	1
3	1
4	0

SURFACE CURRENTS

EDGE NUMBER	CURRENT DENSITY (AMPS/METER)			PHASE(DEG)
	REAL	IMAGINARY	MAGNITUDE	
1	0.12649E-03	0.50176E-02	0.50192E-02	88.556
2	0.11298E-03	0.35006E-02	0.35024E-02	88.152
3	0.72494E-04	0.18988E-02	0.19002E-02	87.814
4	0.00000E+00	0.00000E+00		

3.3.5 FOR004.DAT

C

C FOR004.DAT : SURFACE CURRENT LISTING

C

FREQ= 3.00000E+08

SURFACE CURRENTS

EDGE NUMBER	CURRENT DENSITY (AMPS/METER)			PHASE(DEG)
	REAL	IMAGINARY	MAGNITUDE	
1	0.00000E+00	0.00000E+00	0.00000E+00	
2	0.00000E+00	0.00000E+00	0.00000E+00	
3	0.00000E+00	0.00000E+00	0.00000E+00	
4	-0.13488E-09	-0.64974E-08	0.64988E-08	90.000
5	0.60283E-05	0.51320E-03	0.51324E-03	89.327
6	0.13376E-03	0.42825E-02	0.42846E-02	88.211
7	0.00000E+00	0.00000E+00	0.00000E+00	
8	-0.60286E-05	-0.51321E-03	0.51325E-03	-90.673
9	0.60279E-05	0.51321E-03	0.51324E-03	89.327
10	0.00000E+00	0.00000E+00	0.00000E+00	
11	-0.13376E-03	-0.42825E-02	0.42846E-02	-91.789
12	-0.60277E-05	-0.51320E-03	0.51323E-03	-90.673
13	0.24486E-09	0.83496E-08	0.83532E-08	90.000
14	0.00000E+00	0.00000E+00	0.00000E+00	
15	0.00000E+00	0.00000E+00	0.00000E+00	
16	0.00000E+00	0.00000E+00	0.00000E+00	
1	0.12649E-03	0.50176E-02	0.50192E-02	88.556
2	0.11298E-03	0.35006E-02	0.35024E-02	88.152
3	0.72494E-04	0.18988E-02	0.19002E-02	87.814
4	0.00000E+00	0.00000E+00		

3.3.6 FOR010.DAT

C

C FOR010.DAT : FAR FIELD PATTERN LISTING

C

FAR FIELD PATTERN

ITHETA	THETA	IPHI	PHI	ETH(ITHETA,IPHI)		EPH(ITHETA,IPHI)	
1	0.00	1	0.00	-4.66E-10	-9.81E-10	8.03E-09	-1.34E-09
2	10.00	1	0.00	-1.20E-02	-5.22E-03	4.46E-06	1.15E-04
3	20.00	1	0.00	-2.38E-02	-9.80E-03	8.77E-06	2.26E-04
4	30.00	1	0.00	-3.52E-02	-1.32E-02	1.28E-05	3.30E-04
5	40.00	1	0.00	-4.58E-02	-1.49E-02	1.64E-05	4.23E-04
6	50.00	1	0.00	-5.54E-02	-1.47E-02	1.95E-05	5.03E-04
7	60.00	1	0.00	-6.35E-02	-1.26E-02	2.19E-05	5.68E-04
8	70.00	1	0.00	-6.96E-02	-8.67E-03	2.38E-05	6.15E-04
9	80.00	1	0.00	-7.34E-02	-3.49E-03	2.49E-05	6.43E-04
10	90.00	1	0.00	-7.45E-02	2.34E-03	2.52E-05	6.53E-04
11	100.00	1	0.00	-7.30E-02	8.09E-03	2.49E-05	6.43E-04
12	110.00	1	0.00	-6.89E-02	1.30E-02	2.38E-05	6.15E-04
13	120.00	1	0.00	-6.26E-02	1.65E-02	2.19E-05	5.68E-04
14	130.00	1	0.00	-5.44E-02	1.81E-02	1.95E-05	5.03E-04
15	140.00	1	0.00	-4.48E-02	1.77E-02	1.64E-05	4.23E-04
16	150.00	1	0.00	-3.43E-02	1.53E-02	1.28E-05	3.30E-04
17	160.00	1	0.00	-2.31E-02	1.13E-02	8.77E-06	2.26E-04
18	170.00	1	0.00	-1.16E-02	5.95E-03	4.46E-06	1.15E-04
19	180.00	1	0.00	4.66E-10	9.81E-10	8.03E-09	-1.34E-09

3.3.7 FOR011.DAT

C

C FOR011.DAT : CHARGE DENSITY LISTING

C

SURFACE CHARGE

FACE NUMBER	CHARGE DENSITY (COULOMBS/SQ.METER)			
	REAL	IMAGINARY	MAGNITUDE	PHASE
1	-0.6110300E-10	0.1613704E-11	0.6112431E-10	0.1784872E+03
2	-0.6426055E-10	0.2007158E-11	0.6429188E-10	0.1782110E+03
3	-0.6110310E-10	0.1613705E-11	0.6112440E-10	0.1784872E+03
4	-0.7972668E-10	0.1476099E-11	0.7974035E-10	0.1789393E+03
5	-0.7972668E-10	0.1476101E-11	0.7974035E-10	0.1789393E+03
6	-0.6110314E-10	0.1613713E-11	0.6112445E-10	0.1784872E+03
7	-0.6426052E-10	0.2007157E-11	0.6429186E-10	0.1782110E+03
8	-0.6110301E-10	0.1613709E-11	0.6112431E-10	0.1784872E+03

TOTAL CHARGE ON THE BODY= (-0.2661934E-11 0.6710674E-13) COULOMBS

CHARGE DENSITY ON WIRE

SEGMENT NUMBER	CHARGE DENSITY (COULOMBS/METER)			
	REAL	IMAGINARY	MAGNITUDE	PHASE
1	0.1609630E-10	-0.1434157E-12	0.1609694E-10	-0.5104834E+00
2	0.1699554E-10	-0.4295376E-12	0.1700096E-10	-0.1447760E+01
3	0.2014683E-10	-0.7691813E-12	0.2016151E-10	-0.2186421E+01

TOTAL CHARGE ON THE WIRE= (0.2661934E-11 -0.6710673E-13) COULOMBS

Bibliography

- [1] D. R. Wilton and S. U. Hwu, "Electromagnetic Scattering and Radiation by Arbitrary Configurations of Conducting Bodies and Wires," Technical Report No. 87-17, Applied Electromagnetics Lab., Dept. of Electrical Engr., Univ. of Houston, 1987.
- [2] J. Strauch, "User's Guide for the Interactive Graphics Utility for Automated NEC Analysis (IGUANA) for version 4.1", Unisys Corporation, San Diego, CA, July 1987.
- [3] G. J. Burke and A. J. Poggio, "Numerical Electromagnetic Code (NEC) - Method of Moments," Lawrence Livermore Laboratory, Jan. 1981.
- [4] W. A. Johnson, D. R. Wilton and R. M. Sharpe, "PATCH Code Users Manual." Sandia National Laboratories, Albuquerque, N.M. 87185, 1987.
- [5] J. Dongarra, J. R. Bunch, C. B. Moler, and G. W. Stewart. *LINPACK Users Guide*. SIAM Publications, 1978.

Appendix A

Program Listings

Following is a FORTRAN listing of the computer program *JUNCTION* and its supporting subprograms. Only the called LINPACK [5] subroutines CGEFA and CGESL, which solve linear systems of simultaneous equations are not included in the listing. If these routines are not readily available, any equivalent equation-solving subroutines can be substituted.

A.1 JUNCTION

```
C          PROGRAM JUNCTION V1.1
C-----
C
C ELECTROMAGNETIC SCATTERING AND RADIATION BY WIRES ATTACHED TO
C CONDUCTING SURFACES OF ARBITRARY SHAPE
C
C          SHIAN-UEI HWU
C        APPLIED ELECTROMAGNETICS LAB
C        UNIVERSITY OF HOUSTON
C
C          HOUSTON, TX77004
C
C THIS PROGRAM WAS DEVELOPED AS AN EXTENSION OF THE ELECTRIC FIELD
C INTEGRAL EQUATION SURFACE PATCH CODE AND THIN WIRE CODE
C
C ORIGINAL PATCH VERSION BY S. M. RAO
C ORIGINAL WIRE VERSION BY S. SINGH
C
C REORGANIZATION AND EXTENSIONS BY W. A. JOHNSON OF
C SANDIA NATIONAL LAB 1984
C
C INPUT DATA IN FOR002.DAT(BODIES) AND FOR008.DAT(WIRES)
C OUTPUT DATA IN FOR003.DAT, FOR004.DAT(BODIES), FOR009.DAT(WIRES)
C MODELING CAPABILITIES INCLUDE SYMMETRY PLANES AND/OR GROUND PLANES
C-----
C I. MAIN :
C   1. DATGEN : TO CREATE INPUT DATA OF BODY AND WIRE
C     1.1. WIRDAT : TO CREATE A STRAIGHT WIRE
C     1.2. NECDAT : TO READ INPUT DATA IN NEC FORMATE
C   2. BODIN : TO READ INPUT DATA ASSOCIATE WITH BODIES
C   3. WIRIN : TO READ INPUT DATA ASSOCIATE WITH WIRES
C     3.1. FNDJUN : TO FIND JUNCTION NODES IF THERE ARE ANY
C     3.2. WIRMUL : TO FIND MULTIPLICITY OF WIRE SEGMENTS
C   4. ADWMUL : TO ADJUST MULTIPLICITY OF EACH WIRE NODE IF
C     THERE ARE ANY GROUND PLANE ATTACHED
C   5. FACMUL : TO FIND FACES DATALIST OF BODIES
C   6. ORTFAC : TO ORIENT THE FACE OF BODIES
C   7. CLSBOD : TO FIND NORMAL VECTOR OF CLOSED BODY
C     7.1. FACEDG : TO FIND EDGES ASSOCIATED WITH FACE
C     7.2. FACVTX : TO FIND VERTICES ASSOCIATED WITH FACE
C     7.3. VTXCRD : TO FIND COORDINATES OF THE VERTICES
C   8. FACOUT : TO PRINT THE EDGES AND THE VERTICES OF EACH FACE
```

C 8.2. FACVTX: TO FIND VERTICES ASSOCIATED WITH FACE
 C 9. ADBMUL : TO ADJUST MULTIPLICITY OF EACH EDGE IF
 C THERE ARE ANY GROUND PLANE ATTACHED
 C 10. BODPAR : TO CALCULATE PARAMETERS ASSOCIATED WITH BODIES
 C 10.2. FACVTX: TO FIND VERTICES ASSOCIATED WITH FACE
 C 10.3. VTXCRD: TO FIND COORDINATES OF THE VERTICES
 C 11. EDGFAC : TO FIND FACES ATTACHED TO EACH EDGE
 C 12. SOLTN : TO SOLVE THE MATRIX EQUATION
 C 12.1. JUNFAC : TO FIND FACES ATTACHED TO EACH JUNCTION
 C 12.2. JANGLE : TO FIND VERTEX ANGLE OF EACH JUNCTION FACE
 C 12.3. BCUMUL : TO ACCUMULATE THE MULTIPLICITIES UP TO
 C EACH EDGE
 C 12.4. MTXWIR : TO FILL IMPEDANCE MATRIX
 C FOR SOURCE ON THE WIRES
 C 12.5. ZBB :TO FILL IMPEDANCE MATRIX
 C FOR SOURCE AND OBSERVATION ON THE BODIES
 C 12.6. ZWB :TO FILL IMPEDANCE MATRIX
 C FOR SOURCE ON THE BODIES AND OBSERVATION
 C ON THE WIRES
 C 12.7. CGEFA : TO INVERSE THE IMPEDANCE MATRIX
 C 12.8. CGESL : TO SOLVE THE UNKNOWN MATRIX
 C

 * MXUNKN = MAXIMUM NUMBER OF UNKNOWNNS EXPECTED. *

C BODIES:

C MXEDGS=MAXIMUM NUMBER OF EDGES EXPECTED
 C MXBDND=MAXIMUM NUMBER OF NODES EXPECTED
 C MXFACE=MAXIMUM NUMBER OF FACES EXPECTED
 C MXDJBDD=MAXIMUM NUMBER OF DISJOINT BODIES EXPECTED
 C MXMULT=MAXIMUM MULTIPLICITY OF ANY EDGE OVER ALL EDGES
 C MXEXCI=MAXIMUM NUMBER OF EXCITATIONS.
 C MXFREQ=MAXIMUM NUMBER OF FREQUENCY CASES TO BE RUN.
 C MNJFACE:MAXIMUM NUMBER OF FACE ATTACHING TO JUNCTION POINT
 C

* WIRES:

* MXWNOD = MAXIMUM NUMBER OF WIRE NODES. *
 * MXWMLT = MAXIMUM MULTIPLICITY THAT ANY WIRE NODE MAY HAVE. *
 * MXWSEG = MAXIMUM NUMBER OF WIRE SEGMENTS. *
 * MXWVLT = MAXIMUM NUMBER OF DELTA GAP VOLTAGE SOURCES ON THE WIRES. *
 C=====

PROGRAM JUNCTION
 IMPLICIT COMPLEX(C)

C
 C 732 IS THE MAXIMUM DIMENSION CAN BE RUN ON GEORGE (VAX8530 IN UH)

```

C     PARAMETER(MXUNKN= 732,MXBDND= 500,MXEDGS= 732,MXFACE= 600,
C
C FOR BODIES
C
C     PARAMETER(MXUNKN= 600,MXBDND= 250,MXEDGS= 600,MXFACE= 500,
$     MXDJBD=1,MXMULT= 2,MXEXCI=1,MXFREQ=1,MNJFACE= 34)
C
C FOR WIRES
C
C     PARAMETER(MXWNOD= 41,MXWMLT=1,MXWSEG= 41,MXWVLT=2)
C     COMPLEX CZ(MXUNKN,MXUNKN),CV(MXUNKN),CWORK(MXUNKN),CWVLT(MXWVLT)
C     INTEGER NCONN(3,MXEDGS),IWORK(MXEDGS),NBOUND(3,MXFACE),
$     ISTART(MXDJBD+1),NBE(MXDJBD),IPVT(MXUNKN),
$     IEDGF(MXMULT+1,MXEDGS),IGNDP(3),NBUN(MXWNOD),
$     NWJUN(MXWNOD),MULTW(MXWNOD),NSEGC(2,MXWSEG),
$     INSEG(MXWMLT+1,MXWNOD),NODVLT(MXWVLT)
C     REAL DATNOD(3,MXBDND),EXCITE(7,MXEXCI),
$     WNODE(3,MXWNOD),WSEGH(3,MXWSEG),RAD(MXWSEG)
C
C FOR WIRE
C
C     REAL ANG(MXWNOD,MNJFACE)
C     INTEGER NJFACE(MXWNOD,MNJFACE),MIFACE(MXWNOD),WIRSUM(MXWNOD)
C     CHARACTER*1 ID,IG,IC,IS
C     CHARACTER*15 NECNAM,IGUNAM
C
C     COMMON/TEST/RATIO1,RATIO2,RATIO3
C     COMMON/MCHVAL/VALMAX,VALMIN
C     COMMON/IGUANA/IG
C     COMMON/CPU/TGG,IC
C     MULTI=MXMULT+1
C     VALMAX= 1E35
C     VALMIN=-1E35
C
C SET TESTING BOUNDARY TO USE 1,3, OR 7 SAMPLING POINTS
C IN NUMERICAL INTEGRATION
C
C     RATIO1=1.
C     RATIO2=9.
C     RATIO3=100.
C INPUT DATA IN FOR002.DAT(BODIES) AND FOR008.DAT(WIRES)
C OUTPUT DATA IN FOR003.DAT,FOR004.DAT(BODIES), FOR009.DAT(WIRES)
C     WRITE(6,*)'-----'
C     WRITE(6,*)'          MOMENT METHOD SOLUTION          '
C     WRITE(6,*)'WIRES ATTACHED TO CONDUCTING SURFACES '

```



```

WRITE(6,*)'-----'
WRITE(6,*)' '
C
C CALL DATGEN TO GENERATE INPUT DATA FOR BODY AND WIRE
C
WRITE(6,*)'DO YOU WANT TO GENERATE INPUT DATA? (YES OR NO)'
55 CALL EOFCLR(5)
READ(5,5,END=55)ID
5 FORMAT(A)
IF(ID.EQ.'Y')THEN
CALL DATGEN
WRITE(6,*)'*** INPUT DATA CREATED ***'
WRITE(6,*)' '
ELSE
WRITE(6,*)'IF INPUT DATA EXIST SHOULD BE AS FOR002.DAT(BODIES)
$AND FOR008.DAT(WIRES)'
WRITE(6,*)' '
ENDIF
WRITE(6,*)'DO YOU WANT TO TRANSFER INPUT DATA TO NEC AND IGUANA
$ FORMAT? (YES OR NO)'
56 CALL EOFCLR(5)
READ(5,5,END=56)IG
IF(IG.EQ.'Y')THEN
WRITE(6,*)'GIVE A FILENAME TO NEC FORMAT DATA FILE'
57 CALL EOFCLR(5)
READ(5,5,END=57)NECNAM
C
WRITE(6,*)'GIVE A FILENAME TO IGUANA FORMAT DATA FILE (name.NEC)'
58 CALL EOFCLR(5)
READ(5,5,END=58)IGUNAM
ENDIF
C
WRITE(6,*)'DO YOU WANT TO STOP TO CHECK GEOMETRY DATA ?
$(YES OR NO)'
59 CALL EOFCLR(5)
READ(5,5,END=59)IS
C
IF(IS.NE.'Y')THEN
WRITE(6,*)'DO YOU WANT TO SHOW CPUTIME? (YES OR NO)'
61 CALL EOFCLR(5)
READ(5,5,END=61)IC
ENDIF
C
C CALL RUNTIME LIBRARY
C

```

```

IERR=LIB$ERASE_PAGE(1,1)
C
IERR=LIB$INIT_TIMER( )
C
C OPEN I/O FILES
C
OPEN(2,FILE='FOR002.DAT',TYPE='OLD')
REWIND(2)
READ(2,*)NJCT
REWIND(2)
IF(NJCT.GE.0)THEN
OPEN(8,FILE='FOR008.DAT',TYPE='OLD')
REWIND(8)
ENDIF
IF(IG.EQ.'Y')THEN
OPEN(12,FILE=NECNAM,TYPE='NEW')
REWIND(12)
OPEN(18,FILE=IGUNAM,TYPE='NEW')
REWIND(18)
ENDIF
C
C READ INPUT DATA ASSOCIATE WITH BODIES
C
CALL BODIN(MXBDND,MXEDGS,DATNOD,NCONN,NNODES,
$NEDGES,MXEXCI,EXCITE,NEXCIT,NJCT,MNJUN,MXWVLT,NWVLT,
$NODVLT,CWVLT,IPAT,ITOT)
C
C NBJUN(I):BODY NODE NUMBER OF THE ITH JUNCTION I=1,MNJUN
C NWJUN(I):WIRE NODE NUMBER OF THE ITH JUNCTION I=1,MNJUN
C
IF(NJCT.GE.0) THEN
C
C READ INPUT DATA ASSOCIATE WITH WIRES
C
CALL WIRIN(MXWNOD,MXWMLT,MXWSEG,NWNOD,NWSEG,NWUNKS,WNODE,MULTW,
$NSEGC,WSEGH,RAD,INSEG,DATNOD,NNODES,NJCT,MNJUN,NWJUN,NBJUN)
C
C TO RECALCULATE MULTIPLICITY OF EACH WIRE NODE IF
C THERE ARE ANY GROUND PLANE ATTACHED
C
CALL ADWMUL(NWNOD,WNODE,MULTW,NWUNKS)
ELSE
C
C TO AVOID ZERO DIMENSIONAL ARRAY
C

```

```

      NWNOD=1
      NWSEG=1
      NWUNKS=0
      ENDIF
C
C TO FIND FACES DATALIST OF BODIES
C
      CALL FACMUL(NCONN,NEDGES,IWORK,NBOUND,MXFACE,NFACES,NUNKNB)
C
C TO NUMBER ALL EDGES OF BODIES
C
      CALL ORTFAC(NCONN,NBOUND,NFACES,NEDGES,MXDJBD,IWORK,ISTART,NBODYS,
$           NBE)
      DO 10 I=1,NBODYS
C
C     TO FIND NORMAL VECTOR OF CLOSED BODY
C
      IF(NBE(I).EQ.0)CALL CLSBOD(DATNOD,NCONN,NBOUND,NNODES,NEDGES,
$           NFACES,I,ISTART)
C
C     TO PRINT THE EDGES AND THE VERTICES OF EACH FACE
C
      CALL FACOUT(NCONN,NBOUND,ISTART,I,NEDGES,NFACES,NBODYS,
$           DATNOD,NNODES)
C
10  CONTINUE
C
      IF(IS.EQ.'Y')THEN
      WRITE(6,*)' '
      WRITE(6,*)'*** INPUT DATA FILE IN NEC FORMAT COMPLETED ***'
      WRITE(6,*)' '
      WRITE(6,*)'*** INPUT DATA FILE IN IGUANA FORMAT COMPLETED ***'
      WRITE(6,*)' '
      WRITE(6,*)'*** USE MODEL MAKER IN IGUANA TO PLOT GEOMETRY **'
      WRITE(6,*)' '
      WRITE(6,*)'*** IF DATA TRANSFER TO OTHER COMPUTER SYSTEM CHECK
$ DATA AGAIN **'
      WRITE(6,*)' '
      STOP
      ENDIF
C
C     EDGEF : TO RECALCULATE MULTIPLICITY OF EACH EDGE IF
C             THERE ARE ANY GROUND PLANE ATTACHED
C     BODPAR : TO CALCULATE PARAMETERS ASSOCIATED WITH BODIES
C     EDGFAC : TO FIND FACES ATTACHED TO EACH EDGE

```

```

C
  CALL ADBMUL(NNODES,NEDGES,DATNOD,NCONN,NUNKNB)
  CALL BODPAR(DATNOD,NCONN,NBOUND,NNODES,NEDGES,NFACES,NUNKNB)
  CALL EDGFAC(NCONN,NEDGES,NBOUND,NFACES,IEDGF,MULT1)
C
C NUNKNT: NUMBER OF TOTAL UNKNOWNNS
C NUNKNB: NUMBER OF BODIES UNKNOWNNS
C NWUNKS: NUMBER OF WIRES UNKNOWNNS
C
  NUNKNT=NUNKNB+NWUNKS
C
C CHECK IF DECLARED DIMENSION IS ENOUGH
C
  IF(NUNKNT.GT.MXUNKN) THEN
  WRITE(6,*)'DECLARED DIMENSION FOR TOTAL UNKNOWNNS
$ IS INSUFFICIENT'
  STOP
  ENDIF
C
  WRITE(6,*)'*** GEOMETRY PART COMPLETED ***'
C
  IF(IC.EQ.'Y')THEN
C
  IERR=LIB$STAT_TIMER( 2, ITG1, )
  TGG=FLOAT(ITG1)/6000.
C
  WRITE(6,*)'CPUTIME FOR GEOMETRY = ',TGG,' MINUTES'
  ENDIF
C
  WRITE(6,*)' '
  WRITE(6,*)' EXECUTION ..... '
C
C TO FILL AND SOLVE THE MATRIX EQUATION
C
  CALL SOLTN(CZ,CV,NUNKNT,IPVT,CWORK,DATNOD,NCONN,NBOUND,NEDGES,
$   NFACES,NNODES,IEDGF,MULT1,EXCITE,NEXCIT,
$   MXFREQ,IWORK,NUNKNB,NWNOD,NWSEG,NWUNKS,WNODE,MULTW,NSEGC,
&   WSEGH,RAD,INSEG,NJCT,MNJUN,NWJUN,NBJUN,
&   WIRSUM,ANG,MNJFACE,NJFACE,MIFACE,MXWMLT,NWVLT,NODVLT,CWVLT,
&   IPAT,ITOT)
  END

```

A.2 DATGEN

```
C=====
      SUBROUTINE DATGEN
C-----
* THIS SUBROUTINE CREATES INPUT DATA FOR BODY AND WIRE
* FOR JUNCTION PROGRAM
* MXFACE=MAXIMUM NUMBER OF FACES EXPECTED.
* MXBNOD=MAXIMUM NUMBER OF BODY NODES.
* MXBEDG=MAXIMUM NUMBER OF BODY EDGES.
* CLOSE=THE MAXIMUM SPACE BETWEEN TWO POINTS AND STILL ATTACH.
C-----
      PARAMETER(MXBNOD=1000,MXBEDG=1000,MXFACE=1000)
      PARAMETER(CLOSE=1.E-4)
C
      CHARACTER*1 ID,IN
1      FORMAT(A1)
      INTEGER IBEDGE(? ,MXBEDG),IFACES(3,MXFACE),ITRAK(MXBEDG/2)
      REAL      BNODES(3,MXBNOD)
C
      COMMON /BODY/  NBNODS,NBEDGS,NFACES
      COMMON /SPACE/ SPACE
C
      SPACE=CLOSE
      ID='N'
      IF(ID.EQ.'Y')THEN
          CALL READIN(BNODES,IBEDGE)
      ELSEIF(ID.EQ.'N')THEN
          PRINT*,'***** DEFINE THE BODY AND WIRE *****'
          PRINT*,'HOW MANY BODY/WIRE JUNCTIONS ?'
          PRINT*,'IF NO JUNCTION TYPE 0'
          PRINT*,'IF NO WIRE TYPE -1'
100     CALL EOFCLR(5)
          READ(5,*,END=100)NJUN
          NBNODS=0
          NBEDGS=0
          ELSEIF(ID.EQ.'Q')THEN
              GOTO 999
          ELSE
              GOTO 100
          ENDIF
200     CALL EOFCLR(5)
          PRINT*,'DO YOU WISH TO'
          PRINT*,' Z READ GEOMETRY DATA IN NEC FORMAT'
          PRINT*,' A CREATE A QUADRANGLE'
```

```

PRINT*, ' B CREATE A CYLINDER'
PRINT*, ' C CREATE A CONE'
PRINT*, ' D CREATE A DISK'
PRINT*, ' E CREATE A SPHERE'
PRINT*, ' G CREATE A STRAIGHT WIRE'
PRINT*, ' H QUERY NODES, EDGES, AND/OR FACES'
PRINT*, ' I ADD, DELETE, OR MOVE A NODE OR EDGE'
PRINT*, ' J ENTER TEST DATA'
PRINT*, ' K START OVER'
PRINT*, ' Q QUIT - SAVE NOTHING'
PRINT*, 'INPUT "Z" "A-K" OR "Q"'
READ(S,1,END=200)ID
IF(ID.EQ.'Z')THEN
  IN='Z'
  CALL NECDAT(NJUN)
ELSEIF(ID.EQ.'A')THEN
  CALL BQUAD(BNODES,IBEDGE)
ELSEIF(ID.EQ.'B')THEN
  CALL CYLIND(BNODES,IBEDGE)
ELSEIF(ID.EQ.'C')THEN
  CALL CONE(BNODES,IBEDGE)
ELSEIF(ID.EQ.'D')THEN
  CALL DISK(BNODES,IBEDGE)
ELSEIF(ID.EQ.'E')THEN
  CALL SPHERE(BNODES,IBEDGE)
ELSEIF(ID.EQ.'G')THEN
  CALL WIRDAT
ELSEIF(ID.EQ.'H')THEN
  CALL QUERY(BNODES,IBEDGE,IFACES,ITRAK)
ELSEIF(ID.EQ.'I')THEN
  CALL CHANGE(BNODES,IBEDGE)
ELSEIF(ID.EQ.'J')THEN
  IF(IN.NE.'Z') CALL WRIOUT(BNODES,IBEDGE,NJUN)
300 CALL EOFCLR(5)
  PRINT*, '***** DEFINE TEST PARAMETERS *****'
  ID='Y'
  IF(ID.EQ.'Y')THEN
    CALL TSTDAT
  ELSEIF(ID.NE.'N')THEN
    GOTO 999
  ENDIF
  GOTO 999
ELSEIF(ID.EQ.'K')THEN
  NBNODS=0
  NBEDGS=0

```

```

ELSEIF(ID.EQ.'Q')THEN
  STOP
ENDIF
GOTO 200
999 CONTINUE
END

```

A.3 READIN

```

C=====
SUBROUTINE READIN(BNODES,IBEDGE)
C=====
DIMENSION BNODES(3,*),IBEDGE(2,*)
COMMON /BODY/NBNODS,NBEDGS,NFACES
REWIND(2)
READ(2,*)NBNODS,NBEDGS
DO 10 J=1,NBNODS
  READ(2,*)NODE,(BNODES(I,NODE),I=1,3)
10 CONTINUE
DO 20 J=1,NBEDGS
  READ(2,*)NEDGE,(IBEDGE(I,NEDGE),I=1,2)
20 CONTINUE
RETURN
END

```

A.4 BQUAD

```

C=====
SUBROUTINE BQUAD(BNODES,IBEDGE)
C=====
DIMENSION BNODES(3,*),IBEDGE(2,*),C(3,4)
COMMON /BODY/NBNODS,NBEDGS,NFACES
C
PRINT*,'ENTER (X,Y,Z) FOR THE 4 CORNERS IN SEQUENCE'
DO 10 I=1,4
  PRINT*,'? CORNER ',I
9   CALL EOFCLR(5)
  READ(5,*,END=9)C(1,I),C(2,I),C(3,I)
10  CONTINUE
PRINT*,'HOW MANY EDGES ALONG THE SIDE FORMED BY'
PRINT*,'THE FIRST 2 CORNERS?'

```

```

11  CALL EOFCLR(5)
    READ(5,*,END=11)NSEG
    N=NSEG+1
    PRINT*, 'HOW MANY EDGES ALONG THE ADJACENT SIDE ?'
21  CALL EOFCLR(5)
    READ(5,*,END=21)MSEG
    M=MSEG+1
    IF(NSEG.EQ.0 .OR. MSEG.EQ.0)RETURN
    NNODE1=NBNOES+1
    NVERTS=N*M
    ILDBNOD=NBNOES
    NBNOES=NBNOES+NVERTS
    L1=NNODE1
    L2=NNODE1+NSEG
    L3=NBNOES
    L4=NBNOES-NSEG

```

```

C
* ASSIGN THE VERTICES TO THE PROPER ARRAY ELEMENTS
C

```

```

    BNODES(1,L1)=C(1,1)
    BNODES(2,L1)=C(2,1)
    BNODES(3,L1)=C(3,1)
    BNODES(1,L2)=C(1,2)
    BNODES(2,L2)=C(2,2)
    BNODES(3,L2)=C(3,2)
    BNODES(1,L3)=C(1,3)
    BNODES(2,L3)=C(2,3)
    BNODES(3,L3)=C(3,3)
    BNODES(1,L4)=C(1,4)
    BNODES(2,L4)=C(2,4)
    BNODES(3,L4)=C(3,4)

```

```

C
* COMPUTE EDGE POINTS ALONG THE TOP AND BOTTOM.
C

```

```

    DX1=(BNODES(1,L2)-BNODES(1,L1))/NSEG
    DY1=(BNODES(2,L2)-BNODES(2,L1))/NSEG
    DZ1=(BNODES(3,L2)-BNODES(3,L1))/NSEG
    DX2=(BNODES(1,L3)-BNODES(1,L4))/NSEG
    DY2=(BNODES(2,L3)-BNODES(2,L4))/NSEG
    DZ2=(BNODES(3,L3)-BNODES(3,L4))/NSEG
    DO 30 I=1,N-2
        J=I+L1
        BNODES(1,J)=BNODES(1,L1)+I*DX1
        BNODES(2,J)=BNODES(2,L1)+I*DY1
        BNODES(3,J)=BNODES(3,L1)+I*DZ1

```



```

        J=I+L4
        BNODES(1,J)=BNODES(1,L4)+I*DX2
        BNODES(2,J)=BNODES(2,L4)+I*DY2
        BNODES(3,J)=BNODES(3,L4)+I*DZ2
30    CONTINUE
        L4=L4-1
        L1=L1-1
C
* COMPUTE THE INNER VERTICES.
C
        DO 40 I=1,N
            J1=L4+I
            J2=L1+I
            DX=(BNODES(1,J1)-BNODES(1,J2))/MSEG
            DY=(BNODES(2,J1)-BNODES(2,J2))/MSEG
            DZ=(BNODES(3,J1)-BNODES(3,J2))/MSEG
            DO 50 J=1,M-2
                JN=J*N+J2
                BNODES(1,JN)=BNODES(1,J2)+J*DX
                BNODES(2,JN)=BNODES(2,J2)+J*DY
                BNODES(3,JN)=BNODES(3,J2)+J*DZ
50    CONTINUE
40    CONTINUE
        MODE=0
        CALL KNIT(IBEDGE,NNODE1,ILDBEDG,N,M,MODE)
        CALL ATTACH(BNODES,IBEDGE,ILDBNOD,ILDBEDG)
        RETURN
        END

```

A.5 CYLIND

```

C=====
        SUBROUTINE CYLIND(BNODES,IBEDGE)
C=====
        DIMENSION BNODES(3,*),IBEDGE(2,*)
        CHARACTER*1 IDF,IDM,IDB
1    FORMAT(A1)
        COMMON /BODY/NBNODS,NBEDGS,NFACES
C
* THE FOLLOWING STATEMENT FUNCTIONS ARE FOR USE ON MACHINES
* THAT DON'T HAVE THESE EXTENTIONS TO THE INTRINSIC FUNCTIONS
C
* ENTER CENTER POINT AND ZERO DEGREE POINT ON THE CIRCUMFERENCE

```

* OF THE FRONT AND BACK FACE, AND ANGLE.

```
C
  PRINT*, 'ENTER (X,Y,Z) OF THE CENTER POINT OF THE FRONT FACE'
2  CALL EOFCLR(5)
  READ(5,*,END=2)C1X,C1Y,C1Z
  PRINT*, 'ENTER (X,Y,Z) OF ZERO DEGREE POINT ON THE CIRCUMFERENCE'
  PRINT*, 'OF THE FRONT FACE'
3  CALL EOFCLR(5)
  READ(5,*,END=3)P1X,P1Y,P1Z
  PRINT*, 'ENTER (X,Y,Z) OF THE CENTER POINT OF THE BACK FACE'
4  CALL EOFCLR(5)
  READ(5,*,END=4)C2X,C2Y,C2Z
  CALL EOFCLR(5)
5  PRINT*, 'ENTER (X,Y,Z) OF ZERO DEGREE POINT ON THE CIRCUMFERENCE'
  PRINT*, 'OF THE BACK FACE'
  READ(5,*,END=5)P2X,P2Y,P2Z
11 CALL EOFCLR(5)
  PRINT*, 'HOW MANY EDGES AROUND THE CIRCUMFERENCE?'
  READ(5,*,END=11)NSEG
21 CALL EOFCLR(5)
  PRINT*, 'HOW MANY EDGES ALONG THE LENGTH?'
  READ(5,*,END=21)MSEG
  IF(NSEG.EQ.0 .OR. MSEG.EQ.0)RETURN
  M=MSEG+1
31 CALL EOFCLR(5)
  PRINT*, 'DO YOU WANT THE CYLINDER TO BE SLOTTED?'
  READ(5,1,END=31)IDM
  IF(IDM.EQ.'Y')THEN
    N=NSEG+1
41 CALL EOFCLR(5)
  PRINT*, 'INPUT START AND END ANGLES OF FRONT FACE'
  READ(5,*,END=41)FBGN,FEND
51 CALL EOFCLR(5)
  PRINT*, 'INPUT START AND END ANGLES OF BACK FACE'
  READ(5,*,END=51)BBGN,BEND
  ELSE
    N=NSEG
    FBGN=0.
    FEND=360.
    BBGN=0.
    BEND=360.
  ENDIF
61 CALL EOFCLR(5)
  PRINT*, 'CLOSE FRONT?'
  READ(5,1,END=61)IDF
```

```

71  CALL EOFCLR(5)
    PRINT*, 'CLOSE BACK?'
    READ(5,1,END=71)IDB
    MODE=0
    IF(IDB.EQ.'Y')MODE=MODE+1
    IF(IDM.EQ.'N')MODE=MODE+2
    IF(IDF.EQ.'Y')MODE=MODE+4
    N1=N-1
    NVERTS=N*M
    ILDBNOD=NBNOBS
    NNODE1=NBNOBS+1
    NBNOBS=NBNOBS+NVERTS
    IF(IDF.EQ.'Y')THEN
        BNODES(1,NNODE1)=C1X
        BNODES(2,NNODE1)=C1Y
        BNODES(3,NNODE1)=C1Z
        NBNOBS=NBNOBS+1
        NNODE1=NNODE1+1
    ENDIF
    NODBAK=NBNOBS-N1
C
* U HAT=(C1-C2)/MAGNITUDE(C1-C2)
C
    CALL UNTVEC(C1X,C1Y,C1Z,C2X,C2Y,C2Z,UX,UY,UZ)
C
* PC1=P1-C1 AND PC2=P2-C2
C
    PC1X=P1X-C1X
    PC1Y=P1Y-C1Y
    PC1Z=P1Z-C1Z
    PC2X=P2X-C2X
    PC2Y=P2Y-C2Y
    PC2Z=P2Z-C2Z
C
* V1=(P1-C1)CROSS U HAT AND V2=(P2-C2)CROSS U HAT
C
    CALL XPROD(PC1X,PC1Y,PC1Z,UX,UY,UZ,V1X,V1Y,V1Z)
    CALL XPROD(PC2X,PC2Y,PC2Z,UX,UY,UZ,V2X,V2Y,V2Z)
    IF(FEND.EQ.360. .AND. FBGN.EQ.0.)THEN
        DQF=(FEND-FBGN)/N
        DQB=(BEND-BBGN)/N
    ELSE
        DQF=(FEND-FBGN)/N1
        DQB=(BEND-BBGN)/N1
    ENDIF

```

```

DO 10 I=0,N1
  QF=I*DQF+FBGN
  QB=I*DQB+BBGN
  SINQF=SIND(QF)
  SINQB=SIND(QB)
  COSQF=COSD(QF)
  COSQB=COSD(QB)
  BNODES(1,NNODE1+I)=C1X+PC1X*COSQF+V1X*SINQF
  BNODES(2,NNODE1+I)=C1Y+PC1Y*COSQF+V1Y*SINQF
  BNODES(3,NNODE1+I)=C1Z+PC1Z*COSQF+V1Z*SINQF
  BNODES(1,NODBAK+I)=C2X+PC2X*COSQB+V2X*SINQB
  BNODES(2,NODBAK+I)=C2Y+PC2Y*COSQB+V2Y*SINQB
  BNODES(3,NODBAK+I)=C2Z+PC2Z*COSQB+V2Z*SINQB
10 CONTINUE
C
* COMPUTE THE INNER VERTICES
C
DO 20 I=0,N1
  DX=(BNODES(1,NODBAK+I)-BNODES(1,NNODE1+I))/MSEG
  DY=(BNODES(2,NODBAK+I)-BNODES(2,NNODE1+I))/MSEG
  DZ=(BNODES(3,NODBAK+I)-BNODES(3,NNODE1+I))/MSEG
DO 30 J=1,M-2
  JN=J*N+NNODE1
  BNODES(1,JN+I)=BNODES(1,NNODE1+I)+J*DX
  BNODES(2,JN+I)=BNODES(2,NNODE1+I)+J*DY
  BNODES(3,JN+I)=BNODES(3,NNODE1+I)+J*DZ
30 CONTINUE
20 CONTINUE
IF(IDB.EQ.'Y')THEN
  NBNODS=NBNODS+1
  BNODES(1,NBNODS)=C2X
  BNODES(2,NBNODS)=C2Y
  BNODES(3,NBNODS)=C2Z
ENDIF
CALL KNIT(IBEDGE,NNODE1,ILDBEDG,N,M,MODE)
CALL ATTACH(BNODES,IBEDGE,ILDBNOD,ILDBEDG)
RETURN
END

```

A.6 CONE

```

C=====
SUBROUTINE CONE(BNODES,IBEDGE)

```

```

C=====
      DIMENSION BNODES(3,*),IBEDGE(2,*)
      COMMON /BODY/NBNODS,NBEDGS,NFACES
C
* THE FOLLOWING STATEMENT FUNCTIONS ARE FOR USE ON MACHINES
* THAT DON'T HAVE THESE EXTENTIONS TO THE INTRINSIC FUNCTIONS
C
2  CALL EOFCLR(5)
   PRINT*, 'INPUT (X,Y,Z) OF THE POINT OF THE CONE'
   READ(5,*,END=2)C2X,C2Y,C2Z
3  CALL EOFCLR(5)
   PRINT*, 'INPUT (X,Y,Z) OF THE CENTER POINT OF THE BASE'
   READ(5,*,END=3)C1X,C1Y,C1Z
4  CALL EOFCLR(5)
   PRINT*, 'INPUT (X,Y,Z) OF A POINT ON THE BACK CIRCUMFERENCE'
   READ(5,*,END=4)PX,PY,PZ
11 CALL EOFCLR(5)
   PRINT*, 'HOW MANY EDGES AROUND THE CIRCUMFERENCE?'
   READ(5,*,END=11)NSEG
   N=NSEG
12 CALL EOFCLR(5)
   PRINT*, 'HOW MANY FROM THE POINT TO THE CIRCUMFERENCE?'
   READ(5,*,END=12)MSEG
   IF(NSEG.EQ.0 .OR. MSEG.EQ.0)RETURN
   N1=N-1
   NPOINT=NBNODS+1
   NNODE1=NPOINT+1
   ILDBNOD=NBNODS
   NVERTS=N*MSEG+1
   NBNODS=NBNODS+NVERTS
   NODBAK=NBNODS-N1
   BNODES(1,NPOINT)=C2X
   BNODES(2,NPOINT)=C2Y
   BNODES(3,NPOINT)=C2Z
C
*-----COMPUTE BACK POINTS-----
* U HAT=(C1-C2)/MAGNITUDE(C1-C2)
C
      CALL UNTVEC(C1X,C1Y,C1Z,C2X,C2Y,C2Z,UX,UY,UZ)
C
* PC1=P-C1
C
      PC1X=PX-C1X
      PC1Y=PY-C1Y
      PC1Z=PZ-C1Z

```

```

C
* V1= U HAT CROSS PC1
C
CALL XPROD(UX,UY,UZ,PC1X,PC1Y,PC1Z,V1X,V1Y,V1Z)
DQ=360./N
DO 10 I=0,N1
  Q=I*DQ
  SINQ=SIND(Q)
  COSQ=COSD(Q)
  BNODES(1,NODBAK+I)=C1X+PC1X*COSQ+V1X*SINQ
  BNODES(2,NODBAK+I)=C1Y+PC1Y*COSQ+V1Y*SINQ
  BNODES(3,NODBAK+I)=C1Z+PC1Z*COSQ+V1Z*SINQ
10 CONTINUE
C
*-----COMPUTE INNER VERTICES-----
C
DO 20 I20=0,N1
  DX=(BNODES(1,NODBAK+I20)-BNODES(1,NPOINT))/MSEG
  DY=(BNODES(2,NODBAK+I20)-BNODES(2,NPOINT))/MSEG
  DZ=(BNODES(3,NODBAK+I20)-BNODES(3,NPOINT))/MSEG
  DO 30 I30=0,MSEG-1
    I=NNODE1+I30*N+I20
    I301=I30+1
    BNODES(1,I)=BNODES(1,NPOINT)+I301*DX
    BNODES(2,I)=BNODES(2,NPOINT)+I301*DY
    BNODES(3,I)=BNODES(3,NPOINT)+I301*DZ
30 CONTINUE
20 CONTINUE
MODE=6
C
* ADJUST THE M DIMENSION SO THAT THE BODY LOOKS LIKE A CYLINDER
* WHEN CALLING KNIT
C
CALL KNIT(IBEDGE,NNODE1,ILDBEDG,N,MSEG,MODE)
CALL ATTACH(BNODES,IBEDGE,ILDBNOD,ILDBEDG)
RETURN
END

```

A.7 DISK

```

C=====
SUBROUTINE DISK(BNODES,IBEDGE)
C=====

```

```

        DIMENSION BNODES(3,*),IBEDGE(2,*)
        COMMON /BODY/NBNODS,NBEDGS,NFACES
C
* THE FOLLOWING STATEMENT FUNCTIONS ARE FOR USE ON MACHINES
* THAT DON'T HAVE THESE EXTENTIONS TO THE INTRINSIC FUNCTIONS
C
2   CALL EOFCLR(5)
    PRINT*, 'INPUT (X,Y,Z) OF THE CENTER POINT OF THE DISK'
    READ(5,*,END=2)C1X,C1Y,C1Z
3   CALL EOFCLR(5)
    PRINT*, 'INPUT (X,Y,Z) OF A POINT ON THE CIRCUMFERENCE'
    READ(5,*,END=3)PX,PY,PZ
4   CALL EOFCLR(5)
    PRINT*, 'INPUT (X,Y,Z) OF A POINT PERPENDICULAR TO'
    PRINT*, 'THE CENTER OF THE DISK'
    READ(5,*,END=4)C2X,C2Y,C2Z
11  CALL EOFCLR(5)
    PRINT*, 'HOW MANY EDGES AROUND THE CIRCUMFERENCE?'
    READ(5,*,END=11)NSEG
    N=NSEG
12  CALL EOFCLR(5)
    PRINT*, 'HOW MANY EDGES FROM THE CENTER TO THE CIRCUMFERENCE?'
    READ(5,*,END=12)MSEG
    IF(NSEG.EQ.0 .OR. MSEG.EQ.0)RETURN
    N1=N-1
    NPOINT=NBNODS+1
    NNODE1=NPOINT+1
    ILDBNOD=NBNODS
    NVERTS=N*MSEG+1
    NBNODS=NBNODS+NVERTS
    NODBAK=NBNODS-N1
    BNODES(1,NPOINT)=C1X
    BNODES(2,NPOINT)=C1Y
    BNODES(3,NPOINT)=C1Z
C
*-----COMPUTE OUTER POINTS-----
* U HAT=(C1-C2)/MAGNITUDE(C1-C2)
C
    CALL UNTVEC(C1X,C1Y,C1Z,C2X,C2Y,C2Z,UX,UY,UZ)
C
* PC1=P-C1
C
    PC1X=PX-C1X
    PC1Y=PY-C1Y
    PC1Z=PZ-C1Z

```

```

C
* V1= U HAT CROSS PC1
C
CALL XPROD(UX,UY,UZ,PC1X,PC1Y,PC1Z,V1X,V1Y,V1Z)
DQ=360./N
DO 10 I=0,N1
  Q=I*DQ
  SINQ=SIND(Q)
  COSQ=COSD(Q)
  BNODES(1,NODBAK+I)=C1X+PC1X*COSQ+V1X*SINQ
  BNODES(2,NODBAK+I)=C1Y+PC1Y*COSQ+V1Y*SINQ
  BNODES(3,NODBAK+I)=C1Z+PC1Z*COSQ+V1Z*SINQ
10 CONTINUE
C
*-----COMPUTE INNER VERTICES-----
C
DO 20 I=0,N1
  DX=(BNODES(1,NODBAK+I)-BNODES(1,NPOINT))/MSEG
  DY=(BNODES(2,NODBAK+I)-BNODES(2,NPOINT))/MSEG
  DZ=(BNODES(3,NODBAK+I)-BNODES(3,NPOINT))/MSEG
  DO 30 J=0,MSEG-1
    INDEX=NNODE1+J*N+I
    J1=J+1
    BNODES(1,INDEX)=BNODES(1,NPOINT)+J1*DX
    BNODES(2,INDEX)=BNODES(2,NPOINT)+J1*DY
    BNODES(3,INDEX)=BNODES(3,NPOINT)+J1*DZ
30 CONTINUE
20 CONTINUE
MODE=6
C
* ADJUST THE M DIMENSION SO THAT THE BODY LOOKS LIKE A CYLINDER
* WHEN CALLING KNIT
C
CALL KNIT(IBEDGE,NNODE1,ILDBEDG,N,MSEG,MODE)
CALL ATTACH(BNODES,IBEDGE,ILDBNOD,ILDBEDG)
RETURN
END

```

A.8 SPHERE

```

C=====
SUBROUTINE SPHERE(BNODES,IBEDGE)
C=====

```



```

        DIMENSION BNODES(3,*),IBEDGE(2,*),FLEX(30)
        CHARACTER*1 ID
1       FORMAT(A1)
        COMMON /BODY/NBNODS,NBEDGS,NFACES
C
* THE FOLLOWING STATEMENT FUNCTIONS ARE FOR USE ON MACHINES
* THAT DON'T HAVE THESE EXTENTIONS TO THE INTRINSIC FUNCTIONS
C
100    CALL EOFCLR(5)
        PRINT*, 'INPUT (X,Y,Z) OF THE CENTER POINT'
        READ(5,*,END=100)CX,CY,CZ
11     CALL EOFCLR(5)
        PRINT*, 'INPUT THE RADIUS'
        READ(5,*,END=11)R
2      CALL EOFCLR(5)
        PRINT*, 'INPUT (X,Y,Z) OF A POINT IN THE NORTH POLE DIRECTION'
        READ(5,*,END=2)PX,PY,PZ
3      CALL EOFCLR(5)
        PRINT*, 'INPUT (X,Y,Z) OF A POINT IN THE EQUATORIAL'
        PRINT*, '(0 DEGREES) DIRECTION'
        READ(5,*,END=3)EX,EY,EZ
C
* MAKE POLE AND EQUATOR DIRECTIONAL POINTS INTO UNIT VECTORS
C
        CALL UNTVEC(PX,PY,PZ,CX,CY,CZ,U1X,U1Y,U1Z)
        CALL UNTVEC(EX,EY,EZ,CX,CY,CZ,U2X,U2Y,U2Z)
C
* VERIFY THAT THEY ARE AT RIGHT ANGLES
C
        IF(ABS(U1X*U2X+U1Y*U2Y+U1Z*U2Z).GT.1E-8)THEN
            PRINT*, 'YOUR CENTER, POLE, AND EQUATORIAL POINTS'
            PRINT*, 'DON'T FORM A RIGHT ANGLE. PLEASE TRY AGAIN.'
            GOTO 100
        ENDIF
C
* FIND THE UNIT VECTOR THAT IS U1 CROSS U2
C
        CALL XPROD(U1X,U1Y,U1Z,U2X,U2Y,U2Z,U3X,U3Y,U3Z)
C
* THE POLE AND EQUATOR POINTS ARE THE RADIUS TIMES THE UNIT VECTORS
C
        R1X=U1X*R
        R1Y=U1Y*R
        R1Z=U1Z*R
        R2X=U2X*R

```

```

R2Y=U2Y*R
R2Z=U2Z*R
R3X=U3X*R
R3Y=U3Y*R
R3Z=U3Z*R

```

C

* FIND THE START AND FINISH ANGLES OF THE LONGITUDE AND LATITUDE.

C

```

12 CALL EOFCLR(5)
   PRINT*, 'INPUT THE START AND FINISH ANGLES OF THE LONGITUDE'
   READ(5,*,END=12)BGNLON,ENDLON
13 CALL EOFCLR(5)
   PRINT*, 'HOW MANY EDGES DOWN THE LONGITUDE?'
   READ(5,*,END=13)NSEG
14 CALL EOFCLR(5)
   PRINT*, 'INPUT THE START AND FINISH ANGLES OF THE LATITUDE'
   READ(5,*,END=14)BGNLAT,ENDLAT
15 CALL EOFCLR(5)
   PRINT*, 'HOW MANY EDGES AROUND THE LATITUDE?'
   READ(5,*,END=15)MSEG
   IF(NSEG.EQ.0 .OR. MSEG.EQ.0)RETURN
   MODE=0
   IF(ENDLON.EQ.180.)MODE=MODE+1
   IF(BGNLAT.EQ.0. .AND. ENDLAT.EQ.360.)MODE=MODE+2
   IF(BGNLON.EQ.0.)MODE=MODE+4
   NLONPT=NSEG+1
   IF(MODE.EQ.0 .OR. MODE.EQ.1 .OR. MODE.EQ.4 .OR. MODE.EQ.5)THEN
     NLATPT=MSEG+1
   ELSE
     NLATPT=MSEG
   ENDIF
   IF(MODE.EQ.0 .OR. MODE.EQ.2)THEN
     M=NLONPT
   ELSEIF(MODE.EQ.5 .OR. MODE.EQ.7)THEN
     M=NLONPT-2
   ELSE
     M=NLONPT-1
   ENDIF
   N1=NLATPT-1
   NVERTS=NLATPT*M
   ILDBNOD=NBNODS
   NNODE1=NBNODS+1
   NBNODS=NBNODS+NVERTS
   SUMFLEX=REAL(NSEG)
   DO 30 I=1,NLONPT

```

```

      FLEX(I)=1.
30  CONTINUE
16  CALL EOFCLR(5)
      PRINT*, 'DO YOU WANT UNIFORM LONGITUDINAL SEGMENT LENGTHS?'
      READ(5,1,END=16)ID
      IF(ID.EQ.'N')THEN
40   PRINT*, 'ENTER THE SEGMENT NUMBER AND THE PROPORTION <0 0.>=DONE'
      READ(5,*,END=50)NUMBER,VALUE
      IF(NUMBER.NE.0)THEN
          SUMFLEX=SUMFLEX-FLEX(NUMBER)+VALUE
          FLEX(NUMBER)=VALUE
          GOTO 40
      ENDIF
50   CALL EOFCLR(5)
      ENDIF
      IF(BGNLON.EQ.0)THEN
          BNODES(1,NNODE1)=R1X
          BNODES(2,NNODE1)=R1Y
          BNODES(3,NNODE1)=R1Z
          NBNODS=NBNODS+1
          NNODE1=NNODE1+1
      ENDIF
      DTHETA=ENDLON-BGNLON
      IF(BGNLAT.EQ.0. .AND. ENDLAT.EQ.360.)THEN
          DPSI=360./NLATPT
      ELSE
          DPSI=(ENDLAT-BGNLAT)/N1
      ENDIF
      NODE=NNODE1
      THETA=BGNLON
      DO 10 LONGTU=1,NLONPT
          IF(ANINT(THETA).NE.0. .AND. ANINT(THETA).NE.180.)THEN
              SINQ=SIND(THETA)
              COSQ=COSD(THETA)
              DO 20 LATITU=0,N1
                  PSI=LATITU*DPSI+BGNLAT
                  SINPSI=SIND(PSI)
                  COSPSI=COSD(PSI)
                  BNODES(1,NODE)=R1X*COSQ+R2X*SINQ*COSPSI+R3X*SINQ*SINPSI
                  BNODES(2,NODE)=R1Y*COSQ+R2Y*SINQ*COSPSI+R3Y*SINQ*SINPSI
                  BNODES(3,NODE)=R1Z*COSQ+R2Z*SINQ*COSPSI+R3Z*SINQ*SINPSI
                  NODE=NODE+1
20          CONTINUE
          ENDIF
          THETA=THETA+DTHETA*FLEX(LONGTU)/SUMFLEX

```

```

10 CONTINUE
   IF(ENLON.EQ.180.)THEN
     NBNODS=NBNODS+1
     R4X=CX-(R1X-CX)
     R4Y=CY-(R1Y-CY)
     R4Z=CZ-(R1Z-CZ)
     BNODES(1,NBNODS)=R4X
     BNODES(2,NBNODS)=R4Y
     BNODES(3,NBNODS)=R4Z
   ENDIF
   CALL KNIT(IBEDGE,NNODE1,ILDBEDG,NLATPT,M,MODE)
   CALL ATTACH(BNODES,IBEDGE,ILDBNOD,ILDBEDG)
   RETURN
   END

```

A.9 KNIT

```

C=====
SUBROUTINE KNIT(IBEDGE,NNODE1,ILDBEDG,N,M,MODE)
C=====
C
* ALL BODIES IN THIS PROGRAM CAN BE DESCRIBED BYØ
* 1) AN OPTIONAL FRONT POINT
* 2) AN NXM BODY EITHER OPEN OR CLOSED
* 3) AN OPTIONAL BACK POINT
*
* INPUTØ
* IBEDGE=ARRAY OF EDGES THAT EXIST SO FAR.
* NNODE1=NUMBER OF THE FIRST NODE OF THE NXM BODY OF THIS NEW BODY.
* N=ONE DIMENSION OF THE BODY.
* M=THE OTHER DIMENSION OF THE BODY.
* MODE=THE TYPE OF BODY
*
* FRONT NXM BODY BACK EXAMPLE
* MODE=0 0 0 0 QUADRANGLE, SLOTTED CYLINDER(OPEN ENDS)
* MODE=1 0 0 1 SLOTTED CYLINDER W/CLOSED BACK
* MODE=2 0 1 0 CYLINDER W/OPEN ENDS, SPHERE W/O POLES
* MODE=3 0 1 1 CYLINDER W/OPEN FRONT, SPHERE W/O N POLE
* MODE=4 1 0 0 SLOTTED CYLINDER W/CLOSED FRONT
* MODE=5 1 0 1 SLOTTED SPHERE
* MODE=6 1 1 0 CYLINDER W/ OPEN BACK, DISK, CONE
* MODE=7 1 1 1 CYLINDER W/ CLOSED ENDS, SOLID SPHERE
*
* OUTPUTØ

```

```

* IBEDGE=ARRAY WITH ALL EDGES.
* NBEDGS=NUMBER OF ALL EDGES.
* ILDBEDG=LAST EDGE OF LAST BODY.
C
  DIMENSION IBEDGE(2,*)
  COMMON /BODY/NBNODS,NBEDGS,NFACES
C
  NXEDGE=NBEDGS+1
  ILDBEDG=NBEDGS
C
* COMPUTE EDGE CONNECTIONS.
* IF BACK IS CLOSED, THEN...
C
  IF(MODE.EQ.1.OR.MODE.EQ.3.OR.MODE.EQ.5.OR.MODE.EQ.7)THEN
    LASNOD=NBNODS-1
  ELSE
    LASNOD=NBNODS
  ENDIF
C
* IF FRONT IS CLOSED, THEN ADD EDGES FROM FRONT POINT TO NXM BODY.
C
  IF(MODE.GE.4)THEN
    NBEDGS=NBEDGS+N
    FRSNOD=NNODE1-1
    DO 20 I=0,N-1
      IBEDGE(1,NXEDGE)=FRSNOD
      IBEDGE(2,NXEDGE)=NNODE1+I
      NXEDGE=NXEDGE+1
20    CONTINUE
  ENDIF
  DO 501 K=0,M-2
    J=K*N+NNODE1
C
* GO ACROSS THE N DIMENSION.
C
  DO 5001 I=0,N-2
    IBEDGE(1,NXEDGE)=I+J
    IBEDGE(2,NXEDGE)=I+J+1
    NXEDGE=NXEDGE+1
5001  CONTINUE
C
* IF THE BODY IS CLOSED, THEN ATTACH LAST TO FIRST.
C
  IF(MODE.EQ.2.OR.MODE.EQ.3.OR.MODE.EQ.6.OR.MODE.EQ.7)THEN
    IBEDGE(1,NXEDGE)=J+N-1

```

```

        IBEDGE(2,NXEDGE)=J
        NXEDGE=NXEDGE+1
    ENDIF
C
* GO ACROSS THE MIDDLE.
C
    DO 6001 I=0,N-2
        IBEDGE(1,NXEDGE)=I+J
        IBEDGE(2,NXEDGE)=I+J+N
        NXEDGE=NXEDGE+1
        IBEDGE(1,NXEDGE)=I+J
        IBEDGE(2,NXEDGE)=I+J+N+1
        NXEDGE=NXEDGE+1
6001    CONTINUE
C
* CLOSE THE MIDDLE
C
        IBEDGE(1,NXEDGE)=J+N-1
        IBEDGE(2,NXEDGE)=J+N-1+N
        NXEDGE=NXEDGE+1
C
* IF THE BODY IS CLOSED, THEN CLOSE LAST SLANT.
C
        IF(MODE.EQ.2.OR.MODE.EQ.3.OR.MODE.EQ.6.OR.MODE.EQ.7)THEN
            IBEDGE(1,NXEDGE)=J+N-1
            IBEDGE(2,NXEDGE)=J+N
            NXEDGE=NXEDGE+1
        ENDIF
501    CONTINUE
C
* CLOSE THE BOTTOM.
C
    DO 41 I=LASNOD-N+1,LASNOD-1
        IBEDGE(1,NXEDGE)=I
        IBEDGE(2,NXEDGE)=I+1
        NXEDGE=NXEDGE+1
41    CONTINUE
C
* IF BODY IS CLOSED, THEN CLOSE LAST TO FIRST.
C
        IF(MODE.EQ.2.OR.MODE.EQ.3.OR.MODE.EQ.6.OR.MODE.EQ.7)THEN
            IBEDGE(1,NXEDGE)=LASNOD
            IBEDGE(2,NXEDGE)=LASNOD-N+1
            NXEDGE=NXEDGE+1
        ENDIF

```

```

C
* IF BACK IS CLOSED, THE CONNECT LAST ROW TO BACK POINT.
C
  IF(MODE.EQ.1.OR.MODE.EQ.3.OR.MODE.EQ.5.OR.MODE.EQ.7)THEN
    NBEDGS=NBEDGS+N
    DO 21 I=0,N-1
      IBEDGE(1,NXEDGE)=NBNODS
      IBEDGE(2,NXEDGE)=NBNODS-N+I
      NXEDGE=NXEDGE+1
21  CONTINUE
    ENDIF
    NBEDGS=NXEDGE-1
    RETURN
  END

```

A.10 ATTACH

```

=====
SUBROUTINE ATTACH(BNODES,IBEDGE,ILDBNOD,ILDBEDG)
=====
  DIMENSION BNODES(3,*),IBEDGE(2,*)
  COMMON /BODY/NBNODS,NBEDGS,NFACES
  COMMON /SPACE/SPACE
C
* THE FOLLOWING LINE IS A STATEMENT FUNCTION.
C
  SIZE(X,Y,Z)=SQRT(X*X+Y*Y+Z*Z)
C
* FIND A COMMON NODE.
C
  DO 10 ILDPTR=1,ILDBNOD
    DO 20 NEWPTR=ILDBNOD+1,NBNODS
      XILD=BNODES(1,ILDPTR)
      YILD=BNODES(2,ILDPTR)
      ZILD=BNODES(3,ILDPTR)
      XNEW=BNODES(1,NEWPTR)
      YNEW=BNODES(2,NEWPTR)
      ZNEW=BNODES(3,NEWPTR)
C
* IF IT IS A COMMON NODE...
C
  IF(SIZE(XILD-XNEW,YILD-YNEW,ZILD-ZNEW).LE.SPACE)THEN
    NBNODS=NBNODS-1

```

```

C
* LOOP THROUGH ALL THE EDGES
C
      DO 50 I=ILDBEDG+1,NBEDGS
C
* AND CHANGE ANY THAT HAVE THE NEWEST NODE TO THE OLDER NODE
C
      IF(IBEDGE(1,I).EQ.NEWPTR)THEN
        IBEDGE(1,I)=ILDPTR
      ELSEIF(IBEDGE(2,I).EQ.NEWPTR)THEN
        IBEDGE(2,I)=ILDPTR
      ENDIF
C
* AND DECREASE ANY WITH BIGGER NODE NUMBERS BY ONE.
C
      IF(IBEDGE(1,I).GT.NEWPTR)IBEDGE(1,I)=IBEDGE(1,I)-1
      IF(IBEDGE(2,I).GT.NEWPTR)IBEDGE(2,I)=IBEDGE(2,I)-1
50    CONTINUE
C
* DECREASE BIGGER NODE NUMBERS BY ONE.
C
      DO 60 I=NEWPTR,NBNODS
        BNODES(1,I)=BNODES(1,I+1)
        BNODES(2,I)=BNODES(2,I+1)
        BNODES(3,I)=BNODES(3,I+1)
60    CONTINUE
      GOTC 10
      ENDIF
20    CONTINUE
10    CONTINUE
C
* FIND A COMMON EDGE
C
      DO 30 ILDPTR=1,ILDBEDG
        DO 40 NEWPTR=ILDBEDG+1,NBEDGS
C
* IF IT IS A COMMON EDGE
C
          I1=IBEDGE(1,ILDPTR)
          I2=IBEDGE(2,ILDPTR)
          N1=IBEDGE(1,NEWPTR)
          N2=IBEDGE(2,NEWPTR)
          IF((I1.EQ.N1.AND.I2.EQ.N2).OR.(I1.EQ.N2.AND.I2.EQ.N1))THEN
C
* THEN DELETE THE NEW EDGE BY DECREASING THE REMAINING EDGE NUMBERS

```



```

C
      NBEDGS=NBEDGS-1
      DO 35 I=NEWPTR,NBEDGS
        IBEDGE(1,I)=IBEDGE(1,I+1)
        IBEDGE(2,I)=IBEDGE(2,I+1)
35     CONTINUE
        GOTO 30
      ENDIF
40     CONTINUE
30     RETURN
      END

```

A.11 QUERY

```

C=====
      SUBROUTINE QUERY(BNODES,IBEDGE,IFACES,ITRAK)
C=====
      DIMENSION BNODES(3,*),IBEDGE(2,*),IFACES(3,*),ITRAK(*)
      CHARACTER*1 ID
1     FORMAT(A1)
      COMMON /BODY/BNBODS,NBEDGS,NFACES
C
      ITIME=0
100    CALL EOFCLR(5)
      PRINT*,'INFORMATION ON A NODE, EDGE, OR FACE?'
      PRINT*,'INPUT "N" OR "E" OR "F" OR "Q"'
      READ(5,1,END=100)ID
      IF(ID.EQ.'N')THEN
12     CALL EOFCLR(5)
      PRINT*,'ENTER NODE NUMBER'
      READ(5,*,END=12)ND
      IF(ND.LE.NBNODS)THEN
        X=BNODES(1,ND)
        Y=BNODES(2,ND)
        Z=BNODES(3,ND)
        PRINT*,'NODE ',ND,' IS AT (',X,',',Y,',',Z,')'
        DO 10 IPOINT=1,NBEDGS
          IF(IBEDGE(1,IPOINT).EQ.ND .OR. IBEDGE(2,IPOINT).EQ.ND)
>         PRINT*,'EDGE ',IPOINT,' IS CONNECTED TO IT'
10     CONTINUE
      ELSE
        PRINT*,'NODE ',ND,' DOES NOT EXIST.'

```

```

        ENDIF
    ELSEIF(ID.EQ.'E')THEN
11      CALL EOFCLR(5)
        PRINT*,'ENTER EDGE NUMBER'
        READ(5,*,END=11)IEN
        IF(IEN.LE.NBEDGS)THEN
            IFROM=IBEDGE(1,IEN)
            ITO=IBEDGE(2,IEN)
            PRINT*,'EDGE ',IEN,' GOES FROM NODE ',IFROM,' TO NODE ',ITO
        ELSE
            PRINT*,'EDGE ',IEN,' DOES NOT EXIST.'
        ENDIF
    ELSEIF(ID.EQ.'F')THEN
13      CALL EOFCLR(5)
        PRINT*,'ENTER FACE NUMBER'
        READ(5,*,END=13)IFAC
        IF(IFAC.LE.NFACES)THEN
            ITIME=ITIME+1
            IF(ITIME.EQ.1)CALL FAEMUL(IBEDGE,IFACES,ITRAK,2)
            IONE=IFACES(1,IFAC)
            ITWO=IFACES(2,IFAC)
            ITHREE=IFACES(3,IFAC)
            PRINT*,'FACE ',IFAC,' HAS EDGES ',IONE,ITWO,ITHREE
        ELSE
            PRINT*,'FACE ',IFAC,' DOES NOT EXIST '
        ENDIF
    ELSEIF(ID.EQ.'Q')THEN
        RETURN
    ENDIF
    GOTO 100
END

```

A.12 CHANGE

```

C=====
      SUBROUTINE CHANGE(BNODES,IBEDGE)
C=====
      DIMENSION BNODES(3,*),IBEDGE(2,*)
      CHARACTER*1 ID
1      FORMAT(A1)
      COMMON /BODY/NBNODS,NBEDGS,NFACES
C
100 CALL EOFCLR(5)

```

```

PRINT*, 'DO YOU WANT TO ADD, DELETE, OR MOVE'
PRINT*, 'INPUT "A", "D", "M", OR "Q"'
READ(5,1,END=100)ID
IF(ID.EQ.'A')THEN
200  CALL EOFCLR(5)
    PRINT*, 'DO YOU WISH TO ADD A NODE OR INSERT EDGE?'
    PRINT*, 'INPUT "N", "E", OR "Q"'
    READ(5,1,END=200)ID
    IF(ID.EQ.'N')THEN
        NBNODS=NBNODS+1
11    CALL EOFCLR(5)
        PRINT*, 'INPUT (X,Y,Z) FOR NODE ', NBNODS
        READ(5,*,END=11)X,Y,Z
        BNODES(1,NBNODS)=X
        BNODES(2,NBNODS)=Y
        BNODES(3,NBNODS)=Z
    ELSEIF(ID.EQ.'E')THEN
12    CALL EOFCLR(5)
        PRINT*, 'INPUT EDGE NUMBER AND "FROM" AND "TO" NODES'
        READ(5,*,END=12)IEDGE,IFROM,ITO
        NBEDGS=NBEDGS+1
        DO 50 I=NBEDGS,IEDGE+1,-1
            IBEDGE(1,I)=IBEDGE(1,I-1)
            IBEDGE(2,I)=IBEDGE(2,I-1)
50    CONTINUE
        IBEDGE(1,IEDGE)=IFROM
        IBEDGE(2,IEDGE)=ITO
    ELSEIF(ID.EQ.'Q')THEN
        GOTO 999
    ELSE
        GOTO 200
    ENDIF
ELSEIF(ID.EQ.'D')THEN
300  CALL EOFCLR(5)
    PRINT*, 'DELETE NODES OR EDGES?'
    PRINT*, 'INPUT "N", "E", OR "Q"'
    READ(5,1,END=300)ID
    IF(ID.EQ.'N')THEN
13    CALL EOFCLR(5)
        PRINT*, 'INPUT RANGE OF NODE NUMBERS'
        READ(5,*,END=13)ND1,ND2
        DO 60 ND=ND2,ND1,-1
            IF(ND.LE.NBNODS)THEN
                NBNODS=NBNODS-1
                DO 20 I=ND,NBNODS

```

```

        BNODES(1,I)=BNODES(1,I+1)
        BNODES(2,I)=BNODES(2,I+1)
        BNODES(3,I)=BNODES(3,I+1)
20      CONTINUE
        ISTART=1
        KILLED=0
400     DO 30 I=ISTART,NBEDGS
        IF(IBEDGE(1,I+KILLED).EQ.ND
        > .OR.IBEDGE(2,I+KILLED).EQ.ND)THEN
            KILLED=KILLED+1
            NBEDGS=NBEDGS-1
            ISTART=I
            GOTO 400
        ENDIF
        IBEDGE(1,I)=IBEDGE(1,I+KILLED)
        IBEDGE(2,I)=IBEDGE(2,I+KILLED)
30      CONTINUE
        DO 40 I=1,NBEDGS
            IF(IBEDGE(1,I).GT.ND)IBEDGE(1,I)=IBEDGE(1,I)-1
            IF(IBEDGE(2,I).GT.ND)IBEDGE(2,I)=IBEDGE(2,I)-1
40      CONTINUE
        ELSE
            PRINT*,'NODE ',ND,' DOES NOT EXIST'
        ENDIF
60      CONTINUE
    ELSEIF(ID.EQ.'E')THEN
14      CALL EOFCLR(5)
        PRINT*,'INPUT EDGE NUMBER'
        READ(5,*,END=14)IEN
        IF(IEN.LE.NBEDGS)THEN
            NBEDGS=NBEDGS-1
            DO 10 I=IEN,NBEDGS
                IBEDGE(1,I)=IBEDGE(1,I+1)
                IBEDGE(2,I)=IBEDGE(2,I+1)
10      CONTINUE
        ELSE
            PRINT*,'EDGE ',IEN,' DOES NOT EXIST'
        ENDIF
    ELSEIF(ID.EQ.'Q')THEN
        GOTO 999
    ELSE
        GOTO 300
    ENDIF
    ELSEIF(ID.EQ.'M')THEN
15      CALL EOFCLR(5)

```

```

PRINT*, 'WHICH NODE?'
READ(5,*,END=15)ND
IF(ND.LE.NBNODS)THEN
  X=BNODES(1,ND)
  Y=BNODES(2,ND)
  Z=BNODES(3,ND)
16  CALL EOFCLR(5)
  PRINT*, 'NODE ',ND,' IS AT (',X,',',Y,',',Z,')'
  PRINT*, 'DO YOU WANT TO ENTER (X,Y,Z) OR NODE NUMBER?'
  PRINT*, 'ENTER "X" OR "N"'
  READ(5,1,END=16)ID
  IF(ID.EQ.'X')THEN
2    CALL EOFCLR(5)
    PRINT*, 'INPUT NEW (X,Y,Z)'
    READ(5,*,END=2)X,Y,Z
    BNODES(1,ND)=X
    BNODES(2,ND)=Y
    BNODES(3,ND)=Z
  ELSE
17  CALL EOFCLR(5)
    PRINT*, 'INPUT NODE NUMBER'
    READ(5,*,END=17)NDOLD
    BNODES(1,ND)=BNODES(1,NDOLD)
    BNODES(2,ND)=BNODES(2,NDOLD)
    BNODES(3,ND)=BNODES(3,NDOLD)
  ENDIF
  ELSE
    PRINT*, 'NODE ',ND,' DOES NOT EXIST'
  ENDIF
ELSEIF(ID.EQ.'Q')THEN
  GOTO 999
ENDIF
GOTO 100
999 CONTINUE
CALL PACKIT(BNODES,IBEDGE)
RETURN
END

```

A.13 PACKIT

```

C=====
SUBROUTINE PACKIT(BNODES,IBEDGE)
C=====

```

```

        DIMENSION BNODES(3,*),IBEDGE(2,*)
        COMMON /BODY/NBNODS,NBEDGS,NFACES
        COMMON /SPACE/SPACE
C
* THE FOLLOWING LINE IS A STATEMENT FUNCTION.
C
        SIZE(X,Y,Z)=SQRT(X*X+Y*Y+Z*Z)
C
* FIND A COMMON NODE.
C
        DO 10 ILDPTR=1,NBNODS-1
          DO 20 NEWPTR=ILDPTR+1,NBNODS
            XILD=BNODES(1,ILDPTR)
            YILD=BNODES(2,ILDPTR)
            ZILD=BNODES(3,ILDPTR)
            XNEW=BNODES(1,NEWPTR)
            YNEW=BNODES(2,NEWPTR)
            ZNEW=BNODES(3,NEWPTR)
C
* IF IT IS A COMMON NODE...
C
            IF(SIZE(XILD-XNEW,YILD-YNEW,ZILD-ZNEW).LE.SPACE)THEN
              NBNODS=NBNODS-1
C
* LOOP THROUGH ALL THE EDGES
C
              DO 50 I=1,NBEDGS
C
* AND CHANGE ANY THAT HAVE THE NEWEST NODE TO THE OLDER NODE
C
                IF(IBEDGE(1,I).EQ.NEWPTR)THEN
                  IBEDGE(1,I)=ILDPTR
                ELSEIF(IBEDGE(2,I).EQ.NEWPTR)THEN
                  IBEDGE(2,I)=ILDPTR
                ENDIF
C
* AND DECREASE ANY WITH BIGGER NODE NUMBERS BY ONE.
C
                IF(IBEDGE(1,I).GT.NEWPTR)IBEDGE(1,I)=IBEDGE(1,I)-1
                IF(IBEDGE(2,I).GT.NEWPTR)IBEDGE(2,I)=IBEDGE(2,I)-1
50          CONTINUE
C
* DECREASE BIGGER NODE NUMBERS BY ONE.
C
          DO 60 I=NEWPTR,NBNODS

```

```

        BNODES(1,I)=BNODES(1,I+1)
        BNODES(2,I)=BNODES(2,I+1)
        BNODES(3,I)=BNODES(3,I+1)
60      CONTINUE
        GOTO 10
      ENDIF
20      CONTINUE
10      CONTINUE
C
* FIND A COMMON EDGE
C
      DO 30 ILDPTR=1,NBEDGS-1
        DO 40 NEWPTR=ILDPTR+1,NBEDGS
C
* IF IT IS A COMMON EDGE
C
          IF((IBEDGE(1,ILDPTR).EQ.IBEDGE(1,NEWPTR).AND.IBEDGE(2,ILDPTR).
>EQ.IBEDGE(2,NEWPTR)).OR.(IBEDGE(1,ILDPTR).EQ.IBEDGE(2,NEWPTR).AND.
>IBEDGE(2,ILDPTR).EQ.IBEDGE(1,NEWPTR)))THEN
C
* THEN DELETE THE NEW EDGE BY DECREASING THE REMAINING EDGE NUMBERS
C
          NBEDGS=NBEDGS-1
          DO 35 I=NEWPTR,NBEDGS
            IBEDGE(1,I)=IBEDGE(1,I+1)
            IBEDGE(2,I)=IBEDGE(2,I+1)
35          CONTINUE
          GOTO 30
        ENDIF
40      CONTINUE
30      CONTINUE
      RETURN
      END

```

A.14 WRIOUT

```

C=====
      SUBROUTINE WRIOUT(BNODES,IBEDGE,NJUN)
C=====
      CHARACTER*10 NAME
      DIMENSION BNODES(3,*),IBEDGE(2,*)
      COMMON /BODY/NBNODS,NBEDGS,NFACES
C

```

```

NAME='FOR002.DAT'
4 CALL EOFCLR(5)
3 FORMAT(A10)
OPEN(6,FILE=NAME,TYPE='NEW')
REWIND(6)
IF(NJUN.LE.0)THEN
NJCT=NJUN
NJUN=1
ELSE
NJCT=1
ENDIF
WRITE(6,1)NJCT,NJUN
WRITE(6,1)NBNODS,NBEDGS
1 FORMAT(3I5)
DO 10 J=1,NBNODS
WRITE(6,2)J,(BNODES(I,J),I=1,3)
10 CONTINUE
2 FORMAT(15,1P3E22.14)
DO 20 J=1,NBEDGS
WRITE(6,1)J,(IBEDGE(I,J),I=1,2)
20 CONTINUE
RETURN
END

```

A.15 TSTDAT

```

C=====
SUBROUTINE TSTDAT
C=====
PARAMETER(MXFLD=10,MXEXCI=10)
CHARACTER*1 ID
1 FORMAT(A1)
INTEGER IGNDP(3)
LOGICAL PRINTC,THEV
REAL RFIELD(3,MXFLD)
C
11 CALL EOFCLR(5)
PRINT*,'ENTER THE NUMBER OF SYMMETRY PLANES'
15 CALL EOFCLR(5)
READ(5,*,END=15)NGNDP
IF(NGNDP.EQ.0)THEN
IGNDP(1)=0
IGNDP(2)=0

```



```

      IGNDP(3)=0
    ELSEIF(NGNDP.GE.1.AND.NGNDP.LE.3)THEN
      PRINT*,'ENTER SYMMETRY PLANE TYPE AT X=0'
      PRINT*,' -1 IF THERE IS A P.E.C'
      PRINT*,'  0 IF NO IMAGE PLANE'
      PRINT*,' +1 IF THERE IS A P.M.C'
16     CALL EOFCLR(5)
      READ(5,*,END=16)IGNDP(1)
      PRINT*,'ENTER SYMMETRY PLANE TYPE AT Y=0'
116    CALL EOFCLR(5)
      READ(5,*,END=116)IGNDP(2)
      PRINT*,'ENTER SYMMETRY PLANE TYPE AT Z=0'
216    CALL EOFCLR(5)
      READ(5,*,END=216)IGNDP(3)
    ELSE
      GOTO 15
    ENDIF
17     CALL EOFCLR(5)
      WRITE(3,*)NGNDP
      WRITE(6,*)(IGNDP(I),I=1,3)
C
* PRINTC AND THEV ARE WRITTEN EXPLICITLY AS .TRUE. OR .FALSE.
* BECAUSE PATCH MAY BE RUN ON A DIFFERENT MACHINE THAN BUILD.
* FOR INSTANCE, IF BUILD WERE RUN ON A VAX, THE OUTPUT OF THEV
* IF THEY WERE FALSE WOULD BE F BUT IF PATCH WERE RUN ON A CRAY,
* THE CRAY WOULD NEED A DECIMAL POINT BEFORE THE F
C
21     CALL EOFCLR(5)
      PRINT*,'INPUT IPAT;0 IF NO FAR FIELD,1 OR 2 IF FAR FIELD'
      PRINT*,'IPAT=1 IF A 3 POINT QUADRATURE IS USED'
      PRINT*,'IPAT=2 IF A 1 POINT QUADRATURE IS USED'
      READ(5,*)IPAT
      WRITE(6,*)IPAT
      IF(IPAT.NE.0)THEN
        PRINT*,'INPUT:PHI1,PHI2,NPHI,THETA1,THETA2,NTHETA (DEG)'
        READ(5,*)PHI1,PHI2,NPHI,THETA1,THETA2,NTHETA
        WRITE(6,*)PHI1,PHI2,NPHI,THETA1,THETA2,NTHETA
      ENDIF
      PRINT*,'INPUT ITOT;=0 IF NO CHARGE DENSITY'
      PRINT*,'ITOT;=1 IF CHARGE DENSITY COMPUTATION IS DESIRED'
      READ(5,*)ITOT
      WRITE(6,*)ITOT
      NEXCIT=1
      DO 30 I=1,NEXCIT
        PRINT*,'EXCITATION IS PLANE WAVE, OR VOLTAGE?'

```

```

PRINT*, 'TYPE IN "P", OR "V"'
222 CALL EOFCLR(5)
READ(5,1,END=222)ID
WRITE(6,'(1X,A1)')ID
IF(ID.NE.'V')THEN
PRINT*, 'INPUT THETA, PHI, REAL AND IMAG OF HTHETA,'
PRINT*, 'REAL AND IMAG OF HPHI'
22 CALL EOFCLR(5)
READ(5,*,END=22)THETA,PHI,RHTHET,CHTHET,RHPHI,CHPHI
WRITE(6,'(6(E12.5))')THETA,PHI,RHTHET,CHTHET,RHPHI,CHPHI
ELSEIF(ID.NE.'P')THEN
WRITE(8,'(1X,A1)')ID
PRINT*, 'NUMBER OF VOLTAGE SOURCES ON THE WIRE'
122 CALL EOFCLR(5)
READ(5,*,END=122)NVOLT
WRITE(6,*)NVOLT
WRITE(8,*)NVOLT
DO 5000 III=1,NVOLT
PRINT*, 'DELTA GAP VOLTAGE ON WHICH NODE OF THE WIRE?'
123 CALL EOFCLR(5)
READ(5,*,END=123)NODV
PRINT*, 'VOLTAGE = ? IN COMPLEX FORM, REAL(V), IMAG(V)'
124 CALL EOFCLR(5)
READ(5,*,END=124)RV,XV
WRITE(6,*)NODV,RV,XV
WRITE(8,*)NODV,RV,XV
5000 CONTINUE
ENDIF
30 CONTINUE
C
40 CONTINUE
PRINT*, 'ENTER FREQUENCIES, -1 WHEN DONE'
125 CALL EOFCLR(5)
READ(5,*,END=125)FREQ
WRITE(6,*)FREQ
IF(FREQ.NE.-1)GO TO 40
99 CONTINUE
CLOSE(UNIT=6,DISPOSE='SAVE')
CLOSE(UNIT=8,DISPOSE='SAVE')
RETURN
END

```

A.16 XPROD

```
C=====
SUBROUTINE XPROD(X1,Y1,Z1,X2,Y2,Z2,X3,Y3,Z3)
C=====
X3=Y1*Z2-Y2*Z1
Y3=X2*Z1-X1*Z2
Z3=X1*Y2-X2*Y1
RETURN
END
```

A.17 UNTVEC

```
C=====
SUBROUTINE UNTVEC(X1,Y1,Z1,X2,Y2,Z2,UX,UY,UZ)
C=====
UX=X1-X2
UY=Y1-Y2
UZ=Z1-Z2
SIZE=SQRT(UX*UX+UY*UY+UZ*UZ)
UX=UX/SIZE
UY=UY/SIZE
UZ=UZ/SIZE
RETURN
END
```

A.18 FAEMUL

```
C=====
SUBROUTINE FAEMUL(IBEDGE,IFACES,ITRAK,IFLAG)
C=====
C
* IF IFLAG=1
* THIS SUBROUTINE FILLS IFACES WITH THE NODE NUMBERS THAT FORM THE FACE.
* IF IFLAG=2
* THIS SUBROUTINE FILLS IFACES WITH THE EDGE NUMBERS THAT FORM THE FACE.
* IT RETURNS THE NUMBER OF FACES(NFACES). ITRAK IS A WORK ARRAY.
C
INTEGER IBEDGE(2,*),IFACES(3,*),ITRAK(*)
COMMON /BODY/NBNODS,NBEDGS,NFACES
C
```

```

      NFACES=0
C
* FIND FACES AND LIST THEM IN IFACES.
C
      DO 100 IEDGE=1,NBEDGS-2
        NTRAK=0
C
* LOOK FOR ALL EDGES THAT ATTACH TO EDGE IEDGE AND PUT THEM IN ITRAK.
C
      DO 200 JEDGE=IEDGE+1,NBEDGS
        DO 20 I=1,2
          DO 21 J=1,2
            IF(IBEDGE(I,IEDGE).EQ.IBEDGE(J,JEDGE))THEN
C
* WE HAVE FOUND AN EDGE
C
              NTRAK=NTRAK+1
              ITRAK(NTRAK)=JEDGE
              GOTO 200
            ENDIF
          21      CONTINUE
        20      CONTINUE
      100      CONTINUE
C
* FIND ALL PAIRS OF EDGES THAT FORM A FACE WITH IEDGE.
C
      DO 300 JEDGE=1,NTRAK-1
        DO 301 KEDGE=JEDGE+1,NTRAK
          DO 30 J=1,2
            DO 31 K=1,2
C
* IF THE 2 EDGES IN ITRAK HAVE A COMMON POINT AND
* THE COMMON POINT IS NOT IN COMMON WITH THE IEDGE...
C
              IF(((IBEDGE(J,ITRAK(JEDGE)).EQ.IBEDGE(K,ITRAK(KEDGE)))
                >.AND.(IBEDGE(J,ITRAK(JEDGE)).NE.IBEDGE(1,IEDGE)
                >.AND.IBEDGE(J,ITRAK(JEDGE)).NE.IBEDGE(2,IEDGE)))THEN
C
* THEN WE HAVE FOUND A FACE.
C
              NFACES=NFACES+1
              IF(IFLAG.EQ.1)THEN
C
* PUT THE NODES INTO IFACES
C

```

```

                CALL BFACVT(IBEDGE,IEDGE,ITRAK(JEDGE),ITRAK(KEDGE),
>                NODE1,NODE2,NODE3)
                IFACES(1,NFACES)=NODE1
                IFACES(2,NFACES)=NODE2
                IFACES(3,NFACES)=NODE3
            ELSE
C
* PUT THE EDGES INTO IFACES.
C
                IFACES(1,NFACES)=JEDGE
                IFACES(2,NFACES)=ITRAK(JEDGE)
                IFACES(3,NFACES)=ITRAK(KEDGE)
            ENDIF
            GOTO 300
        ENDIF
    31        CONTINUE
    30        CONTINUE
    301       CONTINUE
    300       CONTINUE
    100      CONTINUE
            RETURN
            END

```

A.19 BFACVT

```

=====
SUBROUTINE BFACVT(IBEDGE,IE1,IE2,IE3,NV1,NV2,NV3)
=====
C
* NV1 IS NODE OPPOSITE EDGE IE1.
* NV2 IS NODE OPPOSITE EDGE IE2.
* NV3 IS NODE OPPOSITE EDGE IE3.
C
    INTEGER IBEDGE(2,*)
    COMMON /BODY/ NBNODS, NBEDGS, NFACES
C
* THE NODE NV1 IS THE NODE THAT EDGES 2 AND 3 HAVE IN COMMON
C
    IF((IBEDGE(1,IE2) EQ IBEDGE(1,IE3)) OR (IBEDGE(1,IE1) EQ IBEDGE(2,IE3)
>)) THEN
        NV1=IBEDGE(1,IE2)
    ELSE
        NV1=IBEDGE(2,IE2)
    ENDIF

```

```

      ENDIF
C
* THE NODE NV2 IS THE NODE THAT EDGES 1 AND 3 HAVE IN COMMON.
C
      IF(IBEDGE(1,IE1).EQ.IBEDGE(1,IE3).OR.IBEDGE(1,IE1).EQ.IBEDGE(2,IE3
>))THEN
          NV2=IBEDGE(1,IE1)
      ELSE
          NV2=IBEDGE(2,IE1)
      ENDIF
C
* THE NODE NV3 IS THE NODE THAT EDGES 1 AND 2 HAVE IN COMMON.
C
      IF(IBEDGE(1,IE1).EQ.IBEDGE(1,IE2).OR.IBEDGE(1,IE1).EQ.IBEDGE(2,IE2
>))THEN
          NV3=IBEDGE(1,IE1)
      ELSE
          NV3=IBEDGE(2,IE1)
      ENDIF
      RETURN
      END

```

A.20 EOFCLR

```

=====
      SUBROUTINE EOFCLR(I)
=====
C=====
* THIS ROUTINE CLEARS AN END-OF-FILE. IT IS USED TO PREVENT THE USER
* FROM BEING BOMBED OFF IF A <CR> IS HIT BY ACCIDENT.
C
      REWIND(I)
      RETURN
      END

```

A.21 WIRDAT

```

C=====
      SUBROUTINE WIRDAT
C-----
C THIS ROUTINE CREATES INPUT DATA OF A STRAIGHT WIRE FOR JUNGRO CODE
C-----

```

```

REAL SH(3)
WRITE(6,*)'THE COORDINATES OF THE END POINT OF THE WIRE?'
WRITE(6,*)'X,Y,Z IN RECTANGULAR COORDINATE SYSTEM'
1  CALL EOFCLR(5)
   READ(5,*,END=1)X1,Y1,Z1
   WRITE(6,*)'THE COORDINATES OF THE OTHER END POINT OF THE WIRE?'
   WRITE(6,*)'X,Y,Z IN RECTANGULAR COORDINATE SYSTEM'
2  CALL EOFCLR(5)
   READ(5,*,END=2)X2,Y2,Z2
   WRITE(6,*)'HOW MANY SEGMENTS ON THE WIRE?'
3  CALL EOFCLR(5)
   READ(5,*,END=3)NSEG
   WRITE(6,*)'RADIUS OF WIRE = ?'
4  CALL EOFCLR(5)
   READ(5,*,END=4)RWIRE
   NODE=NSEG+1
   WRITE(8,*) NODE,NSEG
   SH(1)=X2-X1
   SH(2)=Y2-Y1
   SH(3)=Z2-Z1
   S=SQRT(SH(1)*SH(1)+SH(2)*SH(2)+SH(3)*SH(3))
   DO 400 J=1,3
400 SH(J)=SH(J)/S
   DEL=S/NSEG
   DO N=1,NODE
   DL=DEL*(N-1)
   X=X1+DL*SH(1)
   Y=Y1+DL*SH(2)
   Z=Z1+DL*SH(3)
   WRITE(8,*)N,X,Y,Z
   END DO
   DO NS=1,NSEG
   WRITE(8,*)NS,NS,NS+1,RWIRE
   END DO
   RETURN
   END

```

A.22 NECDAT

```

C=====
C          SUBROUTINE NECDAT(NJUN)
C-----
C THIS SUBROUTINE READ EXISTING NEC FORMATED DATA FILE

```

```

C AND TRANSLATE TO JUNCTION FORMATED INPUT DATA FILES
C
    PARAMETER(MXEDGS=1500,MXWNOD=1500)
    INTEGER NCON(2,MXEDGS),NN(3),NED(3,3)
    REAL DAT(3,MXEDGS),X(3),Y(3),Z(3)
    & ,WNODE(3,MXWNOD),SH(3)
    CHARACTER FILENAME*15,TAG(MXEDGS)*2,AA*2,NAME*15
10  FORMAT(A)
    PRINT*,'WHAT IS THE FILENAME OF NEC FORMAT DATA FILE?'
    1  CALL EOFCLR(5)
    READ(5,10,END=1)FILENAME
    OPEN(10,FILE=FILENAME,STATUS='OLD')
    REWIND(10)
    NAME='FOR002.DAT'
    4  CALL EOFCLR(5)
    3  FORMAT(A10)
    OPEN(6,FILE=NAME,TYPE='NEW')
    REWIND(6)
    IF(NJUN.LE.0)THEN
        NJCT=NJUN
        NJUN=1
    ELSE
        NJCT=1
    ENDIF
    WRITE(6,*)NJCT,NJUN
C
C READ INPUT DATA FILE
C
    DO 500 N=1,MXEDGS
500  READ(10,11,END=600)TAG(N)
    11  FORMAT(A2)
600  LINE=N
    REWIND(10)
    NODE=0
    NEDG=0
    DO 700 NL=1,LINE
C
C READ THE COORDINATES OF THE THREE VERTICES OF THE TRIANGLE
C
    IF(TAG(NL).EQ.'SP')THEN
        READ(10,112)AA,X(1),Y(1),Z(1),X(2),Y(2),Z(2)
    112  FORMAT(A2,8X,6F10.4)
        READ(10,113)AA,X(3),Y(3),Z(3)
    113  FORMAT(A2,8X,3F10.4)
C

```



```

C DISCARD DUPLICATE NODES
C
      DO 50 I=1,3
      DO 20 N=1,NODE
20     IF(DAT(1,N).EQ.X(I).AND.DAT(2,N).EQ.Y(I).
&     AND.DAT(3,N).EQ.Z(I)) GO TO 30
      NODE=NODE+1
      NN(I)=NODE
      DAT(1,NN(I))=X(I)
      DAT(2,NN(I))=Y(I)
      DAT(3,NN(I))=Z(I)
30     NN(I)= N
50     CONTINUE
C
C DISCARD DUPLICATE EDGES
C
      DO 60 I=1,3
      I1=I+1
      IF(I1.EQ.4)I1=1
      NED(1,I)=NN(I)
60     NED(2,I)=NN(I1)
      DO 70 I=1,3
      DO 80 NE=1,NEDG
80     IF((NCON(1,NE).EQ.NED(1,I).AND.NCON(2,NE).EQ.NED(2,I)).OR.
&     (NCON(1,NE).EQ.NED(2,I).AND.NCON(2,NE).EQ.NED(1,I))) GO TO 70
      NEDG=NEDG+1
      NNE=NEDG
      NCON(1,NNE)=NED(1,I)
      NCON(2,NNE)=NED(2,I)
70     CONTINUE
      ENDIF
C
C READ THE TOTAL SUBSEGMENT NUMBER OF THE WIRE
C READ THE COORDINATES OF THE START AND END POINTS OF THE WIRE
C
      IF(TAG(NL).EQ.'GW')THEN
      READ(10,800)AA,ITG,NWSEG,WNODE(1,1),WNODE(2,1),WNODE(3,1)
      $ ,WNODE(1,NWSEG+1),WNODE(2,NWSEG+1),WNODE(3,NWSEG+1),RAD
800    FORMAT(A2,I3,I5,7F10.4)
      ENDIF
700    CONTINUE
C
C TRANSFER NEC FORMAT DATA TO OUR FORMAT DATA
C
      WRITE(6,*)NODE,NEDG

```

```

      DO 100 I=1,NODE
100    WRITE(6,*)I,DAT(1,I),DAT(2,I),DAT(3,I)
      DO 200 I=1,NEDG
200    WRITE(6,*)I,NCON(1,I),NCON(2,I)
      SH(1)=WNODE(1,NWSEG+1)-WNODE(1,1)
      SH(2)=WNODE(2,NWSEG+1)-WNODE(2,1)
      SH(3)=WNODE(3,NWSEG+1)-WNODE(3,1)
      S=SQRT(SH(1)*SH(1)+SH(2)*SH(2)+SH(3)*SH(3))
      DO 400 J=1,3
400    SH(J)=SH(J)/S
      DEL=S/NWSEG
      DO 900 J=1,3
      DO 900 N=1,NWSEG
900    WNODE(J,N+1)=WNODE(J,1)+DEL*SH(J)*N
      WRITE(8,*)NWSEG+1,NWSEG
      DO 300 I=1,NWSEG+1
300    WRITE(8,*)I,WNODE(1,I),WNODE(2,I),WNODE(3,I)
      DO 1000 I=1,NWSEG
1000   WRITE(8,*)I,I,I+1,RAD
      RETURN
      END

```

A.23 BODIN

```

C=====
      SUBROUTINE BODIN(MXBDND,MXEDGS,DATNOD,NCONN,NNODES,
      $NEDGES,MXEXCI,EXCITF,NEXCIT,NJCT,MNJUN,MXWVLT,NWVLT,
      $NODVLT,CWVLT,IPAT,ITOT)
C-----
C THIS SUBROUTINE SETS CONSTANTS FOR COMMON/MEDIUM/. THEN IT READS
C TWO SETS OF INPUT DATA DEFINING THE BODY OR BODIES.
C THE FIRST SET OF DATA CONTAINS NODE NUMBERS
C AND THEIR COORDINATES. EACH NODE ALONG WITH ITS THREE COORDINATES
C IS READ AND STORED IN THE MATRIX DATNOD.
C THE SECOND SET OF DATA CONTAINS EDGE NUMBERS AND
C THE NODES TO WHICH EACH PARTICULAR EDGE IS CONNECTED. THIS
C INFORMATION IS STORED IN THE MATRIX NCONN.
C NCONN(3,EDGE)=-1 BECAUSE THE MULTIPLICITY
C FACTOR IS THE NUMBER OF ATTACHED FACES-1. LATER (IN GEOM) EACH TIME A
C FACE IS FOUND ATTACHED TO THE EDGE, NCONN(3,EDGE) WILL BE INCREMENTED.
C /GPLANE,FINDIF, AND LOOP/.
C FINALLY, IT READS THE INCIDENT FIELD PARAMETERS. THETA AND PHI
C THE DIRECTION OF PROPAGATION OF THE PLANE WAVE.

```

```

C-----
COMPLEX ETHETA,EPHI,HTHETA,HPHI,CWVLT(MXWVLT)
CHARACTER*2 ITYPE
CHARACTER*1 IG
DIMENSION DATNOD(3,MXBDND),EXCITE(7,MXEXCI)
INTEGER NCONN(3,MXEDGS),IGNDP(3),NODVLT(MXWVLT)
REAL MU,IMP
LOGICAL PRINTC,THEV
COMMON/MEDIUM/DEG2RAD,EPSLON,MU,IMP,SL,PI
COMMON/PAT/PHI1,PHI2,NPHI,THETA1,THETA2,NTHETA
COMMON/GPLANE/NGNDP,IGNDP
COMMON/FINDIF/NNFLD,DX,DY,DZ
COMMON/LOOP/PRINTC,THEV,IETHEV MLTTHV
COMMON/IGUANA/IG
C SET CONSTANT PARAMETERS FOR /MEDIUM/
PI=3.14159265358979
SL=2.997925E8
MU=PI*4.0E-07
EPSLON=1.0/(SL*SL*MU)
IMP=SQRT(MU/EPSLON)
DEG2RAD=PI/180.

C
C NJCT=0 FOR NO JUNCTION CASE, NJCT=1 OTHERWISE.
C MNJUN: NUMBER OF JUNCTIONS
  READ(2,*)NJCT,MNJUN
  WRITE(3,*)' NUMBER OF JUNCTION = ',MNJUN

C
  IF(NJCT.LE.0) MNJUN=1
  READ(2,*)NNODES,NEDGES

C
C CHECK IF DECLARED DIMENSION IS ENOUGH
C
  IF(NEDGES.GT.MXEDGS) THEN
    WRITE(6,*)'DECLARED DIMENSION FOR BODY IS INSUFFICIENT'
    STOP
  ENDIF

C
C FILL DATNOD WITH NODE LOCATIONS
C
  DO 10 I=1,NNODES
    READ(2,*) NODE,X,Y,Z
    DATNOD(1,NODE)=X
    DATNOD(2,NODE)=Y
    DATNOD(3,NODE)=Z
10  CONTINUE

```

```

C
C FILL NCONN WITH EDGE CONNECTIONS.
C
      DO 20 I=1,NEDGES
      READ(2,* ) NE,NF,NT
      NCONN(1,NE)=NF
      NCONN(2,NE)=NT
      NCONN(3,NE)=-1
20    CONTINUE
C
C TRANSFER INPUT DATA TO IGUANA FORMAT
C
      IF(IG.EQ.'Y')THEN
      WRITE(18,117)
117    FORMAT(' CM, INPUT DATA IN IGUANA FORMAT')
      WRITE(18,116)
116    FORMAT(' CE,')
      DO 119 NN=1,NEDGES
      N1=NCONN(1,NN)
      N2=NCONN(2,NN)
119    WRITE(18,128)NN,DATNOD(1,N1),DATNOD(2,N1),DATNOD(3,N1)
      $ ,DATNOD(1,N2),DATNOD(2,N2),DATNOD(3,N2)
128    FORMAT(' GW','I3','1','F9.3',' ','F9.3',' ','F9.3',' ','F9.3',' ','F9.3,
      $ ','F9.3','0.1')
118    FORMAT(' GW',I3.4X,'1',6F10.4.5X,'0')
C
      ENDIF
C
C READ GROUND PLANE PARAMETERS. NGNDP=THE NUMBER OF GROUND PLANES.
C ,OR A P.M.C. GROUND PLANE RESPECTIVELY. I=1,2,3, DENOTES GROUND PLANES
C IN THE X=0,Y=0,AND Z=0 PLANES RESPECTIVELY.
C READ IN THE NUMBER OF NODES, EDGES, AND FIELD OBSERVATION POINTS.
      PRINTC=.TRUE.
      NNFLD=0
      READ(2,*)NGNDP
      READ(2,*)IGNDP(1),IGNDP(2),IGNDP(3)
      WRITE(3,1)NGNDP
1    FORMAT(1X,'NUMBER OF IMAGE PLANES=',I3)
      WRITE(3,2)IGNDP(1),IGNDP(2),IGNDP(3)
2    FORMAT(1X,'IMAGE PLANE NOTATION:',
      $/7X,' 0=NO GROUND PLANE',
      $/7X,' 1=A P.M.C. GROUND PLANE',
      $/7X,'-1=A P.E.C. GROUND PLANE',
      $/1X,I3,' IN THE X=0 PLANE',
      $/1X,I3,' IN THE Y=0 PLANE',

```

```

    $/1X,I3,' IN THE Z=0 PLANE')
C
C WRITE INFORMATION TO TAPE3
C
    WRITE(3,19)
19  FORMAT(/20X,'VERTEX COORDINATE LIST')
    WRITE(3,21)
21  FORMAT(18X,'ALL DIMENSIONS ARE IN METERS')
    WRITE(3,22)
22  FORMAT(/1X,'VERTEX NUMBER',3X,'X-COORDINATE',2X,
    $'Y-COORDINATE Z-COORDINATE')
    DO 30 I=1,NNODES
    WRITE(3,23)I,DATNOD(1,I),DATNOD(2,I),DATNOD(3,I)
30  CONTINUE
23  FORMAT(3X,I4,10X,1E12.5,2X,1E12.5,2X,1E12.5)
C   IF FAR FIELD PATTERN OR RADAR CROSS SECTION IS DESIRED.
C   IPAT=1 IF A 3 POINT QUADRATURE IS USED
C   IPAT=2 IF A 1 POINT QUADRATURE IS USED
C   OTHERWISE IPAT=0.
    READ(2,*)IPAT
    WRITE(3,*)' IPAT = ',IPAT
    WRITE(3,*)' IF IPAT.GT.0 FAR FIELD PATTERNS ARE COMPUTED'
    IF(IPAT.GT.0)THEN
C FAR FIELD PATTERN PARAMETERS IN SPHERICAL COORDINATES:
C PHI VARIES FROM PHI1 TO PHI2 (DEGREES) WITH NPHI(>0)POINTS.
C THETA VARIES FROM THETA1 TO THETA2 (DEGREES) WITH NTHETA(>0) POINTS.
        READ(2,*)PHI1,PHI2,NPHI,THETA1,THETA2,NTHETA
        WRITE(3,*)' PATTERN PARAMETERS:'
        WRITE(3,*)' PHI1,      PHI2,      NPHI,      THETA1,
    &  THETA2,      NTHETA
        WRITE(3,8999)PHI1,PHI2,NPHI,THETA1,THETA2,NTHETA
8999  FORMAT(1X,2(2F10.4,15))
        ENDIF
        READ(2,*)ITOT
C        READ(2,*)NEXCIT
        NEXCIT=1
C READ THE INCIDENT FIELDS AND OR VOLTAGE SOURCES FOR WHICH THE-CURRENT
C DISTRIBUTIONS NEED TO BE COMPUTED.
        DO 40 I40=1,NEXCIT
        READ(2,216) ITYPE
216  FORMAT(A2)
        IF(ITYPE.EQ.' P') THEN
        EXCITE(1,I40)=0.
C EXCITE(2-7,I40)=THETA, PHI, REALHTHETA, IMAGHTHETA, REALHPHI, IMAGHPHI
        READ(2,*)(EXCITE(I,I40),I=2,7)

```

```

      HTHETA=CMPLX(EXCITE(4,I40),EXCITE(5,I40))
      HPHI=CMPLX(EXCITE(6,I40),EXCITE(7,I40))
      ETHETA=-IMP*HPHI
      EPHI=IMP*HTHETA
C      WRITE(3,217)EXCITE(2,I40),EXCITE(3,I40)
C 217   FORMAT(/5X,'ANGLE OF INCIDENCE',/,10X,'THETA=',E12.5,1X,
C      '$ DEGREES',/10X,'PHI=',E12.5,1X,'DEGREES')
C      WRITE(3,218) ETHETA,EPHI,HTHETA,HPHI
C 218   FORMAT(/5X,'POLARIZATION',/,10X,'E-THETA= (' ,2E12.5,
      '$) VOLTS/METER',
      $/,10X,'E-PHI= (' ,2E12.5,') VOLTS/METER',
      $/,10X,'H-THETA= (' ,2E12.5,') AMPS/METER',
      $/,10X,'H-PHI= (' ,2E12.5,') AMPS/METER')
      NWVLT=1
      CWVLT(1)=(0.,0.)
      ELSE
      EXCITE(1,I40)=1.
      DO II=2,7
      EXCITE(II,I40)=0.
      ENDDO
C READIN THE VOLTAGE STUFF
      READ(2,*)NWVLT

      WRITE(3,*)' NUMBER OF VOLTAGE SOURCE = ',NWVLTJ
      DO 39 N=1,NWVLT
      READ(2,*)NOD,RV,XV
      NCDVLT(N)=NOD
      CWVLT(N)=CMPLX(RV,XV)
39     WRITE(3,*)' V = ',CWVLT(N),' VOLTS ON WIRE NODE ',NOD
      ENDIF
40    CONTINUE
      RETURN
      END

```

A.24 WIRIN

```

C-----
      SUBROUTINE WIRIN(MXWNOD,MXWMLT,MXWSEG,NWNOD,NWSEG,NWUNKS,WNODE,
      $MULTW,NSEGC,WSEGH,RAD,INSEG,DATNOD,NNODES,NJCT,MNJUN,NWJUN,NBJUN)
C-----
C THIS SUBROUTINE READS
C TWO SETS OF INPUT DATA DEFINING THE WIRE OR WIRES.
* PARAMETERS PASSED INTO WIRCOM:

```

```

* MXWNOD = MAXIMUM NUMBER OF WIRE NODES.
* MXWMLT = MAXIMUM MULTIPLICITY THAT ANY WIRE NODE MAY HAVE.
* MXWSEG = MAXIMUM NUMBER OF WIRE SEGMENTS.
* MXWZN = MAXIMUM NUMBER OF LUMPED IMPEDENCES ON WIRES.
* PARAMETERS PASSED FROM WIRGOM:
* NWNOD = NUMBER OF WIRE NODES.
* NWSEG = NUMBER OF WIRE SEGMENTS.
* NWUNKS = NUMBER OF WIRE UNKNOWNNS.
* WNODE(I,J) I=1,2,3 CONTAINS THE X,Y,Z COORDINATES OF THE JTH WIRE NODE
* MULTW(N) CONTAINS THE MULTIPLICITY OF THE NTH NODE.
* NSEGC(I,J):THE JTH WIRE SEGMENT RUNS FROM NSEGC(1,J) TO NSEGC(2,J).
* WSEGH(I,J) I=1,2,3 CONTAINS THE X,Y,Z COORDINATES OF THE MIDPOINT OF
* JTH WIRE SEGMENT.
* RAD(N) CONTAINS THE RADIUS OF THE NTH WIRE SEGMENT.
* INSEG(M,N) M=1,...,MULTW(N)+1 ARE THE SEGMENTS (IN INCREASING ORDER)
* ATTACHED TO THE NTH WIRE NODE.
* NWCZN = THE NUMBER OF LUMPED IMPEDENCE LOADS ON THE WIRE.
* IWCZN(1,I) CONTAINS THE NODE LOCATION OF THE ITH IMPEDENCE LOAD,
* IWCZN(2,I)=1 IF THE LOAD IS ON THE FIRST NODE OF THE SEGMENT
*           2 IF THE LOAD IS ON THE SECOND NODE OF THE SEGMENT
* I=1,...,NWCZN.
* CWZN(I) CONTAINS THE VALUE OF THE COMPLEX LUMPED IMPEDENCE FOR THE
* IWCZN(1,I)TH NODE, I=1,...,NWCZN.

```

```

-----
C      DIMENSION WNODE(3,MXWNOD),WSEGH(3,MXWSEG),RAD(MXWSEG),
>      MULTW(MXWNOD),NSEGC(2,MXWSEG),INSEG(MXWMLT+1,MXWNOD),
>      DATNOD(3,NNODES),NWJUN(MNJUN),NBJUN(MNJUN)
C      CHARACTER*1 IG
C      COMMON/IGUANA/IG
C
C      READ(8,*)NWNOD,NWSEG
C      WRITE(9,*)'      NODES      SEGMENTS'
C      WRITE(9,*)NWNOD,NWSEG
C      WRITE(9,*)' '
C
C      CHECK IF DECLARED DIMENSION IS ENOUGH
C
C      IF(NWNOD.GT.MXWNOD.OR.NWSEG.GT.MXWSEG) THEN
C      WRITE(6,*)'DECLARED DIMENSION FOR WIRE IS INSUFFICIENT'
C      STOP
C      ENDIF
C
C      * READ WIRE NODE DATA COORDINATES
C
C      PI=3.1415926

```

```

      AK=2.*PI
      WRITE(9,*)'      NODE #      X      Y      Z
      DO 10 N=1,NWNOD
      READ(8,*)NN,WNODE(1,NN),WNODE(2,NN),WNODE(3,NN)
      WRITE(9,*)NN,WNODE(1,NN),WNODE(2,NN),WNODE(3,NN)
10  CONTINUE
      WRITE(9,*)      SEG #      FROM      TO      RADIUS
      DO 20 N=1,NWSEG
      READ(8,*) NN,NSEGC(1,NN),NSEGC(2,NN),RAD(NN)
      WRITE(9,*) NN,NSEGC(1,NN),NSEGC(2,NN),RAD(NN)
20  CONTINUE
C
C TRANSFER INPUT DATA TO NEC FORMAT
C
      IF(IG.EQ.'Y')THEN
      WRITE(12,117)
117  FORMAT('CM INPUT DATA IN NEC FORMAT')
      WRITE(12,116)
116  FORMAT('CE')
      WRITE(12,118)ITG,NWSEG,WNODE(1,1),WNODE(2,1),WNODE(3,1)
      $ ,WNODE(1,NWNOD),WNODE(2,NWNOD),WNODE(3,NWNOD),RAD(1)
C
C TRANSFER INPUT DATA TO IGUANA FORMAT
C
      WRITE(18,125)NWSEG,WNODE(1,1),WNODE(2,1),WNODE(3,1)
      $ ,WNODE(1,NWNOD),WNODE(2,NWNOD),WNODE(3,NWNOD),RAD(1)
118  FORMAT('GW',I3,I5,7F10.4)
125  FORMAT(' GW,999,',I3,',',F8.2,',',F8.2,',',F8.2,',',F8.2,',',F8.2,
      $ ',F8.2,',',F8.4)
      WRITE(18,126)
126  FORMAT(' GE,')
      WRITE(18,127)
127  FORMAT(' EN,')
C
      ENDIF
C
* COMPUTE WIRE HALF NODES
C
      WRITE(9,*)'      SEG #      CENTER POINT COORDINATES'
      DO 30 N=1,NWSEG
      NFROM=NSEGC(1,N)
      NTO=NSEGC(2,N)
      DO 25 J=1,3
      WSEGH(J,N)=(WNODE(J,NFROM)+WNODE(J,NTO))* .5
25  CONTINUE

```



```

        WRITE(9,*)N,WSEGH(1,N),WSEGH(2,N),WSEGH(3,N)
30    CONTINUE
C
C MNJUN: NUMBER OF JUNCTION
C NWJUN(I): WIRE NODE NUMBER OF THE ITH JUNCTION I=1,MNJUN
C NBJUN(I): BODY NODE NUMBER OF THE ITH JUNCTION I=1,MNJUN
C
      IF(NJCT.EQ.1)THEN
        CALL FNDJUN(WNODE,NWNOD,DATNOD,NNODES,MNJUN,NWJUN,NBJUN)
      ELSE
        MNJUN=1
        NWJUN(1)=0
        NBJUN(1)=0
      ENDIF
C
      CALL WIRMUL(NJCT,MXWMLT,NWNOD,NWSEG,NSEGC,NWUNKS,MULTW,
&INSEG,MNJUN,NWJUN)
      RETURN
      END

```

A.25 FACMUL

```

-----
      SUBROUTINE FACMUL(NCONN,NEDGES,ITRAK,
$                   NBOUND,MXFACE,NFACES,NUNKNB)
-----
C THIS SUBROUTINE FILLS NBOUND WITH THE FACES FORMED BY THE EDGES
C IN NCONN. IT ALSO FILLS IN THE MULTIPLICITY FACTOR OF THE EDGE
C IN NCONN(3,EDGE). IT RETURNS THE NUMBER OF FACES(NFACES),
C AND THE NUMBER OF BODY UNKNOWNNS(NUNKNB) WHICH IS EQUAL TO THE
C SUMMATION OF THE MULTIPLICITY FACTORS OF THE EDGES BEFORE GROUND
C PLANES ARE CONSIDERED. ITRAK IS A WORK ARRAY.
-----
      INTEGER NCONN(3,NEDGES),NBOUND(3,MXFACE),ITRAK(NEDGES)
      NFACES=0
      DO 100 IEDGE=1,NEDGES-2
        NTRAK=0
        DO 200 JEDGE=IEDGE+1,NEDGES
          DO 20 I=1,2
            DO 21 J=1,2
              IF(NCONN(I,IEDGE).EQ.NCONN(J,JEDGE))THEN
                NTRAK=NTRAK+1
                ITRAK(NTRAK)=JEDGE
              ENDIF
            END DO
          END DO
        END DO
      END DO

```

```

                GOTO 200
            ENDIF
21          CONTINUE
20          CONTINUE
200         CONTINUE
           DO 300 JEDGE=1,NTRAK-1
           DO 301 KEDGE=JEDGE+1,NTRAK
           DO 30 J=1,2
           DO 31 K=1,2
           IF((NCONN(J,ITRAK(JEDGE)).EQ.NCONN(K,ITRAK(KEDGE))) .AND.
$          (NCONN(J,ITRAK(JEDGE)).NE.NCONN(1,IEDGE) .AND. NCONN(J,ITRAK
$          (JEDGE)).NE.NCONN(2,IEDGE))) THEN
           NFACES=NFACES+1
           NBOUND(1,NFACES)=IEDGE
           NBOUND(2,NFACES)=ITRAK(JEDGE)
           NBOUND(3,NFACES)=ITRAK(KEDGE)
           NCONN(3,IEDGE)=NCONN(3,IEDGE)+1
           NCONN(3,ITRAK(JEDGE))=NCONN(3,ITRAK(JEDGE))+1
           NCONN(3,ITRAK(KEDGE))=NCONN(3,ITRAK(KEDGE))+1
           GOTO 300
           ENDIF
31          CONTINUE
30          CONTINUE
301         CONTINUE
300         CONTINUE
100        CONTINUE
           MUNKNB=3*NFACES-NEDGES
           RETURN
           END

```

A.26 ORTFAC

```

C=====
C          SUBROUTINE ORTFAC(NCONN,NBOUND,NFACES,NEDGES,MXDJBD,ITREE,ITART,
$          NBOBYS,NBE)
C-----
C  ORIENT FACE NUMBER
C  ALL EDGES IN THE FIRST DISJOINT SURFACE ARE NUMBERED
C  CONSECUTIVELY STARTING FROM 1. THE EDGES IN THE NEXT DISJOINT
C  SURFACE ARE NUMBERED CONSECUTIVELY, STARTING WHERE THE LAST SURFACE
C  LEFT OFF.
C
C INPUT

```

AD-A200 315

JUNCTION CODE USER'S MANUAL ELECTROMAGNETIC SCATTERING

2/3

AND RADIATION BY G. (U) HOUSTON UNIV ELECTROMAGNETICS

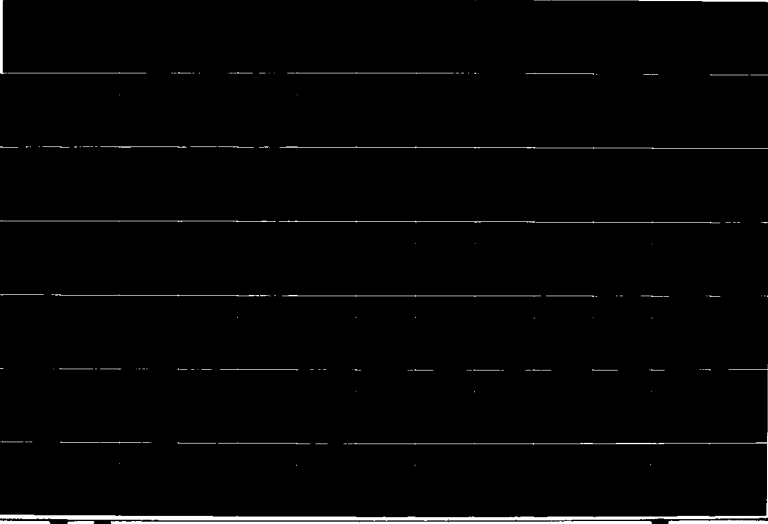
LAB D B MILTON ET AL. AUG 88 TR-87-18 NOSC-10-1214

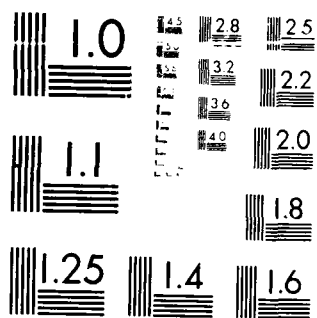
UNCLASSIFIED

NS6001-85-D-0203

P/G 28/14

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

```

C
C NCONN(3,NEDGES): EDGE J RUNS FROM VERTEX NCONN(1,J)
C TO VERTEX NCONN(2,J) NCONN(3,J)=MULTIPLICITY FACTOR OF THE EDGE
C NBOUND(3,NFACES):EACH FACE J HAS ORDERED EDGES NBOUND(I,J) I=1,2,3
C J=1,2,...,NFACES.
C NFACES EQUALS THE TOTAL NUMBER OF FACES.
C NEDGES EQUALS THE TOTAL NUMBER OF EDGES.
C MXDJBD EQUALS THE MAXIMUM NUMBER OF EXPECTED BODYS.
C
C OUTPUT:
C
C ISTART(MXDJBD+1):ISTART(I)=THE LOWEST NUMBERED FACE ON THE ITH
C TREE(DISJOINT SURFACE)
C ISTART(NBODYS+1)=NFACES+1
C MXDJBD.GE.NBODYS OR ROUTINE STOPS AND PRINTS A WARNING.
C ITREE(NFACES)
C ITREE(I) I=1,...,ISTART(2)-1 =THE FACES ON THE FIRST TREE.
C ITREE(I) I=ISTART(J),...,ISTART(J+1)-1,=THE FACES ON THE JTH TREE.
C NBODYS EQUALS THE NUMBER OF DISJOINT SURFACES.
C NBE(MXDJBD):NBE(I) CONTAINS THE NUMBER OF BOUNDARY EDGES FOR BODY I.
C
C FOR EACH FACE,
C IFACE=1,...,NFACES, THE EDGES IN NBOUND(IFACE,II),II=1,2,3
C ARE REARRANGED SO THAT THEIR ORIENTATION IS CONSISTENT WITH
C THE LOWEST NUMBERED FACE IN THE TREE CONTAINING IFACE.
C TWO FACES WITH A COMMON EDGE ARE ORIENTED CONSISTENTLY WHEN THE
C EDGE CROSS THE NORMAL TO THAT FACE POINTS OUT OF THAT FACE ACROSS THE
C COMMON EDGE WHILE THE COMMON EDGE CROSS THE NORMAL OF THE CONTIGUOUS
C FACE POINTS INTO THE CONTIGUOUS FACE ACROSS THE COMMON EDGE.
C FOR CONVENIENCE ONE MAY CYCLICALLY PERMUTE EDGES 1,2,3 TO 2,3,1 OR
C 3,1,2 WITHOUT CHANGING THE FACE ORIENTATION.
C-----
C INTEGER NCONN(3,NEDGES),NBOUND(3,NFACES),ITREE(NFACES),
C $ ISTART(MXDJBD+1),NBE(MXDJBD)
C NFACE1=NFACES+1
C NTREE=1
C LNF=1
C ITREE(LNF)=LNF
C ISTART(1)=LNF
C DO 1 NBODYS=1,MXDJBD
C DO 40 I40=1,3
C IF(NCONN(3,NBOUND(I40,LNF)).EQ.0)NBE(NBODYS)=NBE(NBODYS)+1
40 CONTINUE
51 DO 50 IFACE=LNF+1,NFACES
C DO 10 JTREE=LNF,NTREE

```

```

        IF(IFACE.EQ.ITREE(JTREE))GOTO 50
10      CONTINUE
        LOWFACE=NFACE1
        DO 20 JTREE=LNF,NTREE
          DO 30 I=1,3
            DO 31 J=1,3
              IF(NBOUND(I,IFACE).EQ.NBOUND(J,ITREE(JTREE)).AND.
                $ ITREE(JTREE).LT.LOWFACE)LOWFACE=ITREE(JTREE)
31          CONTINUE
30          CONTINUE
20          CONTINUE
          IF(LOWFACE.NE.NFACE1)THEN
            DO 60 I=1,3
              DO 61 J=1,3
                IF(NBOUND(I,IFACE).EQ.NBOUND(J,LOWFACE))THEN
                  NTREE=NTREE+1
                  ITREE(NTREE)=IFACE
                  DO 41 I41=1,3
                    IF(NCONN(3,NBOUND(I41,IFACE)).EQ.0)
                      $ NBE(NBODYS)=NBE(NBODYS)+1
41          CONTINUE
                  ISLOT=I+1
                  JSLOT=J+1
                  IF(ISLOT.EQ.4)ISLOT=1
                  IF(JSLOT.EQ.4)JSLOT=1
                  IF(NCONN(1,NBOUND(ISLOT,IFACE)).EQ.NCONN(1,NBOUND(JSLOT,
                    $LOWFACE)).OR.NCONN(1,NBOUND(ISLOT,IFACE)).EQ.NCONN(2,NBOUND(JSLOT,
                    $LOWFACE)).OR.NCONN(2,NBOUND(ISLOT,IFACE)).EQ.NCONN(1,NBOUND(JSLOT,
                    $LOWFACE)).OR.NCONN(2,NBOUND(ISLOT,IFACE)).EQ.NCONN(2,NBOUND(JSLOT,
                    $LOWFACE)))THEN
                    IDUM=NBOUND(1,IFACE)
                    NBOUND(1,IFACE)=NBOUND(2,IFACE)
                    NBOUND(2,IFACE)=IDUM
                    ENDIF
                    GOTO 51
                  ENDIF
61          CONTINUE
60          CONTINUE
          ENDIF
50          CONTINUE
          LNF=NTREE+1
          IF(LNF.LE.NFACES)THEN
            ISTART(NBODYS+1)=LNF
            NTREE=NTREE+1
            ITREE(NTREE)=LNF

```

```

        ELSE
          GOTO 999
        ENDIF
1      CONTINUE
      WRITE(1,99)
99     FORMAT(1X,'WARNING IN CURDIR MXDJBD FOUND BUT STILL HAVE FACES
$LEFT')
      STOP
999   CONTINUE
      ISTART(NBODYS+1)=NFACES+1
      RETURN
      END

```

A.27 CLSBOD

```

C=====
      SUBROUTINE CLSBOD(DATNOD,NCONN,NBOUND,NNODES,NEDGES,NFACES,I,
$          ISTART)
C-----
C THIS SUBROUTINE IS CALLED ONLY IF THE BODY IS CLOSED. IN THIS
C CASE, THIS SUBROUTINE ORIENTS THE NORMAL POINTING INTO THE
C SURROUNDING MEDIUM. THE VOLUME OF THE BODY IS ALSO CALCULATED.
C THE VOLUME IS COMPUTED BY USE OF THE IDENTITY:
C VOLUME = THE INTEGRAL OVER THE SURFACE BOUNDING THE VOLUME OF THE X
C COMPONENT OF THE SURFACE NORMAL TIMES X TIMES THE
C DIFFERENTIAL SURFACE AREA.
C-----
      DIMENSION DATNOD(3,NNODES),J(3),NJ(3),DJ(3,3)
      INTEGER NCONN(3,NEDGES),NBOUND(3,NFACES),ISTART(NFACES+1)
      COMMON/VOL/VOLUME
      VOLUME=0.0
      DO 15 IJ= ISTART(I),ISTART(I+1)-1
        CALL FACEDG(NFACES,NBOUND,IJ,J)
        CALL FACVTX(NCONN,NEDGES,J,NJ)
        X=(DATNOD(1,NJ(1))+DATNOD(1,NJ(2))+DATNOD(1,NJ(3)))/3.0
        CALL VTXCRD(DATNOD,NNODES,NJ,DJ)
        ARJ1=(DJ(2,2)-DJ(1,2))*(DJ(3,3)-DJ(1,3))-(DJ(3,2)-DJ(1,2))*
$         (DJ(2,3)-DJ(1,3))
        VOLUME=VOLUME+X*ARJ1/2.0
15     CONTINUE
      IF(VOLUME LT.0.0) THEN
        DO 16 I16= ISTART(I),ISTART(I+1)-1
          IDUMMY=NBOUND(1,I16)

```

```

        NBOUND(1,I16)=NBOUND(2,I16)
        NBOUND(2,I16)=IDUMMY
16      CONTINUE
        ENDIF
        RETURN
        END

```

A.28 FACOUT

```

C=====
C      SUBROUTINE FACOUT(NCONN,NBOUND,ISTART,I,NEDGES,NFACES,
C      $                NBODYS,DATNOD,NNODES)
C-----
C THIS SUBROUTINE PRINTS THE EDGES AND THE VERTICES OF EACH FACE.
C INPUT:
C NCONN HAS THE VERTICIES AND THE MULTIPLICITY FACTOR FOR EACH EDGE.
C NBOUND HAS THE EDGES FOR EACH FACE.
C ISTART HAS THE BEGINNING FACES FOR EACH BODY.
C I IS THE PRESENT BODY.
C NEDGES IS THE TOTAL NUMBER OF EDGES.
C NFACES IS THE TOTAL NUMBER OF FACES.
C NBODYS IS THE TOTAL NUMBER OF BODYS.
C-----
C      INTEGER NCONN(3,NEDGES),NBOUND(3,NFACES),ISTART(NBODYS+1),NV(3)
C      $                ,NB(3)
C      REAL DATNOD(3,NNODES),DN(3,3)
C      CHARACTER*1 IG
C      COMMON/IGUANA/IG
C      WRITE(3,102)I
102  FORMAT(/20X,'FOR BODY NUMBER:',I3/)
C      DO 10 I10=ISTART(I),ISTART(I+1)-1
C OBTAIN THE EDGES OF THE TRIANGLE
C      DO 2 I=1,3
2      NB(I)=NBOUND(I,I10)
C OBTAIN THE VERTICES OF THE TRIANGLE
C      CALL FACVTX(NCONN,NEDGES,NB,NV)
C      WRITE(3,98)I10,NBOUND(1,I10),NBOUND(2,I10),NBOUND(3,I10),
C      $ NV(1),NV(2),NV(3)
C
C TRANSFER INPUT DATA TO NEC FORMAT
C
C      IF(IG.EQ.'Y')THEN
C OBTAIN THE COORDINATES OF THE TRIANGLE'S VERTICIES

```



```

          CALL VTXCRD(DATNOD,NNODES,NV, DN)
58      WRITE(12,112)DN(1,1),DN(1,2),DN(1,3),DN(2,1),
      $   DN(2,2),DN(2,3)
          WRITE(12,113)DN(3,1),DN(3,2),DN(3,3)
          ENDIF
112     FORMAT('SP',7X,'2',6F10.4)
113     FORMAT('SC',7X,'2',3F10.4)
C
10      CONTINUE
98      FORMAT(1X,'FACE',I4,' HAS EDGES',3I4,' WITH VERTICES',3I4)
C TRANSFER INPUT DATA TO NEC FORMAT
C
          IF(IG.EQ.'Y')THEN
          WRITE(12,116)
116     FORMAT('GE')
          WRITE(12,117)
117     FORMAT('EN')
          ENDIF
          RETURN
          END

```

A.29 ADBMUL

```

C=====
          SUBROUTINE ADBMUL(NNODES,NEDGES,DATNOD,NCONN,NUNKNB)
C-----
C ADJUST MULTIPLICITY OF BODY EDGE
C INPUT:
C NNODES=THE NUMBER OF BODY NODES.
C NEDGES=THE NUMBER OF EDGES.
C NUNKNB=THE NUMBER OF BODY UNKNOWNNS BEFORE CONSIDERING THE
C   GROUND PLANE ATTACHMENTS.
C DATNOD(I,N)=THE X,Y,Z COMPONENTS(I=1,2,3) OF THE NTH NODE N=1,NNODES.
C NCONN(I,IE) I=1,2,3: EDGE IE (IE=1,NEDGES) RUNS FROM NODE NCONN(1,IE)
C   TO NCONN(2,IE) AND HAS MULTIPLICITY NCONN(3,IE)(BEFORE ANY GROUND
C   PLANE ATTACHMENTS ARE CONSIDERED).
C
C OUTPUT:
C FOR EACH EDGE IE THAT IS CONNECTED TO A P.E.C. GROUND PLANE AND IS
C NOT CONNECTED TO A P.M.C. GROUND PLANE, ITS MULTIPLICITY(NCONN(IE,3))
C AND THE NUMBER OF BODY UNKNOWNNS(NUNKNB) ARE INCRIMENTED BY 1.
C THE EDGE VERTEX CONNECTION LIST WITH EDGE MULTIPLICITIES IS OUTPUTTED
C AFTER ACCOUNTING FOR ALL GROUND PLANE ATTACHMENTS.

```

```

C-----
  DIMENSION DATNOD(3,NNODES),NCONN(3,NEDGES),IGNDP(3),D1(3),D2(3),
$          DM(3)
  COMMON/GPLANE/NGNDP,IGNDP
  IF(NGNDP.GT.0)THEN
    EDGED=1.E-4
    DO 100 IE=1,NEDGES
      N1=NCONN(1,IE)
      N2=NCONN(2,IE)
    DO 2 I=1,3
      D1(I)=ABS(DATNOD(I,N1))
      D2(I)=ABS(DATNOD(I,N2))
2     DM(I)=AMAX1(D1(I),D2(I))
      IX=0
      IY=0
      IZ=0
      IF(DM(1).LE.EDGED)IX=IGNDP(1)
      IF(IX.NE.1)THEN
        IF(DM(2).LE.EDGED)IY=IGNDP(2)
        IF(IY.NE.1.AND.DM(3).LE.EDGED)IZ=IGNDP(3)
      ENDIF
      IMAX=AMAXO(IX,IY,IZ)
      IF(IMAX.NE.1)THEN
        IMIN=AMINO(IX,IY,IZ)
        NUNKNB=NUNKNB-IMIN
        NCONN(3,IE)=NCONN(3,IE)-IMIN
      ENDIF
100    CONTINUE
    ENDIF
    WRITE(3,29)
29    FORMAT(/14X,'EDGE-VERTEX CONNECTION LIST'/)
    DO 40 I=1,NEDGES
      WRITE(3,331)I,NCONN(1,I),NCONN(2,I),NCONN(3,I)
40    CONTINUE
331   FORMAT(3X,'EDGE',I4,' GOES FROM VERTEX',I4,' TO VERTEX',I4,
$ ' MULT=',I2)
    RETURN
  END

```

A.30 BODPAR

```

C=====
  SUBROUTINE BODPAR(DATNOD,NCONN,NBOUND,NNODES,NEDGES,NFACES,NUNKNB)

```

```

C-----
C COMPUTE PARAMETERS ASSOCIATED WITH BODY
C INPUT:
C DATNOD(I,J) I=1,2,3 ARE THE X,Y,Z COORDINATES OF THE JTH NODE.
C NCONN(3,J): EDGE J RUNS FROM NODE NCONN(1,J) TO NODE NCONN(2,J)
C           NCONN(3,J) IS THE MULTIPLICITY OF THE JTH EDGE.
C NBOUND(I,J): CONTAINS THE ITH EDGE OF THE JTH FACE I=1,2,3.
C NNODES = THE NUMBER OF BODY NODES.
C NEDGES = THE NUMBER OF EDGES.
C NFACES = THE NUMBER OF FACES.
C NUNKNB = THE NUMBER OF BODY UNKNOWNNS.
C
C OUTPUT:
C AVERAGE = THE AVERAGE EDGE LENGTH(METERS**2) INCLUDING MULTIPLICITY.
C EDGEMX = THE MAXIMUM EDGE LENGTH(METERS).
C MXEDGE = THE EDGE NUMBER OF THE EDGE WITH LENGTH EDGEMX.
C EDGEMN = THE MINIMUM EDGE LENGTH(METERS).
C MNEDGE = THE EDGE NUMBER OF THE EDGE WITH LENGTH EDGEMN.
C TAREA = THE SURFACE AREA OF THE SCATTER(METERS**2):FOR THIN
C         STRUCTURES ONLY ONE SIDE IS CONSIDERED IN THE SURFACE AREA.
C AVAREA = THE AVERAGE AREA OF THE FACES.
C MXAREA = THE NUMBER OF THE FACE WITH THE MAXIMUM AREA(AREAMX).
C MNAREA = THE NUMBER OF THE FACE WITH THE MINIMUM AREA(AREAMN).
C RATIO = THE MINIMUM HEIGHT TO BASE RATIO OVER ALL FACES.
C MNRTIO = THE FACE NUMBER THAT HAS A HEIGHT TO BASE RATIO OF "RATIO".
C-----
COMMON/PARAMS/AVEDGE,EDGEMX,MXEDGE,EDGEMN,MNEDGE,TAREA,AVAREA,
$           MXAREA,MNAREA,AREAMX,AREAMN,RATIO,MNRTIO
COMMON/MCHVAL/VALMAX,VALMIN
DIMENSION DATNOD(3,NNODES),DL(3,3),AL(3),RS(3),HTB(3),DV(3),DX(3),
$           DN(3,3)
INTEGER NCONN(3,NEDGES),NBOUND(3,NFACES),IS(3),NV(3)
SIZE(X,Y,Z)=SQRT(X*X+Y*Y+Z*Z)
SEDGL=0
EDGEMX=VALMIN
EDGEMN=VALMAX
DO 20 IE=1,NEDGES
  MULT=NCONN(3,IE)
  N1=NCONN(1,IE)
  N2=NCONN(2,IE)
  DO 2 I=1,3
2    DX(I)=DATNOD(I,N2)-DATNOD(I,N1)
    EDGL=SIZE(DX(1),DX(2),DX(3))
    SEDGL=SEDGL+MULT*EDGL
    IF(EDGL.GT.EDGEMX)THEN

```

```

        EDGEMX=EDGL
        MXEDGE=IE
    ENDIF
    IF(EDGL.LT.EDGEMN)THEN
        EDGEMN=EDGL
        MNEDGE=IE
    ENDIF
20  CONTINUE
    AVEDGE=SEDGL/NUNKNB
    RATIO=VALMAX
    AREAMX=VALMIN
    AREAMN=VALMAX
    TAREA=0.
    DO 40 IFACE=1,NFACES
        DO 4 I=1,3
            IS(I)=NBOUND(I,IFACE)
            CALL FACVTX(NCONN,NEDGES,IS,NV)
            CALL VTXCRD(DATNOD,NNODES,NV,DN)
            DO 6 I=1,3
                IP1=MOD(I,3)+1
                IM1=MOD(I+1,3)+1
                DO 6 J=1,3
                    DL(I,J)=DN(IM1,J)-DN(IP1,J)
                    DO 8 J=1,3
                        JP1=MOD(J,3)+1
                        JM1=MOD(J+1,3)+1
                        DV(J)=DL(2,JP1)*DL(1,JM1)-DL(2,JM1)*DL(1,JP1)
                        AREA2=SIZE(DV(1),DV(2),DV(3))
                        AREA=.5*AREA2
                        DO 3 I=1,3
                            AL(I)=SIZE(DL(I,1),DL(I,2),DL(I,3))
                            RS(I)=AL(I)*AL(I)
                        3  HTB(I)=AREA2/RS(I)
                            HTBMIN=AMIN1(HTB(1),HTB(2),HTB(3))
                            TAREA=TAREA+AREA
                            IF(AREA.GT.AREAMX)THEN
                                MXAREA=IFACE
                                AREAMX=AREA
                            ENDIF
                            IF(AREA.LT.AREAMN)THEN
                                MNAREA=IFACE
                                AREAMN=AREA
                            ENDIF
                            IF(HTBMIN.IT.RATIO)THEN
                                MNRTIO=IFACE

```

```

        RATIO=HTBMIN
    ENDIF
40  CONTINUE
    AVAREA=TAREA/NFACES
    WRITE(3,110)
110  FORMAT(/25X,'BODY PARAMETER LIST'/)
    WRITE(3,111) NNODES,NEDGES,NFACES,NUNKNB
111  FORMAT(10X,'NUMBER OF VERTICES= ',I4,
    $/10X,'NUMBER OF EDGES= ',I4,
    $/10X,'NUMBER OF FACES= ',I4,
    $/10X,'NUMBER OF EDGES INCLUDING MULTIPLICITY= ',I4)
    WRITE(3,205)
205  FORMAT(/25X,'MODELING PARAMETER LIST (METERS)'/)
    WRITE(3,206) TAREA
206  FORMAT(10X,'SURFACE AREA OF THE SCATTERER=',E12.5,1X,'SQ.METERS')
    WRITE(3,209) AVEDGE,MXEDGE,EDGEMX,MNEDGE,EDGEMN
209  FORMAT(10X,'AVERAGE EDGE LENGTH=',E12.5,1X,'METERS',
    $/10X,'MAXIMUM EDGE LENGTH(EDGE NO. ',I3,1X,')=',E12.5,1X,'METERS',
    $/10X,'MINIMUM EDGE LENGTH(EDGE NO. ',I3,1X,')=',E12.5,1X,'METERS')
    WRITE(3,210) AVAREA,MXAREA,AREAMX,MNAREA,AREAMN
210  FORMAT(10X,'AVERAGE FACE AREA =',E12.5,1X,'SQ.METERS',/10X,
    $'MAXIMUM FACE AREA (FACE NO. ',I4,1X,')=',E12.5,1X,'SQ.METERS',/
    $10X,'MINIMUM FACE AREA (FACE NO. ',I4,1X,')=',E12.5,1X,'SQ.METERS')
    WRITE(3,211) MNRTIO,RATIO
211  FORMAT(10X,'MINIMUM FACE HEIGHT TO BASE RATIO (FACE NO. ',
    $I4,1X,')=',E11.5)
    RETURN
    END

```

A.31 EDGFAC

```

C=====
C          SUBROUTINE EDGFAC(NCONN,NEDGES,NBOUND,NFACES,IEDGF,MULT1)
C-----
C MAPPING FROM EDGE TO FACE
C INPUT:
C  EDGE IE RUNS FROM VERTEX NCONN(1,IE) TO VERTEX NCONN(2,IE)
C  AND HAS MULTIPLICITY NCONN(3,IE).
C  FACE IFACE HAS EDGES NBOUND(J,IFACE) J=1 2,3
C  MULT1 IS SET IN THE MAIN PROGRAM AND MULT1-1.GE.THE
C  MAXIMUM MULTIPLICITY OF ANY EDGE.
C OUTPUT:
C  ARRAY IEDGF

```

```

C FOR AN EDGE WITH MULTIPLICITY MULT
C IEDGF(2,IE)=THE NEXT LOWEST NUMBERED FACE CONNECTED TO IE.
C .....
C IEGDF(MM,IE)=THE LAST FACE CONNECTED TO IE.
C WHERE MM IS THE NUMBER OF FACES CONNECTED TO EDGE IE.
C-----
      INTEGER NCONN(3,NEDGES),NBOUND(3,NFACES),IEDGF(MULT1,NEDGES)
      DO 5 IE=1,NEDGES
        DO 6 M=1,MULT1
          IEDGF(M,IE)=0
6      CONTINUE
5      CONTINUE
      DO 100 IE=1,NEDGES
        MULTE1=NCONN(3,IE)+1
        M=0
        DO 50 IF=1,NFACES
          IF(M.GE.MULTE1)GO TO 100
          IF(IE.EQ.NBOUND(1,IF).OR.IE.EQ.NBOUND(2,IF).OR.IE.EQ.
$ NBOUND(3,IF))THEN
            M=M+1
            IEDGF(M,IE)=IF
          ENDIF
50      CONTINUE
100     CONTINUE
      DO 200 IE=1,NEDGES
        WRITE(3,201)IE
        WRITE(3,*)(IEDGF(M,IE),M=1,NCONN(3,IE)+1)
200     CONTINUE
201     FORMAT(1X,'EDGE',14,' IS ATTACHED TO FACES')
      RETURN
      END

```

A.32 SOLTN

```

C=====
      SUBROUTINE SOLTN(CZ,CV,NUNKNT,IPVT,CWORK,DATNOD,NCONN,NBOUND,
$ NEDGES,NFACES,NNODES,IEDGF,MULT1,EXCITE,NEXCIT,
$ MXFREQ,INDSUM,NUNKNB,NWNOD,NWSEG,NWUNKS,WNODE,MULTW,NSEGC,
& WSEGH,RAD,INSEG,NJCT,MNJUN,NWJUN,NBJUN,
&WIRSUM,ANG,MNJFACE,NJFACE,MIFACE,MXWMLT,NWWVLT,NODVLT,CWWVLT,
&IPAT,ITOT)
C-----
C IN THIS SUBROUTINE, THE MATRIX EQUATION ZI=V IS SOLVED.

```

```

      IMPLICIT COMPLEX (C)
C
      REAL ANG(MNJUN,MNJFACE)
      INTEGER NJFACE(MNJUN,MNJFACE),MIFACE(MNJUN),NBJUN(MNJUN),
&  NWJUN(MNJUN)
C
      DIMENSION WNODE(3,NWNOD),WSEGH(3,NWSEG),RAD(NWSEG)
      INTEGER MULTW(NWNOD),NSEGC(2,NWSEG),WIRSUM(NWNOD),
>  INSEG(MXWMLT+1,NWNOD)
      COMPLEX CZ(NUNKNT,NUNKNT),CV(NUNKNT),CWORK(NUNKNT)
      COMPLEX HTHETA,HPHI,ETHETA,EPHI
      REAL LAMBDA,K,MU,IMP
      DIMENSION DATNOD(3,NNODES),EXCITE(7,NEXCIT)
      INTEGER NCONN(3,NEDGES),NBOUND(3,NFACES),IEDGF(MULT1,NEDGES),
$IPVT(NUNKNT),INDSUM(NEDGES)
C FOR WIRE
      INTEGER NODVLT(NWWVLT)
      COMPLEX CWWVLT(NWWVLT)
      LOGICAL PRINTC
      CHARACTER*1 IC
C
      COMMON/LOOP/PRINTC,THEV,IETHEV,MLTTHV
      COMMON/PARAM/THETA,PHI,IFIELD
      COMMON/WAVE/OMEGA,LAMBDA,K
      COMMON/MEDIUM/DEG2RAD,EPSLON,MU,IMP,SL,PI
      COMMON/INC/HTHETA,HPHI,ETHETA,EPHI
      COMMON/FINDIF/NNFLD,DX,DY,DZ
      COMMON/F/FREQ
      COMMON/CPU/TGG,IC
      DO 20 IFREQ=1,MXFREQ
C READ IN THE FREQUENCY IN HERTZ.
      READ( 2,*) FREQ
      IF(FREQ.EQ.-1.)STOP
      WRITE(3,999)FREQ
      WRITE(4,999)FREQ
999  FORMAT(1X,'FREQ=',1PE12.5)
C LAMBDA=THE WAVELENGTH.
C K=THE WAVE NUMBER.
C OMEGA=THE ANGULAR FREQUENCY.
      OMEGA=2.*PI*FREQ
      K=OMEGA/SL
      LAMBDA=SL/FREQ
      IFIELD=0
      DO 11 J=1,NUNKNT
        DO 10 I=1,NUNKNT

```

```

          CZ(I,J)=(0.0,0.0)
10      CONTINUE
11      CONTINUE
          DO 40 I40=1,NEXCIT
              IFIELD=IFIELD+1
C INITIALIZE THE VOLTAGE VECTORS.
          DO 14 I=1,NUNKNT
              CV(I)=(0.0,0.0)
14      CONTINUE
          IF(EXCITE(1,NEXCIT).EQ.0.) THEN
C IF EXCITATION IS A PLANE WAVE
C HTHETA AND HPHI REPRESENT THE AMPLITUDE OF THE INCIDENT PLANE WAVE.
          HTHETA=CMPLX(EXCITE(4,I40),EXCITE(5,I40))
          HPHI=CMPLX(EXCITE(6,I40),EXCITE(7,I40))
          ETHETA=-IMP*HPHI
          EPHI=IMP*HTHETA
          THETA=EXCITE(2,I40)*DEG2RAD
          PHI=EXCITE(3,I40)*DEG2RAD
          ELSE
C IF EXCITATION IS A VOLTAGE SOURCE
          IF(NJCT.GE.0) THEN
              CALL WCUMUL(NWNOB,NUNKNB,MULTW,WIRSUM)
              DO 144 IV=1,NWWVLT
                  NODV=NDDVLT(IV)
                  NROW=WIRSUM(NODV)+1
144             CV(NROW)=CWWVLT(IV)
              ENDIF
          ENDIF
C
C INPUT:
C MNJUN: NUMBER OF JUNCTION
C NWJUN(I): WIRE NODE NUMBER OF THE ITH JUNCTION I=1,MNJUN
C NBUN(I): BODY NODE NUMBER OF THE ITH JUNCTION I=1,MNJUN
C MNJFACE: MAXIMUM NUMBER OF FACE ATTACHED TO JUNCTION
C OUTPUT:
C NJFACE(I,J): FACE NUMBER OF THE JTH FACE ATTACHED TO THE ITH JUNCTION
C I=1,MNJUN, J=1,MJFACE(I)
C MJFACE(I): MAXIMUM NUMBER OF FACE ATTACHED TO THE ITH JUNCTION
C ANG(I,J): ANGLE FACTOR OF THE JTH PATCH ATTACHED TO THE ITH JUNCTION
C
          IF(NJCT.EQ.1) THEN
C
C COMPUTE PARAMETERS ASSOCIATE WITH JUNCTIONS
C
          CALL JUNFAC(NBJUN,MNJUN,NBOUND,NFACES,NCONN,NEDGES,

```



```

& NJFACE,MNJFACE,MIFACE)
CALL JANGLE(DATNOD,NNODES,NBJUN,MNJUN,NBOUND,NFACES,NCONN,
& NEDGES,NJFACE,MNJFACE,MIFACE,ANG)
ENDIF
C
CALL BCUMUL(NCONN,INDSUM,NEDGES)
IF(NJCT.GE.0) THEN
C
C FILL ZWW, ZBW, AND ZJW OF THE IMPEDANCE MATRIX
C
CALL WIRMAN(IFREQ,CZ,CV,NUNKNT,NUNKNB,NWNOD,NWSEG,NWUNKS,
& WNODE,MULTW,NSEGC,WSEGH,RAD,INSEG,WIRSUM,DATNOD,
& NCONN,NBOUND,NNODES,NEDGES,NFACES,IEDGF,MULT1,INDSUM,NJCT,MNJUN,
& NBJUN,NBJUN,NJFACE,MNJFACE,MIFACE,ANG,MXWMLT,IPAT)
ENDIF
C
C FILL ZBB OF THE IMPEDANCE MATRIX
C
CALL ZBB(DATNOD,NCONN,NBOUND,IEDGF,MULT1,NNODES,NEDGES,
$ NFACES,NUNKNT,CZ,CV,INDSUM,NJCT,MNJUN,MNJFACE,NWNOD,WIRSUM,
& ANG,NJFACE,MIFACE,NWJUN,NBJUN)
C
C FILL ZWB OF THE IMPEDANCE MATRIX
C
IF(NJCT.GE.0) THEN
CALL ZWB(DATNOD,NCONN,NBOUND,IEDGF,MULT1,NNODES,NEDGES,
$ NFACES,NUNKNT,CZ,INDSUM,NWNOD,NWSEG,WNODE,NSEGC,WSEGH,
$ MXWMLT,MULTW,INSEG,WIRSUM,NUNKNB,NJCT,MNJUN,MNJFACE,ANG,
& NJFACE,MIFACE,NWJUN,NBJUN)
ENDIF
C
WRITE(6,*)' '
WRITE(6,*)'*** MATRIX FILLING COMPLETED ***'
C
IF(IC.EQ.'Y')THEN
C
C CALL RUNTIME LIBRARY
C
IERR=LIB$STAT_TIMER( 2, ITG1. )
TG2=FLOAT(ITG1)/6000.
TGF=TG2-TGG
C
WRITE(6,*)'CPUTIME FOR MATRIX FILLING = ',TGF,' MINUTES'
ENDIF
C

```

```

        WRITE(6,*)' '
        WRITE(6,*)' EXECUTION ..... '
C
C CALL ROUTINES TO SOLVE MATRIX EQUATION
C CGEFA AND CGESL ARE SUBROUTINES IN LINPACK LIBRARY
C
        IF(IFIELD.EQ.1)CALL CGEFA(CZ,NUNKNT,NUNKNT,IPVT,INFO)
        IF(INFO.EQ.0) THEN
            CALL CGESL(CZ,NUNKNT,NUNKNT,IPVT,CV,0)
        ELSE
            WRITE(1,8)
8          FORMAT(23H THE MATRIX IS SINGULAR)
            STOP
        ENDIF
777 CONTINUE
        WRITE(6,*)' '
        WRITE(6,*)'*** MATRIX EQUATION SOLVED ***'
        WRITE(6,*)' '
C
        IF(IC.EQ.'Y')THEN
C
C CALL RUNTIME LIBRARY
C
        IERR=LIB$STAT_TIMER( 2, ITG2, )
        TG3=FLOAT(ITG2)/6000.
        TGS=TG3-TG2
C
        WRITE(6,*)'CPUTIME FOR MATRIX SOLVING = ',TGS,' MINUTES'
        WRITE(6,*)' '
        WRITE(6,*)'          TOTAL UNKNOWN NUMBER = ',NUNKNT
        WRITE(6,*)' '
        WRITE(6,*)'          TOTAL CPUTIME = ',TG3,' MINUTES'
        ENDIF
        WRITE(6,*)' '
        WRITE(6,*)'*** OUTPUT DATA FOR BODIES IN FOR003.DAT ***'
        IF(NJCT.GE.0)WRITE(6,*)'*** OUTPUT DATA FOR WIRES IN FOR009.DAT
&***'
        WRITE(6,*)'*** CURRENTS ON SYSTEM IN FOR004.DAT ***'
C
        IF(PRINTC)THEN
C
C WRITE THE CURRENT DENSITY TABLE.
C
        WRITE(3,22)
        WRITE(4,22)

```

```

22      FORMAT(/28X,'SURFACE CURRENTS'/)
        WRITE(3,23)
        WRITE(4,23)
23      FORMAT(1X,'EDGE NUMBER ',13X,'CURRENT DENSITY (AMPS/METER)')
        WRITE(3,24)
        WRITE(4,24)
24      FORMAT(14X,'REAL',9X,'IMAGINARY',7X,'MAGNITUDE',7X,
$'PHASE(DEG)')
        CO=(0.,0.)
        K1=0
        DO 50 I50=1,NEDGES
          IF(NCONN(3,I50).EQ.0)THEN
            WRITE(3,101)I50,CO,0.
            WRITE(4,101)I50,CO,0.
            A=0
            ELSE
              DO 35 I35=1,NCONN(3,I50)
                K1=K1+1
                RA1=REAL(CV(K1))
                RA2=AIMAG(CV(K1))
                RA3=CABS(CV(K1))
                EPS=1.E-7
                IF(ABS(RA1).LT.EPS)THEN
                  RA4=90.
                ELSE
                  RA4=ATAN2(RA2,RA1)/DEG2RAD
                ENDIF
                WRITE(3,101) I50,RA1,RA2,RA3,RA4
                WRITE(4,101) I50,RA1,RA2,RA3,RA4
35          CONTINUE
            ENDIF
50          CONTINUE
101         FORMAT(2X,I4,2X,3(2X,E12.5,2X),F12.3)
        ENDIF
C
C PRINT CURRENTS ON THE WIRES
C
        CALL WIROUT(CV,NUNKNT,NUNKNB,MULTW,NWNOD)
C
40      CONTINUE
-----
C COMPUTE FAR FIELD
C
        IF(IPAT.GT.0) THEN
C

```

```

      CALL PATTEN(DATNOD,NCONN,NBOUND,IEDGF,MULT1,NNODES,WIRSUM,
$NEDGES,NFACES,NUNKNT,CV,INDSUM,IPAT,NJCT,MNJUN,MNJFACE,NWNOD,
$ANG,NJFACE,MIFACE,NWJUN,NBJUN,MXWMLT,NWSEG,NUNKNB,WNODE,MULTW,
$NSEGC,WSEGH,RAD,INSEG)
C
      WRITE(6,*)'*** PATTERN OUTPUT IN FOR010.DAT ***'
C
      ENDIF
-----
C COMPUTE CHARGE DENSITY
C
      IF(ITOT.GT.0)THEN
C
      CALL CHARGE(NJCT,CV,DATNOD,NCONN,NBOUND,NNODES,NEDGES,
$NFACES,NUNKNT,NWNOD,NWSEG,NUNKNB,WNODE,WIRSUM,NSEGC,MULTW,
$ MNJUN,MNJFACE,ANG,NJFACE,MIFACE,NWJUN,NBJUN,INDSUM,
$IEDGF,MULT1,INSEG,MXWMLT)
C
      WRITE(6,*)'*** CHARGE DENSITY OUTPUT IN FOR011.DAT ***'
      ENDIF
-----
20  CONTINUE
      RETURN
      END

```

A.33 WIRMUL

```

C=====
      SUBROUTINE WIRMUL(NJCT,MXWMLT,NWNOD,NWSEG,NSEGC,NWUNKS,MULTW,
& INSEG,MNJUN,NWJUN)
C-----
C FIND MULTIPLICITY FOR EACH NODE
* INPUT:
* MXWMLT=MAXIMUM MULTIPLICITY OF ANY WIRE NODE.
* NWNOD=NUMBER OF WIRE NODES.
* NWSEG=NUMBER OF WIRE SEGMENTS.
* NSEGC(J,N) J=1,2 N=1,NWNOD
* THE NTH WIRE SEGMENT RUNS FROM NODE NSEGC(1,N) TO NSEGC(2,N).
* OUTPUT:
* NWUNKS=THE NUMBER OF WIRE UNKNOWNNS.
* MULTW(N)=THE MULTIPLICITY OF THE NTH WIRE NODE,N=1,...,NWNOD.
* INSEG(M,N) M=1,...,MULTW(N)+1 ARE THE NUMBERS OF THE SEGMENTS
* (IN INCREASING ORDER)THAT ARE ATTACHED TO THE NTH NODE.

```

```

C-----
      DIMENSION MULTW(NWNOD),NSEGC(2,NWSEG),INSEG(MXWMLT+1,NWNOD)
      &           ,NWJUN(MNJUN)
C
      NWUNKS=0
      DO 1 N=1,NWNOD
      MULTW(N)=-1
1     CONTINUE
      DO 6 N=1,NWNOD
        DO 5 J=1,MXWMLT+1
          INSEG(J,N)=0
        5     CONTINUE
      6     CONTINUE
      DO 100 N=1,NWNOD
      MULTW(N)=-1
C
C ADD 1 TO MULTW(N) IF N IS A JUNCTION NODE
C
      IF(NJCT.EQ.1) THEN
      DO 11 NJ=1,MNJUN
11     IF(N.EQ.NWJUN(NJ))MULTW(N)=0
      ENDIF
C
      DO 50 NS=1,NWSEG
        IF(NSEGC(1,NS).EQ.N.OR.NSEGC(2,NS).EQ.N) THEN
          MULTW(N)=MULTW(N)+1
          M=MULTW(N)+1
          INSEG(M,N)=NS
        ENDIF
50     CONTINUE
        NWUNKS=NWUNKS+MULTW(N)
100    CONTINUE
      WRITE(9,*)'   TOTAL UNKNOWN NUMBER = ',NWUNKS
      WRITE(9,*)' '
      WRITE(9,*)'   NODE #           MULTIPLICITY'
      DO 60 J=1,NWNOD
        WRITE(9,*)J,MULTW(J)
      60    CONTINUE
      DO 70 N=1,NWNOD
        DO 65 J=1,MXWMLT+1
          WRITE(9,*)N,J,INSEG(J,N)
        65    CONTINUE
      70    CONTINUE
      RETURN
      END

```

A.34 EDGMUL

```
C=====
C          SUBROUTINE EDGMUL(IEDGF,MULT1,NEDGES,MULTE,IEDGE,IFACE,JE1,JE2)
C-----
C  INPUT:
C  IEDGF FROM ROUTINE EDGFAC CONTAINS THE FACES CONNECTED TO EACH EDGE.
C  MULT1= THE MAXIMUM MULTIPLICITY OF ANY EDGE +1.
C  NEDGES= THE NUMBER OF EDGES.
C  MULTE IS THE MULTIPLICITY OF EDGE IEDGE.
C  IEDGE IS THE EDGE NUMBER.
C  IFACE IS THE FACE NUMBER.
C  OUTPUT:JE1 JG2
C  IF IFACE IS THE LOWEST NUMBERED FACE ATTACHED TO IEDGE THEN
C    JE1=1 AND JE2=MULTE
C  ELSEIF IFACE IS THE ITH FACE ATTACHEL TO IEDGE THEN
C    JE1=JE2=I-1
C-----
          INTEGER IEDGF(MULT1,NEDGES)
          IF(IEDGF(1,IEDGE).EQ.IFACE)THEN
              JE1=1
              JE2=MULTE
          ELSE
              M1=MULTE+1
              DO 10 I=2,M1
                  IF(IFACE.EQ.IEDGF(I,IEDGE))THEN
                      I1=I-1
                      JE1=I1
                      JE2=I1
                      GO TO 11
                  ENDIF
              CONTINUE
10          CONTINUE
11          CONTINUE
          ENDIF
          RETURN
          END
```

A.35 BCUMUL

```
C=====
      SUBROUTINE BCUMUL(NCONN,INDSUM,NEDGES)
C-----
C CUMULATE MULTIPLICITY OF EDGE
C INPUT:
C   NEDGES = THE NUMBER OF EDGES.
C   IE = A SPECIFIC EDGE NUMBER 1.LE.IE.LE.NEDGES.
C   NCONN(I,J) I=1,2,3: THE JTH EDGE RUNS FROM NODE NCONN(1,J) TO
C     NCONN(2,J) AND HAS MULTIPLICITY NCONN(3,J) (AFTER ACCOUNTING
C     FOR ALL GROUND PLANE ATTACHMENTS).
C
C OUTPUT:
C   IND = THE SUM OF THE EDGE MULTIPLICITIES UP TO (BUT NOT INCLUDING)
C     THE CURRENT EDGE(IE).
C-----
      DIMENSION NCONN(3,NEDGES),INDSUM(NEDGES)
      INDSUM(1)=0
      DO 10 I=2,NEDGES
      INDSUM(I)=NCONN(3,I-1)+INDSUM(I-1)
10  CONTINUE
      RETURN
      END
```

A.36 FACEDG

```
C=====
      SUBROUTINE FACEDG(NFACES,NBOUND,IFACE,IEDG)
C-----
C MAPPING FROM FACE TO EDGE
      INTEGER NBOUND(3,NFACES),IEDG(3)
      DO 2 I=1,3
2   IEDG(I)=NBOUND(I,IFACE)
      RETURN
      END
```

A.37 FACVTX

```
C=====
      SUBROUTINE FACVTX(NCONN,NEDGES,IE,NV)
C-----
```

```

C-----
C MAPPING FROM FACE TO VERTEX
  INTEGER NCONN(3,NEDGES),IE(3),NV(3)
  IF(NCONN(1,IE(2)).EQ.NCONN(1,IE(3)).OR.NCONN(1,IE(2)).EQ.NCONN(2,
$IE(3)))THEN
    NV(1)=NCONN(1,IE(2))
  ELSE
    NV(1)=NCONN(2,IE(2))
  ENDIF
  IF(NCONN(1,IE(1)).EQ.NCONN(1,IE(3)).OR.NCONN(1,IE(1)).EQ.NCONN(2,
$IE(3)))THEN
    NV(2)=NCONN(1,IE(1))
  ELSE
    NV(2)=NCONN(2,IE(1))
  ENDIF
  IF(NCONN(1,IE(1)).EQ.NCONN(1,IE(2)).OR.NCONN(1,IE(1)).EQ.NCONN(2,
$IE(2)))THEN
    NV(3)=NCONN(1,IE(1))
  ELSE
    NV(3)=NCONN(2,IE(1))
  ENDIF
  RETURN
  END

```

A.38 VTXCRD

```

C=====
  SUBROUTINE VTXCRD(DATNOD,NNODES,N,DN)
C-----
C MAPPING FROM VERTEX TO COORDINATE
  DIMENSION DATNOD(3,NNODES),N(3),DN(3,3)
  DO 2 I=1,3
  DO 2 J=1,3
2   DN(I,J)=DATNOD(J,N(I))
  RETURN
  END

```

A.39 WIRMAN

```

C=====
  SUBROUTINE WIRMAN(IFREQ,CZ,CV,NUNKNT,NUNKNB,NWNOD,NWSEG,NWUNKS,

```



```

& WNODE,MULTW,NSEGC,WSEGH,RAD,INSEG,WIRSUM,DATNOD,
& NCONN,NBOUND,NNODES,NEDGES,NFACES,IEDGF,MULT1,INDSUM,NJCT,MNJUN,
& NWJUN,NBJUN,NJFACE,MNJFACE,MIFACE,ANG,MXWMLT,IPAT)

```

```

C-----
* MXUNKN = MAXIMUM NUMBER OF UNKNOWNNS EXPECTED. *
* MXWNOD = MAXIMUM NUMBER OF WIRE NODES. *
* MXWMLT = MAXIMUM MULTIPLICITY THAT ANY WIRE NODE MAY HAVE. *
* MXWSEG = MAXIMUM NUMBER OF WIRE SEGMENTS. *
* MXWZN = MAXIMUM NUMBER OF LUMPED IMPEDENCES ON WIRES. *
* MXWVLT = MAXIMUM NUMBER OF DELTA GAP VOLTAGE SOURCES ON THE WIRES. *
* MXFREQ = MAXIMUM NUMBER OF FREQUENCIES EXPECTED. *
* MXCIT=MAXIMUM NUMBER OF EXCITATIONS ASSUMED THE SAME FOR ALL FREQS. *
* MXCIV=MAXIMUM NUMBER OF VOLTAGE EXCITATIONS. *
* MXCIP=MAXIMUM NUMBER OF PLANE WAVE EXCITATIONS. *
C-----

```

```

C
C PARAMETERS ASSOCIATE WITH JUNCTIONS
C

```

```

REAL ANG(MNJUN,MNJFACE)
INTEGER NJFACE(MNJUN,MNJFACE),MIFACE(MNJUN),NBJUN(MNJUN),
& NWJUN(MNJUN)

```

```

C
INTEGER IEDGF(MULT1,NEDGES)
DIMENSION INDSUM(NEDGES)
INTEGER NCONN(3,NEDGES),NBOUND(3,NFACES),IM(3)
DIMENSION DATNOD(3,NNODES),TMAT(3,3),SM(3)
DIMENSION WNODE(3,NWNOD),WSEGH(3,NWSEG),RAD(NWSEG)
INTEGER MULTW(NWNOD),NSEGC(2,NWSEG),
> INSEG(MXWMLT+1,NWNOD),WIRSUM(NWNOD)
COMPLEX CZ(NUNKNT,NUNKNT),CV(NUNKNT)
COMMON/WIRE/IQUADW
COMMON/WIRSLF/IQWS
IQUADW=4
IQWS=8

```

```

C
CALL WCUMUL(NWNOD,NUNKNB,MULTW,WIRSUM)
CALL MTXWIR(MXWMLT,NWNOD,NWSEG,NUNKNB,NUNKNT,WNODE,MULTW,
> NSEGC,WSEGH,RAD,INSEG,CZ,CV,WIRSUM,DATNOD,NCONN,NBOUND,
$ NNODES,NEDGES,NFACES,IEDGF,MULT1,INDSUM,NJCT,MNJUN,NWJUN,
$ NBJUN,NJFACE,MNJFACE,MIFACE,ANG,IPAT)
RETURN
END

```

A.40 MTXWIR

```

C=====
SUBROUTINE MTXWIR(MXWMLT,NWNOD,NWSEG,NUNKNB,NUNKNT,WNODE,MULTW,
> NSEGC,WSEGH,RAD,INSEG,CZ,CV,WIRSUM,DATNOD,NCONN,NBOUND,
$ NNODES,NEDGES,NFACES,IEDGF,MULT1,INDSUM,NJCT,MNJUN,NWJUN,
$ NBJUN,NJFACE,MNJFACE,MIFACE,ANG,IPAT)
C-----
C FILL MATRIX ELEMENTS WHICH SOURCE POINT ON WIRE
C PARAMETERS ASSOCIATE WITH JUNCTIONS
C
  REAL ANG(MNJUN,MNJFACE)
  INTEGER NJFACE(MNJUN,MNJFACE),MIFACE(MNJUN),NBJUN(MNJUN),
&     NWJUN(MNJUN)
C
  INTEGER IEDGF(MULT1,NEDGES)
  DIMENSION INDSUM(NEDGES)
  INTEGER NCONN(3,NEDGES),NBOUND(3,NFACES),IM(3)
  DIMENSION DATNOD(3,NNODES),TMAT(3,3),SM(3)
C
  DIMENSION WNODE(3,NWNOD),MULTW(NWNOD),NSEGC(2,NWSEG),
>     WSEGH(3,NWSEG),RAD(NWSEG),INSEG(MXWMLT+1,NWNOD)
  INTEGER WIRSUM(NWNOD)
  COMPLEX CZ(NUNKNT,NUNKNT),CV(NUNKNT)
  REAL LAMBDA,MU,K,IMP
C
* DATA PASSED INTO WSOLTN FROM WINDAT
C
  SAVE /WAVE/,/PARAM/,/INC/,/MEDIUM/
  COMMON/WAVE/OMEGA,LAMBDA,K
  COMMON/PARAM/THETA,PHI,IFIELD
  COMMON/INC/HTHETA,HPHI,ETHETA,EPHI
  COMMON/MEDIUM/DEG2RAD,EPSLON,MU,IMP,SL,PI
  COMMON/F/FREQ
C
  CALL ZWW(MXWMLT,NWNOD,NWSEG,NUNKNB,NUNKNT,WNODE,MULTW,NSEGC,
>     WSEGH,RAD,INSEG,CZ,CV,WIRSUM,NWJUN,MNJUN)
C
  CALL ZBW(MXWMLT,NWNOD,NWSEG,NUNKNB,NUNKNT,WNODE,MULTW,NSEGC,
$     WSEGH,RAD,INSEG,CZ,WIRSUM,DATNOD,NCONN,NBOUND,NNODES,NEDGES,
$     NFACES,IEDGF,MULT1,INDSUM,NJCT,MNJUN,MNJFACE,ANG,NJFACE,MIFACE
$     ,NWJUN,NBJUN)
C
40  CONTINUE
499 CONTINUE

```

RETURN
END

A.41 ZWW

```
C=====
  SUBROUTINE ZWW(MXWMLT,NWNOD,NWSEG,NUNKNB,NUNKNT,WNODE,MULTW,NSEGC,
    > WSEGH,RAD,INSEG,CZ,CV,WIRSUM,NWJUN,MNJUN)
C-----
C FILL MATRIX ELEMENTS WHICH SOURCE AND MATCHING POINTS ON WIRE
C
* INPUT:
* MXWMLT = MAXIMUM MULTIPLICITY THAT ANY WIRE NODE MAY HAVE.
* NWNOD = NUMBER OF WIRE NODES.
* NWSEG = NUMBER OF WIRE SEGMENTS.
* NUNKNB = NUMBER OF BODY UNKNOWNNS.
* NUNKNT = TOTAL NUMBER OF UNKNOWNNS(BODY AND WIRE).
* WNODE(I,J) I=1,2,3 CONTAINS THE X,Y,Z COORDINATES OF THE JTH WIRE
* NODE.
* MULTW(N) CONTAINS THE MULTIPLICITY OF THE NTH NODE.
* NSEGC(I,J):THE JTH WIRE SEGMENT RUNS FROM NSEGC(1,J) TO NSEGC(2,J).
* WSEGH(I,J) I=1,2,3 CONTAINS THE X,Y,Z COORDINATES OF THE MIDPOINT OF
* JTH WIRE SEGMENT.
* RAD(N) CONTAINS THE RADIUS OF THE NTH WIRE SEGMENT.
* INSEG(M,N) M=1,...,MULTW(N) ARE THE SEGMENTS (IN INCREASING
* ORDER) ATTACHED TO THE NTH WIRE NODE.
* NWCZN = THE NUMBER OF LUMPED IMPEDENCE LOADS ON THE WIRE.
* IWCZN(1,I) CONTAINS THE NODE LOCATION OF THE ITH IMPEDENCE LOAD.
* IWCZN(2,I)=1 IF THE LOAD IS ON THE FIRST NODE OF THE SEGMENT
*           2 IF THE LOAD IS ON THE SECOND NODE OF THE SEGMENT
* I=1,...,NWCZN.
* CWZN(I) CONTAINS THE VALUE OF THE COMPLEX LUMPED IMPEDENCE FOR THE
* IWCZN(1,I)TH NODE, I=1,...,NWCZN.
* NWVLT = THE NUMBER OF DELTA GAP VOLTAGE SOURCES ON THE WIRES.
* IWVLT(1,I) CONTAINS THE NODE LOCATION OF THE ITH DELTA GAP VOLTAGE
* SOURCE ON THE WIRES.
* IWVLT(2,I)=1 IF THE SOURCE IS LOCATED ON THE FIRST NODE
*           2 IF THE SOURCE IS LOCATED ON THE 2ND NODE OF THE SEGMENT
* CWVLT(I) CONTAINS THE VALUE OF THE COMPLEX DELTA GAP VOLTAGE
* SOURCE OF THE IWVLT(1,I)TH NODE. A PASSIVE SIGN CONVENTION IS WITH
* RESPECT TO THE CURRENT IN THE DIRECTION OF THAT SEGMENT.
* CZ THE IMPEDANCE MATRIX DIMENSIONED NUNKNT*NUNKNT
* CV IS THE VOLTAGE FORCING FUNCTION VECTOR DIMENSIONED NUNKNT
```

```

* OUTPUT:
* THE PORTION OF THE IMPEDANCE MATRIX ASSOCIATED WITH THE WIRE WIRE
* COUPLING IS FILLED AND ALSO THE WIRE PORTION OF THE FORCING VECTOR IS
* FILLED.
* CZ(MM,NN)=J*OMEGA*(VAP(M,N) DOT VLP(M,N)+VAM(M,N) DOT VLM(M))
*           -(SPOTP(M,N)-SPOTM(M,N))
* CV(MM)=VEP(M) DOT VLP(M)+VEM(M) DOT VLM(M) -ANY WIRE VOLTAGE SOURCE
* MM=M+NUNKNB NN=N+NUNKNB
* WHERE
* VAP(M,N) AND VAM(M,N) ARE THE VECTOR POTENTIALS DUE TO THE
* N TH WIRE BASIS FUNCTIONS EVALUATED AT THE + AND - CENTROIDS
* OF THE M TH WIRE BASIS FUNCTION.
* SPOTP(M,N) AND SPOTM(M,N) ARE THE SCALAR POTENTIALS DUE TO
* THE N TH WIRE BASIS FUNCTION EVALUATED AT THE + AND THE -
* CENTROIDS OF THE M TH WIRE BASIS FUNCTION.
* N=1,...,NWUNKS M=1,...,NWUNKS
* VLP(M) IS THE VECTOR WHICH RUNS FROM THE THE BEGINNING OF
* THE + SEGMENT TO THE CENTROID OF THE + SEGMENT .THIS + SEGMENT IS THE
* + SEGMENT ASSOCIATED WITH THE M TH BASIS FUNCTION.
* VLM(M) IS THE VECTOR WHICH RUNS FROM THE - CENTROID OF THE M TH
* BASIS FUNCTION TO THE ENDPOINT OF THAT SEGMENT.

```

```

-----
DIMENSION WNODE(3,NWNOD),MULTW(NWNOD),NDEGC(2,NWSEG),
>         WSEGH(3,NWSEG),RAD(NWSEG),INTEG(MXWMLT+1,NWNOD),
>         RMK(3),SH(3),RSK(3),T(3)
COMPLEX CZ(NUNKNT,NUNKNT),CV(NUNKNT),
>         CSPW,HTHETA,HPhi,ETHETA,EPhi,EX,EY,EZ,ANS(3),
>         SPOT,VAP(3),VAM(3),SSPOT,VA(3),CETEMP,EDOTT,CARG,CTEMP
>         ,CKRED,CKTOT,CKSELF,CANS
INTEGER WIRSUM(NWNOD),NWJUN(MNJUN)
REAL LAMBDA,K,MU,IMP
EXTERNAL CKRED,CKTOT,CKSELF
LOGICAL LOWS,LOWM,PLUS
SAVE /WAVE/,/PARAM/,/INC/,/MEDIUM/,/WKERNL/,/WIRE/,/WKER/
COMMON/POT/NM,NS,DEL,DELS,DELH,DELRH,RVPW,CSPW,RADMKS
COMMON/WAVE/OMEGA,LAMBDA,K
COMMON/MEDIUM/DEG2RAD,EPSLON,MU,IMP,SL,PI
COMMON/PARAM/THETA,PHI,NFIELD
COMMON/INC/HTHETA,HPhi,ETHETA,EPhi
COMMON/WKERNL/RSK,SH,RMK,RADSK,RADSKS
COMMON/WKER/DPAR,RHO,RHOPR,RHOPRS,RHOMRS
COMMON/WIRE/IQUADW
COMMON/WIRSLF/IQWS

```

C

```

* SET CONSTANTS.
C
  EQUAD=IQUADW
  EQUAWS=IQWS
  PI4=4.*PI
  RVPW=MU/PI4
  CSPW=CMPLX(0.,1./((OMEGA*EPSLON*PI4))
  CTHETA=COS(THETA)
  STHETA=SIN(THETA)
  CPHI=COS(PHI)
  SPHI=SIN(PHI)
C
* CARTESIAN COMPONENTS OF THE INCIDENT FIELD ARE:
C
  EX=ETHETA*CTHETA*CPHI-EPHI*SPHI
  EY=ETHETA*CTHETA*SPHI+EPHI*CPHI
  EZ=-ETHETA*STHETA
C
* LOOP OVER THE SOURCE SEGMENTS.
C
  DO 1000 NS=1,NWSEG
    IF(NS.EQ.1.OR.NFIELD.LE.1)THEN
      RADSK=K*RAD(NS)
      RADSKS=RADSK*RADSK
      DELRH=15.*RADSK
C
* OBTAIN K* THE COORDINATES OF THE SOURCE SEGMENT CENTROID
C
      NSF=NSEGC(1,NS)
      NST=NSEGC(2,NS)
      DO 2 J=1,3
        RSK(J)=K*WSEGH(J,NS)
        SH(J)=K*(WNODE(J,NST)-WNODE(J,NSF))
2      CONTINUE
      DEL=SQRT(SH(1)*SH(1)+SH(2)*SH(2)+SH(3)*SH(3))
      DELS=DEL*DEL
      DELH=.5*DEL
      DO 4 J=1,3
        SH(J)=SH(J)/DEL
4      CONTINUE
C
* LOOP OVER THE MATCH SEGMENTS.
C
      DO 500 NM=1,NWSEG
        RADMK=K*RAD(NM)

```

```

RADMKS=RADMK*RADMK
C
* OBTAIN COORDINATES OF TEST VECTOR AND MATCH SEGMENT CENTROID TIMES K.
C
      NMF=NSEGC(1,NM)
      NMT=NSEGC(2,NM)
      DO 5 J=1,3
        T(J)=.5*(WNODE(J,NMT)-WNODE(J,NMF))
        RMK(J)=K*WSEGH(J,NM)
5     CONTINUE
C
* COMPUTE QUANTITIES FOR CALCULATION OF THE VECTOR AND SCALAR POTENTIALS
C
      CALL POTWIR(SSPOT,VAP,VAM,0)
      IV=0
C
* LOOP OVER NODES ATTACHED TO THE SOURCE SEGMENT.
C
      DO 400 JNS=1,2
        NODES=NSEGC(JNS,NS)
        MULTS=MULTW(NODES)
        IF(MULTS.GT.0)THEN
C
* COMPUTE COLUMN INDEX FOR SOURCE SEGMENT.
C
          INDC=WIRSUM(NODES)
C
* DETERMINE WHETHER SEGMENT NS IS THE LOWEST SEGMENT ATTACHED TO NODE
* NODES. LOOP OVER THE NUMBER OF SEGMENTS ATTACHED TO NS AT NODE NODES.
C
          CALL NODMUL(INSEG,MXWMLT,NWNOD,NODES,MULTS,NS,JS1,JS2,
>             LOWS)
          IF(LOWS.AND.NODES.EQ.NWJUN(MNJUN)) THEN
            JS1=1
            JS2=1
            LOWS=.FALSE.
          ENDIF
          DO 300 J=JS1,JS2
            ICOL=INDC+J
            IV=IV+1
            IF(NODES.EQ.NSEGC(2,NS))THEN
              PLUS=.TRUE.
            ELSE
              PLUS=.FALSE.
            ENDIF

```

```

                IF(LOWS)THEN
C
* OBTAIN THE ATTACHED SEGMENT.
C
                NSEGAS=INSEG(J+1,NODES)
C
* SGN=+1 IF SEGMENT NS & ATTACHED SEGMENT RUN IN THE SAME DIRECTION,
* SGN=-1 IF SEGMENT NS & ATTACHED SEGMENT RUN IN THE OPPOSITE DIRECTION
C
                IF(NSEGC(2,NS).EQ.NSEGC(1,NSEGAS).OR.
>                 NSEGC(1,NS).EQ.NSEGC(2,NSEGAS))THEN
                    SGN=1.
                ELSE
                    SGN=-1.
                ENDIF
                ELSE
                    SGN=1.
                ENDIF
C
C
* COMPUTE APPROPRIATE VECTOR AND SCALAR POTENTIALS.
C
                IF(PLUS)THEN
C
C FOR PLUS SOURCE SEGMENT SET SIGN= (-1) TO VECTOR AND SCALAR POTENTIALS
C
                IF(NODES.EQ.NWJUN(MNJUN)) SGN=-1.
                    DO 30 JJ=1,3
                        VA(JJ)=VAP(JJ)*SGN
30                 CONTINUE
                    SPOT=SSPOT*SGN
                ELSE
C
C FOR MINUS SOURCE SEGMENT SET SIGN= (+1) TO VECTOR POTENTIAL AND
C (-1) TO SCALAR POTENTIALS
C
                IF(NODES.EQ.NWJUN(MNJUN)) SGN=1.
                    DO 40 JJ=1,3
                        VA(JJ)=VAM(JJ)*SGN
40                 CONTINUE
                    SPOT=-SSPOT*SGN
                ENDIF
                CTEMP=(VA(1)*T(1)+VA(2)*T(2)+VA(3)*T(3))*
>                 CMLX(0.,OMEGA)
                IF(NS.EQ.1 .AND. IV.EQ.1)THEN

```

```

C
* COMPUTE QUANTITIES ASSOCIATED WITH THE INCIDENT FIELD.
C
      ARGMNT=(RMK(1)*CPHI+RMK(2)*SPHI)*STHETA+
>         RMK(3)*CTHETA
      CARG=CMPLX(0.,ARGMNT)
      EDOTT=EX*T(1)+EY*T(2)+EZ*T(3)
      CETEMP=EDOTT*CEXP(CARG)
      ENDIF
C
* LOOP OVER THE NODES ATTACHED TO THE MATCH SEGMENT.
C
      DO 250 JMN=1,2
      NODEM=NSEGC(JMN,NM)
      MULTM=MULTW(NODEM)
      IF(MULTM.GT.0)THEN
C
* COMPUTE ROW INDEX FOR SOURCE SEGMENT.
C
      INDR=WIRSUM(NODEM)
C
* DETERMINE WHETHER SEGMENT NM IS THE LOWEST SEGMENT ATTACHED
* TO SEGMENT NM AT NODE NODEM.LOOP OVER THE NUMBER OF SEGMENTS ATTACHED
* TO SEGMENT NM AT NODE NODEM.
C
      CALL NODMUL(INSEG,MXWMLT,NWNOD,NODEM,MULTM,NM,
>         JM1,JM2,LOWM)
      IF(LOWM.AND.NODEM.EQ.NWJUN(MNJUN)) THEN
      JM1=1
      JM2=1
      LOWM=.FALSE.
      ENDIF
      DO 200 JM=JM1,JM2
C
* DETERMINE WHETHER SEGMENT IS A PLUS OR MINUS SEGMENT.
C
      IF(NODEM.EQ.NSEGC(2,NM))THEN
      SGN1=1.
      ELSE
      SGN1=-1.
      ENDIF
      IF(LOWM)THEN
C
* CASE: SEGMENT NM IS THE LOWEST NUMBERED SEGMENT ATTACHED TO NODE
* NODEM. DIRECTION OF THE BASIS FUNCTION IS DETERMINED BY THE ATTACHED

```



```

* SEGMENT. OBTAIN ATTACHED SEGMENT.
C
      NSEGAM=INSEG(JM+1,NODEM)
      IF(NSEGC(2,NM).EQ.NSEGC(1,NSEGAM).OR.
>      NSEGC(1,NM).EQ.NSEGC(2,NSEGAM))THEN
          SGN=1.
      ELSE
          SGN=-1.
      ENDIF
      ELSE
          SGN=1.
      ENDIF
C CHECK IF THIS IS A JUNCTION NODE
      IF(NODEM.EQ.NWJUN(MNJUN))THEN
C CHECK IF THIS IS A PLUS SEGMENT
          IF(NODEM.EQ.NSEGC(2,NM))THEN
C
C FOR PLUS MATCHING SEGMENT SET SIGN= (-1) TO VECTOR POTENTIAL AND
C (+1) TO SCALAR POTENTIALS
C
C FOR MINUS MATCHING SEGMENT SET SIGN= (+1) TO VECTOR POTENTIAL AND
C (+1) TO SCALAR POTENTIALS
C
      SGN=-1.
      SGN1=1.
      ELSE
          SGN=1.
          SGN1=-1.
      ENDIF
      ENDIF
C
      IROW=INDR+JM
      IF(NS.EQ.1.AND.IV.EQ.1)CV(IROW)=
>      CV(IROW)+SGN*CETEMP
      IF(NFIELD.EQ.1)CZ(IROW,ICOL)=
>      CZ(IROW,ICOL)+SGN*(CTEMP-SPOT*SGN1)
200          CONTINUE
          ENDIF
250          CONTINUE
300          CONTINUE
          ENDIF
400          CONTINUE
500          CONTINUE
          ENDIF
1000 CONTINUE

```

RETURN
END

A.42 ZBW

```
C-----  
      SUBROUTINE ZBW(MXWMLT,NWNOD,NWSEG,NUNKNB,NUNKNT,WNODE,MULTW,NSEGC,  
$      WSEGH,RAD,INSEG,CZ,WIRSUM,DATNOD,NCONN,NBOUND,NNODES,NEDGES,  
$      NFACES,IEDGF,MULT1,INDSUM,NJCT,MNJUN,MNJFACE,ANG,NJFACE,MIFACE  
$      ,NWJUN,NBJUN)  
C-----  
C THIS ROUTINE FILL OUT THE MATRIX ELEMENTS OF ZBW WHICH IS SOURCE POINT  
C ON THE WIRE AND MATCH POINT ON THE BODY  
C PARAMATERS ASSOCIATE WITH JUNCTION PART  
C  
      REAL ANG(MNJUN,MNJFACE),XM(3),YM(3),ZM(3),VMJUN(3)  
      INTEGER NJFACE(MNJUN,MNJFACE),MIFACE(MNJUN),NBUN(MNJUN),  
&      NWJUN(MNJUN),NM(3)  
C  
      INTEGER IEDGF(MULT1,NEDGES)  
      DIMENSION INDSUM(NEDGES)  
      DIMENSION WNODE(3,NWNOD),MULTW(NWNOD),NSEGC(2,NWSEG),  
>      WSEGH(3,NWSEG),RAD(NWSEG),INSEG(MXWMLT+1,NWNOD),  
>      RMK(3),SH(3),RSK(3),T(3)  
      INTEGER NCONN(3,NEDGES),NBOUND(3,NFACES),IM(3)  
      DIMENSION DATNOD(3,NNODES),TMAT(3,3),SM(3)  
      COMPLEX CZ(NUNKNT,NUNKNT),  
>      CSPW,HTHETA,HPHI,ETHETA,EPHI,EX,EY,EZ,ANS(3),  
>      SPOT,VAP(3),VAM(3),SSPOT,VA(3),CETEMP,EDOTT,CARG,CTEMP  
>      ,CKRED,CKTOT,CKSELF,CANS,CJWEMP,CX  
      INTEGER WIRSUM(NWNOD),IGNDP(3)  
      REAL LAMBDA,K,MU,IMP,DM(3,3),DLM(3),DMC(3)  
      EXTERNAL CKRED,CKTOT,CKSELF  
      LOGICAL LOWS,LOWM,PLUS  
      SAVE /WAVE/,/PARAM/,/INC/,/MEDIUM/,/WKERNL/,/WIRE/,/WKER/  
      COMMON/POT/NFM,NS,DEL,DELS,DELH,DELRH,RVPW,CSPW,RADMKS  
      COMMON/WAVE/OMEGA,LAMBDA,K  
      COMMON/MEDIUM/DEG2RAD,EPSLON,MU,IMP,SL,PI  
      COMMON/PARAM/THETA,PHI,NFIELD  
      COMMON/INC/HTHETA,HPHI,ETHETA,EPHI  
      COMMON/WKERNL/RSK,SH,RMK,RADSK,RADSKS  
      COMMON/WKER/DPAR,RHO,RHOPR,RHOPRS,RHOMRS  
      COMMON/WIRE/IQUADW
```

```

COMMON/WIRSLF/IQWS
COMMON/GPLANE/NGNDP,IGNDP
C
* SET CONSTANTS.
C
SIZE(X,Y,Z)=SQRT(X*X+Y*Y+Z*Z)
NFM=0
IQUAD=IQUADW
IQUAWS=IQWS
PI4=4.*PI
RVPW=MU/PI4
CSPW=CMPLX(0.,1./(OMEGA*EPSLON*PI4))
C
* LOOP OVER THE SOURCE SEGMENTS OF THE WIRE
C
CALL BCUMUL(NCONN,INDSUM,NEDGES)
DO 1000 NS=1,NWSEG
  IF(NS.EQ.1.OR.NFIELD.LE.1)THEN
    RADSK=K*RAD(NS)
    RADSKS=RADSK*RADSK
    DELRH=15.*RADSK
C
* OBTAIN K* THE COORDINATES OF THE SOURCE SEGMENT CENTROID
C
NSF=NSEGC(1,NS)
NST=NSEGC(2,NS)
DO 2 J=1,3
  RSK(J)=K*WSEGH(J,NS)
  SH(J)=K*(WNODE(J,NST)-WNODE(J,NSF))
2 CONTINUE
DEL=SQRT(SH(1)*SH(1)+SH(2)*SH(2)+SH(3)*SH(3))
DELS=DEL*DEL
DELH=.5*DEL
DO 4 J=1,3
  SH(J)=SH(J)/DEL
4 CONTINUE
C
C LOOP OVER FACE NUMBERS OF THE MATCH TRIANGLES OF THE BODY
DO 500 IFM=1,NFACES
  IIM=1
C OBTAIN THE EDGES OF THE MATCH TRIANGLE
CALL FACEDG(NFACES,NBOUND,IFM,IM)
C OBTAIN THE VERTICES OF THE MATCH TRIANGLE
CALL FACVTX(NCONN,NEDGES,IM,NM)
C OBTAIN THE COORDINATES OF THE MATCH TRIANGLE'S VERTICIES

```

```

        CALL VTXCRD(DATNOD,NNODES,NM,DM)
    DO 14 J=1,3
C CALCULATE THE CENTROID OF THE MATCH TRIANGLE
        DMC(J)=(DM(1,J)+DM(2,J)+DM(3,J))/3
C RMK(J) IS THE MATCH POINT
14      RMK(J)=K*DMC(J)
        DO 6 I=1,3
        DO 6 J=1,3
C TESTING VECTOR = 0.5*THE VECTOR RUNNING FROM THE ITH VERTICE TO CENTROID
6        TMAT(I,J)=(DMC(J)-DM(I,J))/2.
        DO 8 I=1,3
        IP1=MOD(I,3)+1
        IM1=MOD(I+1,3)+1
C COMPUTE THE EDGE LENGTH OF THE MATCH TRIANGLE
        DO 7 J=1,3
7          DLM(J)=DM(IM1,J)-DM(IP1,J)
8          SM(I)=SIZE(DLM(1),DLM(2),DLM(3))
C-----
C IF NJCT=0, NO WIRE JUNCTION WITH BCJY
C IF NJCT=1, CHECK IF ANY JUNCTION IN THIS MATCH TRIANGLE
C IF JMF=0, NO JUNCTION IN THIS MATCH TRIANGLE
C IF JMF=N, N=1,2,3, FIND OUT ASSOCIATED PARAMETERS
C
        IF(NJCT.EQ.1)THEN
C
C TO FIND PARAMATERS ASSOCIATE WITH THIS JUNCTION
C
        CALL JUNPAR(JMF,DM,NM,IFM,ANGM,VMJUN,JMROW,
&          WIRSUM,MNJUN,MNJFACE,NWNOD,ANG,NJFACE,MIFACE,NWJUN,NBJUN)
C
C ANGM IS THE ANGULAR DISTRIBUTION COEFFICIENCY OF MATCH TRIANGLE
C
        ELSE
        JMF=0
        ENDIF
C
* COMPUTE QUANTITIES FOR CALCULATION OF THE VECTOR AND SCALAR POTENTIALS
C
        CALL POTWIR(SSPOT,VAP,VAM,')
        IV=0
C
* LOOP OVER NODES ATTACHED TO THE SOURCE SEGMENT
C
        DO 400 JNS=1,2
        NODES=NSEGC(JNS,NS)

```

```

        MULTS=MULTW(NODES)
        IF(MULTS.GT.0)THEN
C
* COMPUTE COLUMN INDEX FOR SOURCE SEGMENT.
C
        INDC=WIRSUM(NODES)
C
* DETERMINE WHETHER SEGMENT NS IS THE LOWEST SEGMENT ATTACHED TO NODE
* NODES. LOOP OVER THE NUMBER OF SEGMENTS ATTACHED TO NS AT NODE NODES.
C
C CHECK IF SOURCE POINT IS A JUNCTION POINT
        CALL NODMUL(INSEG,MXWMLT,NWNOD,NODES,MULTS,NS,JS1,JS2,
>          LOWS)
        IF(LOWS.AND.NODES.EQ.NWJUN(MNJUN)) THEN
            JM1=1
            JM2=1
            LOWS=.FALSE.
            ENDIF
                DO 300 J=JS1,JS2
                    ICOL=INDC+J
                    IV=IV+1
                    IF(NODES.EQ.NSEGC(2,NS))THEN
                        PLUS=.TRUE.
                    ELSE
                        PLUS=.FALSE.
                    ENDIF
                    IF(LOWS)THEN
C
* OBTAIN THE ATTACHED SEGMENT.
C
                    NSEGAS=INSEG(J+1,NODES)
C
* SGN=+1 IF SEGMENT NS & ATTACHED SEGMENT RUN IN THE SAME DIRECTION,
* SGN=-1 IF SEGMENT NS & ATTACHED SEGMENT RUN IN THE OPPOSITE DIRECTION
C
                    IF(NSEGC(2,NS).EQ.NSEGC(1,NSEGAS).OR.
>          NSEGC(1,NS).EQ.NSEGC(2,NSEGAS))THEN
                        SGN=1.
                    ELSE
                        SGN=-1.
                    ENDIF
                    ELSE
                        SGN=1.
                    ENDIF
C

```

```

* COMPUTE APPROPRIATE VECTOR AND SCALAR POTENTIALS.
C
      IF(PLUS)THEN
C FOR THE JUNCTION POINT SET SGN=-1 TO THE PLUS SEGMENT
      IF(NODES.EQ.NWJUN(MNJUN)) SGN=-1.
      DO 30 JJ=1,3
C VECTOR POTENTIAL
      VA(JJ)=VAP(JJ)*SGN
      30      CONTINUE
C SCALAR POTENTIAL
      SPOT=SSPOT*SGN
      ELSE
C FOR THE JUNCTION SET SGN=1 TO THE MINUS SEGMENT
      IF(NODES.EQ.NWJUN(MNJUN)) SGN=1.
      DO 40 JJ=1,3
C VECTOR POTENTIAL
      VA(JJ)=VAM(JJ)*SGN
      40      CONTINUE
C SCALAR POTENTIAL
      SPOT=-SSPOT*SGN
      ENDIF

C
C ZJW : SOURCE POINT ON THE WIRE, MATCH POINT ON THE JUNCTION TRIANGLE
C
      IF(JMF.NE.0) THEN
C VECTOR POTENTIAL DOT WITH TESTING VECTOR
      CJWEMP=(VA(1)*VMJUN(1)+VA(2)*VMJUN(2)+VA(3)*VMJUN(3))*
      &      CMLX(0.,OMEGA)
C SET + TO VECTOR POTENTIAL AND - TO SCALAR POTENTIAL
      CX=(CJWEMP-SPOT)*ANGM
      IF(NFIELD.EQ.1)CZ(JMROW,ICOL)=CZ(JMROW,ICOL)
      $ +CX
      ENDIF

C
C LOOP OVER THE EDGES OF THE MATCH TRIANGLE OF THE BODY
      DO 100 IML=1,3
      IF(NCONN(3,IM(IML)).GT.0)THEN
      T1=TMAT(IML,1)
      T2=TMAT(IML,2)
      T3=TMAT(IML,3)
      FLAG=1.
      IF(IML.EQ.1)THEN
      IF(NCONN(1,IM(1)).EQ.NM(3))FLAG=-1.
      ELSEIF(IML.EQ.2)THEN
      IF(NCONN(1,IM(2)).EQ.NM(1))FLAG=-1.

```

```

ELSE
  IF(NCONN(1,IM(3)).EQ.NM(2))FLAG=-1.
ENDIF
CTEMP=(VA(1)*T1+VA(2)*T2+VA(3)*T3)*
$      CMLX(O.,OMEGA)
MULTM=NCONN(3,IM(IML))
IF(MULTM.GT.O)THEN
  IMM=IM(IML)
C FOR AN EDGE WITH MULTIPLICITY MULT, THE LOWEST NUMBERED FACE
C CONTRIBUTES TO MULT BASIS FUNCTIONS ASSOCIATED WITH THAT EDGE,
C WHILE EACH REMAINING FACE CONTRIBUTES TO ONE BASIS FUNCTION ASSOCIATED
C WITH THAT EDGE.
      CALL EDGMUL(IEDGF,MULT1,NEDGES,MULTM,IMM,IFM,
$JM1,JM2)
      INDM=INDSUM(IMM)
      DO 50 JM=JM1,JM2
        IROW=INDM+JM
        CX=FLAG*(CTEMP-SPOT)*SM(IML)
        IF(NFIELD.EQ.1)CZ(IROW,ICOL)=
>          CZ(IROW,ICOL)+CX
50      CONTINUE
      ENDIF
      ENDIF
100     CONTINUE
300     CONTINUE
      ENDIF
400     CONTINUE
500     CONTINUE
      ENDIF
1000  CONTINUE
      RETURN
      END

```

A.43 ZBB

```

C=====
  SUBROUTINE ZBB(DATNOD,NCONN,NBOUND,IEDGF,MULT1,NNODES,NEDGES,
$    NFACES,NUNKNT,CZ,CV,INDSUM,NJCT,MNJUN,MNJFACE,NWNOD,WIRSUM,
$    ANG,NJFACE,MIFACE,NWJUN,NBJUN)
C-----
C THIS ROUTINE FILLS THE BODY-BODY ELEMENTS OF THE IMPEDANCE MATRIX.
C ZBBCZ(M,N)=L(M)*[J*OMEGA*(VAP(M,N)DOT VRCP(M)/2+VAM(M,N)DOT VRC(M,N)/2)
C    -SPOTP(M,N)+SPOTM(M,N)]

```

```

C CV(M)=L(M)*[VEP(M)DOT VRCP(M)/2+VEM(M)DOT VRM(M)]
C WHERE:
C VAP(M,N) AND VAM(M,N) ARE THE VECTOR POTENTIALS DUE TO THE NTH BODY
C BASIS FUNCTION EVALUATED AT THE + AND - CENTROIDS OF THE MTH BODY
C BASIS FUNCTION.
C L(M) IS THE LENGTH OF THE EDGE ASSOCIATED WITH THE MTH BASIS
C FUNCTION TRIANGLES.
C SPOTP(M,N) AND SPOTM(M,N) ARE THE SCALAR POTENTIALS DUE TO THE NTH
C BASIS FUNCTION EVALUATED AT THE + AND - CENTROIDS OF THE MTH BASIS
C FUNCTION TRIANGLES.
C VRCP(M) AND VRM(M) ARE THE COORDINATE VECTORS OF THE + AND -
C CENTROIDS OF THE MTH BASIS FUNCTION TRIANGLES.
C VEP(M) AND VEM(M) ARE THE ELECTRIC FIELD VECTORS EVALUATED AT THE +
C AND - CENTROIDS OF THE MTH BASIS FUNCTION TRIANGLES. N=1,...,NUNKNB
C M=1,...,NUNKNB, WHERE NUNKNB IS THE NUMBER OF BODY UNKNOWNNS.
C
C INPUT:
C DATNOD(I,J) I=1,2,3 ARE THE X,Y,Z COMPONENTS OF THE JTH VERTEX.
C THE JTH EDGE RUNS FROM NCONN(1,J) TO NCONN(2,J). NCONN(3,J) IS THE
C MULTIPLICITY OF THE JTH EDGE(I.E. THE NUMBER OF UNKNOWNNS ASSOCIATED
C WITH THAT EDGE.)
C NBOUND(I,J) I=1,2,3 ARE THE 3 EDGES OF THE JTH FACE.
C NNODES IS THE NUMBER OF NODES ON THE BODY.
C NFACES IS THE NUMBER OF BODY FACES.
C NUNKNT IS THE TOTAL NUMBER OF UNKNOWNNS.
C ALSO THE COMMON FIELDS /WAVE,PARAM,INC/ IN PARTICULAR NFIELD
C COMPUTED IN SOLTN AND /MEDIUM/ COMPUTED IN INDATA ARE PASSED.
C OUTPUT:
C IF NFIELD .LE. 1 BOTH CONTRIBUTIONS TO CZ AND CV ARE FILLED.
C IF NFIELD .GT. 1 ONLY CONTRIBUTIONS TO CV ARE FILLED.
C-----
C PARAMATERS ASSOCIATE WITH JUNCTION PART
  REAL ANG(MNJUN,MNJFACE),VMJUN(3),VSJUN(3),H(3)
  INTEGER NJFACE(MNJUN,MNJFACE),MIFACE(MNJUN),NBJUN(MNJUN),
&     NWJUN(MNJUN),WIRSUM(NWNOD)
C
  INTEGER NCONN(3,NEDGES),NBOUND(3,NFACES),IEDGF(MULT1,NEDGES),
$     IM(3),IS(3),NS(3),NM(3),IGNDP(3)
  DIMENSION INDSUM(NEDGES),DN(3,3),DM(3,3),DMC(3),DLM(3),AL(3)
$     ,SN(3),DLN(3),DATNOD(3,NNODES),TMAT(3,3),RK(3,3),RMK(3),
$     SM(3),DL(3,3)
  COMPLEX CJBEMP,CBJEMP,CJETEMP,CJJEMP
$     ,CZ(NUNKNT,NUNKNT),CV(NUNKNT),HTheta,HPHI,ETHETA,EPHI,
$     ,EX,EY,EZ,EDOTT,CVPB,CSPB,C,CI(3),CA(3),CSJPOT,CSPOT,CFLAG,CARG,
$     ,CTEMP,CETEMP,CP(3),CPP,CMM,CAJ(3),CAJT(3),CAJ2(3),CAJT2(3)

```



```

C
C VARIABLES ASSOCIATED WITH IMAGE PATCH
  $,CII(3),CPPI,CMMI,CAI(3),CAJTI(3),C1(3),CAA(3,3)
C
  REAL LAMBDA,K,MU,IMP
  COMMON/PARA/DL,DET,H,AL
  COMMON/WAVE/OMEGA,LAMBDA,K
  COMMON/MEDIUM/DEG2RAD,EPSLON,MU,IMP,SL,PI
  COMMON/PARAM/THETA,PHI,NFIELD
  COMMON/INC/HTHETA,HPHI,ETHETA,EPHI
  COMMON/GPLANE/NGNDP,IGNDP
  SIZE(X,Y,Z)=SQRT(X*X+Y*Y+Z*Z)
  CVPB=CMPLX(0.,IMP/(4.*PI))
  CSPB=CMPLX(0.,.5/(PI*OMEGA*EPSLON))
  CTHETA=COS(THETA)
  STHETA=SIN(THETA)
  CPHI=COS(PHI)
  SPHI=SIN(PHI)
  EX=ETHETA*CTHETA*CPHI-EPHI*SPHI
  EY=ETHETA*CTHETA*SPHI+EPHI*CPHI
  EZ=-ETHETA*STHETA
C
  DO 999 IFS=1,NFACES
  IIS=1
    IF(IFS.EQ.1.OR.NFIELD.LE.1)THEN
C OBTAIN THE EDGES OF THE SOURCE TRIANGLE
      CALL FACEDG(NFACES,NBOUND,IFS,IS)
C OBTAIN THE VERTICES OF THE SOURCE TRIANGLE
      CALL FACVTX(NCONN,NEDGES,IS,NS)
C OBTAIN THE COORDINATES OF THE SOURCE TRIANGLE'S VERTICIES
      CALL VTXCRD(DATNOD,NNODES,NS,DN)
      DO 2 I=1,3
      DO 2 J=1,3
2      RK(I,J)=K*DN(I,J)
      DO 88 I=1,3
      IP1=MOD(I,3)+1
      IM1=MOD(I+1,3)+1
      DO 77 J=1,3
77      DLN(J)=DN(IM1,J)-DN(IP1,J)
88      SN(I)=SIZE(DLN(1),DLN(2),DLN(3))
C IF NJCT=0, NO JUNCTION CASE
C IF NJCT=1, CHECK IF ANY JUNCTION IN THIS SOURCE TRIANGLE
C IF JSF=0, NO JUNCTION IN THIS SOURCE TRIANGLE
C IF JSF=N, N=1,2,3, FIND OUT ASSOCIATED PARAMETERS
C

```

```

      IF(NJCT.EQ.1) THEN
C TO FIND PARAMATERS ASSOCIATE WITH THIS JUNCTION
      CALL JUNPAR(JSF, DN, NS, IFS, ANGS, VSJUN, JSCOL,
      &      WIRSUM, MNJUN, MNJFACE, NWNOD, ANG, NJFACE, MIFACE, NWJUN, NBJUN)
C ANGS IS THE ANGULAR DISTRIBUTION COEFFICIENCY OF SOURCE TRIANGLE
C SN(JMF) IS THE LENGTH OF OPPOSITE EDGE
      IF(JSF.NE.0)      ANGS=ANGS/SN(JSF)
      ELSE
      JSF=0
      ENDIF
C
      DO 499 IFM=1, NFACES
      IIM=1
      IIV=1
C OBTAIN THE EDGES OF THE MATCH TRIANGLE
      CALL FACEDG(NFACES, NBOUND, IFM, IM)
C OBTAIN THE VERTICES OF THE MATCH TRIANGLE
      CALL FACVTX(NCONN, NEDGES, IM, NM)
C OBTAIN THE COORDINATES OF THE MATCH TRIANGLE'S VERTICIES
      CALL VTXC RD(DATNOD, NNODES, NM, DM)
C CALCULATE THE CENTROID OF THE MATCH TRIANGLE
      DO 4 J=1, 3
      DMC(J)=(DM(1, J)+DM(2, J)+DM(3, J))/3.
C RMK(J) IS THE MATCH POINT
4      RMK(J)=K*DMC(J)
      DO 6 I=1, 3
      DO 6 J=1, 3
6      TMAT(I, J)=(DMC(J)-DM(I, J))/2.
      DO 8 I=1, 3
      IP1=MOD(I, 3)+1
      IM1=MOD(I+1, 3)+1
      DO 7 J=1, 3
7      DLM(J)=DM(IM1, J)-DM(IP1, J)
8      SM(I)=SIZE(DLM(1), DLM(2), DLM(3))
C
C IF NJCT=0, NO JUNCTION CASE
C IF NJCT=1, CHECK IF ANY JUNCTION IN THIS MATCH TRIANGLE
C IF JMF=0, NO JUNCTION IN THIS MATCH TRIANGLE
C IF JMF=N, N=1, 2, 3, FIND OUT ASSOCIATED PARAMETERS
C
      IF(NJCT.EQ.1) THEN
C TO GET PARAMATERS ASSOCIATE WITH THIS JUNCTION
      CALL JUNPAR(JMF, DM, NM, IFM, ANGM, VMJUN, JMROW,
      &      WIRSUM, MNJUN, MNJFACE, NWNOD, ANG, NJFACE, MIFACE, NWJUN, NBJUN)
C ANGM IS THE ANGULAR DISTRIBUTION COEFFICIENCY OF MATCH TRIANGLE

```

```

C
  ELSE
  JMF=0
  ENDIF
C
* COMPUTE QUANTITIES FOR CALCULATION OF THE VECTOR AND SCALAR POTENTIALS
C
C CAA(I)= INTEGRAL OF ( XSI(I)*(EXP(-JKR)-XSING)/R D (XSI(I))
C D(XSI(I+1))) IF XSING=1 THEN +1/(2*AREA)*INTEGRAL OF XSI(I)/R
C D(S') XSI(I),I=1,3 DENOTE ZETA, XSI AND ETA, XSING=0 OR 1
C DL(I,J) IS THE VECTOR FROM THE I+1TH VERTEX TO THE ITH VERTEX
C AL(I) IS THE LENGTH OF THE DL(I,J)
C
  CALL PCTBOD(JSF,RK,RMK,C,CAA,CAJT,0)
  IF(JSF.NE.0) THEN
  ISL=JSF
C SET (+) TO CA (-) TO CAJT (+) TO CSJPOT
  FLAG=AL(JSF)
  CFLAG=CVPB*FLAG
C
C CAA(J),J=1,3 ARE THE X,Y,Z COMPONENTS OF THE VECTOR PONTETIAL
C DUE TO THE TRIANGULAR BASIS FUNCTION
C CAA=(SGN)*MU*AL(I)/4*PI*(CI(I+1)*DL(I-1,J)-CI(I-1)*DL(I+1,J))
C CAJT(J) IS THE VECTOR POTENTIAL DUE TO THE JUNCTION PART OF
C THE JUNCTION BASIS FUNCTION
C CAJ(J) IS THE VECTOR POTENTIAL DUE TO THE JUNCTION
C BASIS FUNCTION
C SET (+) TO CA (-) TO CAJT (+) TO CSJPOT
C
  DO 29 J=1,3
29  CAJ(J)=(CFLAG*CAA(JSF,J)-CVPB*CAJT(J))
C
C SET (+) TO CA (-) TO CAJT (+) TO CSJPOT
C CSJPOT IS THE SCALAR POTENTIAL ASSOCIATED WITH THIS JUNCTION
C
  CSJPOT=CSPB*C*AL(JSF)
  ENDIF
  IV=0
  DO 460 ISL=1,3
  ISS=IS(ISL)
  MULTS=NCONN(3,ISS)
  IF(MULTS.NE.0)THEN
  IF(ISL.EQ.1)THEN
  FLAG=AL(1)
  IF(NCONN(1,IS(1)).EQ.NS(3))FLAG=-FLAG

```

```

ELSEIF(ISL.EQ.2)THEN
  FLAG=AL(2)
  IF(NCONN(1,IS(2)).EQ.NS(1))FLAG=-FLAG
ELSE
  FLAG=AL(3)
  IF(NCONN(1,IS(3)).EQ.NS(2))FLAG=-FLAG
ENDIF
CFLAG=CVPB*FLAG
IP1=MOD(ISL,3)+1
IM1=MOD(IP1,3)+1
C
C CA(J),J=1,3 ARE THE X,Y,Z COMPONENTS OF THE VECTOR PONTETIAL
C CA=(SGN)*MU*AL(I)/4*PI*(CI(I+1)*DL(I-1,J)-CI(I-1)*DL(I+1,J))
C
DO 9 J=1,3
9   CA(J)=CFLAG*CAA(ISL,J)
C
C CSPOT IS THE SCALAR POTENTIAL
C
CSPOT=CSPB*FLAG*C
C
C FOR AN EDGE WITH MULTIPLICITY MULT, THE LOWEST NUMBERED FACE
C CONTRIBUTES TO MULT BASIS FUNCTIONS ASSOCIATED WITH THAT EDGE,
C WHILE EACH REMAINING FACE CONTRIBUTES TO ONE BASIS FUNCTION ASSOCIATED
C WITH THAT EDGE.
CALL EDGMUL(IEDGF,MULT1,NEDGES,MULTS,ISS,IFS,JS1,JS2)
INDS=INDSUM(ISS)
DO 450 JS=JS1,JS2
IV=IV+1
ICOL=INDS+JS
C
C ZJB : NO JUNCTION IN THIS SOURCE TRIANGLE,
C HAS JUNCTION IN THIS MATCH TRIANGLE
C
IF(JMF.NE.0) THEN
C VECTOR POTENTIAL DOT WITH TESTING VECTOR
CJBEMP=CA(1)*VMJUN(1)+CA(2)*VMJUN(2)+CA(3)*VMJUN(3)
C SET + TO VECTOR POTENTIAL AND - TO SCALAR POTENTIAL
IF(NFIELD.EQ.1)CZ(JMROW,ICOL)=CZ(JMROW,ICOL)
$ +(CJBEMP-CSPOT)*ANGM
ENDIF
C
DO 100 IML=1,3
IF(NCONN(3,IM(IML)).GT.0)THEN
T1=TMAT(IML,1)

```

```

T2=TMAT(IML,2)
T3=TMAT(IML,3)
FLAG=1.
IF(IML.EQ.1)THEN
  IF(NCONN(1,IM(1)).EQ.NM(3))FLAG=-1.
ELSEIF(IML.EQ.2)THEN
  IF(NCONN(1,IM(2)).EQ.NM(1))FLAG=-1.
ELSE
  IF(NCONN(1,IM(3)).EQ.NM(2))FLAG=-1.
ENDIF

C
IF(IFS.EQ.1 .AND. IV.EQ.1)THEN
  ARGMT=DMC(1)*STHETA*CPHI+DMC(2)*STHETA*SPHI
  $      +DMC(3)*CTHETA
  CARG=CMPLX(0.,K*ARGMT)
  EDOTT=EX*T1+EY*T2+EZ*T3
  CETEMP=EDOTT*CEXP(CARG)
ENDIF
IF(NFIELD.EQ.1)CTEMP=CA(1)*T1+CA(2)*T2+CA(3)*T3
MULTM=NCONN(3,IM(IML))
IF(MULTM.GT.0)THEN
  IMM=IM(IML)
C FOR AN EDGE WITH MULTIPLICITY MULT, THE LOWEST NUMBERED FACE
C CONTRIBUTES TO MULT BASIS FUNCTIONS ASSOCIATED WITH THAT EDGE,
C WHILE EACH REMAINING FACE CONTRIBUTES TO ONE BASIS FUNCTION ASSOCIATED
C WITH THAT EDGE.
  CALL EDGMUL(IEDGF,MULT1,NEDGES,MULTM,IMM,IFM,
  $JM1,JM2)
  INDM=INDSUM(IMM)
  DO 50 JM=JM1,JM2
    IROW=INDM+JM
    IF(NFIELD.EQ.1)CZ(IROW,ICOL)=CZ(IROW,ICOL)
  $      +FLAG*(CTEMP-CSPOT)*SM(IML)
    IF(IFS.EQ.1 .AND. IV.EQ.1)CV(IROW)=CV(IROW)+
  $FLAG*SM(IML)*CETEMP
C
C ZBJ : HAS JUNCTION IN THIS SOURCE TRIANGLE,
C NO JUNCTION IN THIS MATCH TRIANGLE
C
IF(JSF.NE.0 .AND. IV.LE.1) THEN
  CBJEMP=CAJ(1)*T1+CAJ(2)*T2+CAJ(3)*T3
  IF(NFIELD.EQ.1)CZ(IROW,JSCOL)=CZ(IROW,JSCOL)
  $      +FLAG*(CBJEMP-CSJPOT)*SM(IML)*ANGS
ENDIF
C

```

```

C VJ: FORCE AT JUNCTION POINT
C
  IF(JMF.NE.O.AND.IV.EQ.1)THEN
    EDOTT=EX*VMJUN(1)+EY*VMJUN(2)+EZ*VMJUN(3)
    CJETEMP=EDOTT*CEXP(CARG)
    IF(IFS.EQ.1.AND.IV.EQ.1)CV(JMROW)=CV(JMROW)-
$CJETEMP*ANGM
  ENDIF
50      CONTINUE
      ENDIF
      CONTINUE
100     CONTINUE
450     CONTINUE
      ENDIF
460     CONTINUE
C
C ZJJ : HAS JUNCTION IN THIS SOURCE TRIANGLE,
C       HAS JUNCTION IN THIS MATCH TRIANGLE
C
  IF(JSF.NE.O.AND.JMF.NE.O)THEN
C VECTOR POTENTIAL DOT WITH TESTING VECTOR
  CJJEMP=CAJ(1)*VMJUN(1)+CAJ(2)*VMJUN(2)+CAJ(3)*VMJUN(3)
C SET + TO VECTOR POTENTIAL AND - TO SCALAR POTENTIAL
  IF(NFIELD.EQ.1)CZ(JMROW,JSCOL)=CZ(JMROW,JSCOL)
  $ +(CJJEMP-CSJPOT)*ANGM*ANGS
  ENDIF
C
499     CONTINUE
      ENDIF
999     CONTINUE
      RETURN
      END

```

A.44 ZWB

```

C=====
SUBROUTINE ZWB(DATNOD,NCONN,NBOUND,IEDGF,MULT1,NNODES,NEDGES,
$ NFACES,NUNKNT,CZ,INDSUM,NWNOD,NWSEG,WNODE,NSEGC,WSEGH,
$ MXWMLT,MULTW,INSEG,WIRSUM,NUNKNB,
& NJCT,MNJUN,MNJFACE,ANG,NJFACE,MIFACE,NWJUN,NBJUN)
C-----
C THIS ROUTINE FILL OUT THE MATRIX ELEMENTS OF ZWB WHICH IS SOURCE POINT
C ON THE BODY AND MATCH POINT ON THE WIRE

```

```

C-----
REAL ANG(MNJUN,MNJFACE),XS(3),YS(3),ZS(3),VSJUN(3)
& ,DATNOD(3,NNODES),TMAT(3,3),RK(3,3),RMK(3),SM(3)
& ,LAMBDA,K,MU,IMP
INTEGER NJFACE(MNJUN,MNJFACE),MIFACE(MNJUN),NBUN(MNJUN),
& ,NWJUN(MNJUN),NS(3),IGNDP(3)
& ,NCONN(3,NEDGES),NBOUND(3,NFACES),IEDGF(MULT1,NEDGES),
$ IM(3),IS(3),MULTW(NWNOD),INSEG(MXWMLT+1,NWNOD),WIRSUM(NWNOD)
DIMENSION WNODE(3,NWNOD),NSEGC(2,NWSEG),WSEGH(3,NWSEG),T(3)
& ,INDSUM(NEDGES),AL(3),DL(3,3),H(3),
& ,DN(3,3),DLN(3),SN(3),DM(3,3),DMC(3),DLM(3)
COMPLEX CWJEMP,CZ(NUNKNT,NUNKNT),HTheta,HPhi,ETHeta,EPhi,
$ EX,EY,EZ,EDOTT,CVPB,CSPB,C,CI(3),CAX,CAY,CAZ,CSPOT,CFLAG,CARG,
$ CTEMP,CETEMP,CP(3),CPP,CMM,CAJ(3),CAJT(3),CA(3),CSJPOT

C
C VARIABLES ASSOCIATED WITH IMAGE PATCH
$ ,CII(3),CPPI,CMMI,CAI(3),CAJTI(3),CAA(3,3),CX,CJ

C
LOGICAL LOWS,LOWM,PLUS

C
COMMON/PARA/DL,DET,H,AL
COMMON/GPLANE/NGNDP,IGNDP

C
COMMON/WAVE/OMEGA,LAMBDA,K
COMMON/MEDIUM/DEG2RAD,EPSLON,MU,IMP,SL,PI
COMMON/PARAM/THETA,PHI,NFIELD
COMMON/INC/HTheta,HPhi,ETHeta,EPhi
C THE FOLLOWING LINE IS A STATEMENT FUNCTION.
SIZE(X,Y,Z)=SQRT(X*X+Y*Y+Z*Z)
C SET CONSTANTS.
CVPB=CMPLX(0.,IMP/(4.*PI))
CSPB=CMPLX(0.,.5/(PI*OMEGA*EPSLON))
C LOOP OVER THE FACE NUMBERS OF THE SOURCE TRIANGLES.
DO 999 IFS=1,NFACES
IIS=1
IF(IFS.EQ.1.OR.NFIELD.LE.1)THEN
C OBTAIN THE EDGES OF THE SOURCE TRIANGLE.
C OBTAIN THE COORDINATES OF THE SOURCE TRIANGLE'S VERTICIES.
C STORE IN RK(J,I) I=1,2,3 K*THE X,Y,Z COORDINATES OF THE JTH VERTEX
C OF THE SOURCE TRIANGLE.
C
C OBTAIN THE EDGES OF THE SOURCE TRIANGLE
CALL FACEDG(NFACES,NBOUND,IFS,IS)
C OBTAIN THE VERTICES OF THE SOURCE TRIANGLE
CALL FACVTX(NCONN,NEDGES,IS,NS)

```

```

C OBTAIN THE COORDINATES OF THE SOURCE TRIANGLE'S VERTICIES
  CALL VTXC RD(DATNOD,NNODES,NS, DN)
  DO 2 I=1,3
  DO 2 J=1,3
2   RK(I,J)=K*DN(I,J)
     DO 88 I=1,3
     IP1=MOD(I,3)+1
     IM1=MOD(I+1,3)+1
     DO 77 J=1,3
77    DLN(J)=DN(IM1,J)-DN(IP1,J)
88    SN(I)=SIZE(DLN(1),DLN(2),DLN(3))
C
C IF NJCT=0, NO JUNCTION CASE
C IF NJCT=1, CHECK IF ANY JUNCTION IN THIS SOURCE TRIANGLE
C IF JSF=0, NO JUNCTION IN THIS SOURCE TRIANGLE
C IF JSF=N, N=1,2,3, FIND OUT ASSOCIATED PARAMETERS
C
  IF(NJCT.EQ.1) THEN
C TO GET PARAMATERS ASSOCIATE WITH THIS JUNCTION
  CALL JUNPAR(JSF, DN, NS, IFS, ANGS, V SJUN, JSCOL,
    & WIRSUM, MNJUN, MNJFACE, NWNOD, ANG, NJFACE, MIFACE, NWJUN, NBJUN)
C ANGS IS THE ANGULAR DISTRIBUTION COEFFICIENCY OF MATCH TRIANGLE
C SN(JMF) IS THE LENGTH OF OPPOSITE EDGE
  IF(JSF.NE.0) ANGS=ANGS/SN(JSF)
  ELSE
  JSF=0
  ENDIF
C
  * LOOP OVER THE MATCH SEGMENTS.
C
  DO 499 NM=1, NWSEG
C
  * OBTAIN COORDINATES OF TEST VECTOR AND MATCH SEGMENT CENTROID TIMES K.
C
  NMF=NSEGC(1, NM)
  NMT=NSEGC(2, NM)
  DO 5 J=1,3
  T(J)=.5*(WNODE(J, NMT)-WNODE(J, NMF))
  RMK(J)=K*WSEGH(J, NM)
5   CONTINUE
C
  * COMPUTE QUANTITIES FOR CALCULATION OF THE VECTOR AND SCALAR POTENTIALS
C
C CAA(I)= INTEGRAL OF ( XSI(I)*(EXP(-JKR)-XSING)/R D (XSI(I))
C D(XSI(I+1))) IF XSING=1 THEN +1/(2*AREA)*INTEGRAL OF XSI(I)/R

```



```

C D(S') XSI(I),I=1,3 DENOTE ZETA, XSI AND ETA, XSING=0 OR 1
C DL(I,J) IS THE VECTOR FROM THE I+1TH VERTEX TO THE ITH VERTEX
C AL(I) IS THE LENGTH OF THE DL(I,J)
C
      CALL POTBOD(JSF,RK,RMK,C,CAA,CAJT,0)
      IF(JSF.NE.0) THEN
        ISL=JSF
C SET (+) TO CA (-) TO CAJT (+) TO CSJPOT
        FLAG=AL(JSF)
          CFLAG=CVPB*FLAG
C
C CAA(J),J=1,3 ARE THE X,Y,Z COMPONENTS OF THE VECTOR PONTETIAL
C DUE TO THE TRIANGULAR BASIS FUNCTION
C CAA=(SGN)*MU*AL(I)/4*PI*(CI(I+1)*DL(I-1,J)-CI(I-1)*DL(I+1,J))
C CAJT(J) IS THE VECTOR POTENTIAL DUE TO THE JUNCTION PART OF
C THE JUNCTION BASIS FUNCTION
C CAJ(J) IS THE VECTOR POTENTIAL DUE TO THE JUNCTION
C BASIS FUNCTION
      DO 29 J=1,3
29      CAJ(J)=(CFLAG*CAA(JSF,J)-CVPB*CAJT(J))
C COMPUTE CSJPOT THE SCALAR POTENTIAL ASSOCIATED WITH JUNCTION
      CSJPOT=CSPB*C*AL(JSF)
      ENDIF
      IV=0
      DO 460 ISL=1,3
        ISS=IS(ISL)
        MULTS=NCONN(3,ISS)
        IF(MULTS.NE.0)THEN
          IF(ISL.EQ.1)THEN
            FLAG=AL(1)
            IF(NCONN(1,IS(1)).EQ.NS(3))FLAG=-FLAG
          ELSEIF(ISL.EQ.2)THEN
            FLAG=AL(2)
            IF(NCONN(1,IS(2)).EQ.NS(1))FLAG=-FLAG
          ELSE
            FLAG=AL(3)
            IF(NCONN(1,IS(3)).EQ.NS(2))FLAG=-FLAG
          ENDIF
          CFLAG=CVPB*FLAG
        IP1=MOD(ISL,3)+1
        IM1=MOD(ISL+1,3)+1
C
C CA(J),J=1,3 ARE THE X,Y,Z COMPONENTS OF THE VECTOR PONTETIAL
C CA=(SGN)*MU*AL(I)/4*PI*(CI(I+1)*DL(I-1,J)-CI(I-1)*DL(I+1,J))
C

```

```

          DO 9 J=1,3
9         CA(J)=CFLAG*CAA(ISL,J)
C
C COMPUTE CSPOT THE SCALAR POTENTIAL.
          CSPOT=CSPB*FLAG*C
C FOR AN EDGE WITH MULTIPLICITY MULT, THE LOWEST NUMBERED FACE
C CONTRIBUTES TO MULT BASIS FUNCTIONS ASSOCIATED WITH THAT EDGE,
C WHILE EACH REMAINING FACE CONTRIBUTES TO ONE BASIS FUNCTION ASSOCIATED
C WITH THAT EDGE.
          CALL EDMUL(IEDGF,MULT1,NEDGES,MULTS,ISS,IFS,JS1,JS2)
          INDS=INDSUM(ISS)
          DO 450 JS=JS1,JS2
            IV=IV+1
            ICOL=INDS+JS
C COMPUTE QUANTITIES ASSOCIATED WITH THE INCIDENT FIELD.
            CTEMP=(CA(1)*T(1)+CA(2)*T(2)+CA(3)*T(3))
C
* LOOP OVER THE NODES ATTACHED TO THE MATCH SEGMENT.
C
          DO 250 JMN=1,2
            NODEM=NSEGC(JMN,NM)
            MULTM=MULTW(NODEM)
            IF(MULTM.GT.0)THEN
C
* COMPUTE ROW INDEX FOR SOURCE SEGMENT.
C
              INDR=WIRSUM(NODEM)
C
* DETERMINE WHETHER SEGMENT NM IS THE LOWEST SEGMENT ATTACHED
* TO SEGMENT NM AT NODE NODEM.LOOP OVER THE NUMBER OF SEGMENTS ATTACHED
* TO SEGMENT NM AT NODE NODEM.
C
              IF(NODEM.EQ.NWJUN(MNJUN)) THEN
                JM1=1
                JM2=1
                LOWM=.FALSE.
              ELSE
                CALL NODMUL(INSEG,MXWMLT,NWNOD,NODEM,MULTM,NM,
                >                JM1,JM2,LOWM)
              ENDIF
            DO 200 JM=JM1,JM2
C
* DETERMINE WHETHER SEGMENT IS A PLUS OR MINUS SEGMENT
C
              IF(NODEM.EQ.NSEGC(2,NM))THEN

```

```

                SGN1=1.
            ELSE
                SGN1=-1.
            ENDIF
            IF(LOWM)THEN
C
* CASE: SEGMENT NM IS THE LOWEST NUMBERED SEGMENT ATTACHED TO NODE
* NODEM. DIRECTION OF THE BASIS FUNCTION IS DETERMINED BY THE ATTACHED
* SEGMENT. OBTAIN ATTACHED SEGMENT.
C
                NSEGAM=INSEG(JM+1,NODEM)
                IF(NSEGC(2,NM).EQ.NSEGC(1,NSEGAM).OR.
>                 NSEGC(1,NM).EQ.NSEGC(2,NSEGAM))THEN
                    SGN=1.
                ELSE
                    SGN=-1.
                ENDIF
            ELSE
                SGN=1.
            ENDIF
            IF(NODEM.EQ.NWJUN(MNJUN))THEN
                IF(NODEM.EQ.NSEGC(2,NM))THEN
                    SGN=-1.
                    SGN1=1.
                ELSE
                    SGN=1.
                    SGN1=-1.
                ENDIF
            ENDIF

                IROW=INDR+JM
                IF(NFIELD.EQ.1)CZ(IROW,ICOL)=CZ(IROW,ICOL)
$                +SGN*(CTEMP-CSPOT*SGN1)
C
C ZWJ: HAS JUNCTION IN THIS SOURCE TRIANGLE
C
                IF(JSF.NE.0.AND.IV.LE.1)THEN
C VECTOR POTENTIAL DOT WITH TESTING VECTOR
                CWJEMP=CAJ(1)*T(1)+CAJ(2)*T(2)+CAJ(3)*T(3)
                IF(NFIELD.EQ.1)CZ(IROW,JSCOL)=CZ(IROW,JSCOL)
$                +SGN*(CWJEMP-CSJPOT*SGN1)*ANGS
                ENDIF
C
200                CONTINUE
                ENDIF
250                CONTINUE

```

```

450         CONTINUE
           ENDIF
460         CONTINUE
499         CONTINUE
           ENDIF
999        CONTINUE
           RETURN
           END

```

A.45 PATTEN

```

C=====
C          SUBROUTINE PATTEN(DATNOD,NCONN,NBOUND,IEDGF,MULT1,NNODES,WIRSUM,
C          $NEDGES,NFACES,NUNKNT,CJ,INDSUM,IPAT,NJCT,MNJUN,MNJFACE,NWNOD,
C          $ANG,NJFACE,MIFACE,NWJUN,NBJUN,MXWMLT,NWSEG,NUNKNB,WNODE,MULTW,
C          $NSEGC,WSEGH,RAD,INSEG)
C-----
C INPUT:
C DATNOD(I,J) I=1,2,3 ARE THE X,Y,Z COMPONENTS OF THE JTH VERTEX.
C THE JTH EDGE RUNS FROM NCONN(1,J) TO NCONN(2,J). NCONN(3,J) IS THE
C MULTIPLICITY OF THE JTH EDGE(I.E. THE NUMBER OF UNKNOWNNS ASSOCIATED
C WITH THAT EDGE.)
C NBOUND(I,J) I=1,2,3 ARE THE 3 EDGES OF THE JTH FACE.
C NNODES IS THE NUMBER OF NODES ON THE BODY.
C NEDGES IS THE NUMBER OF EDGES.
C NFACES IS THE NUMBER OF BODY FACES.
C NUNKNT IS THE TOTAL NUMBER OF UNKNOWNNS.
C CJ IS A COMPLEX ARRAY CONTAINING THE CURRENT AMPLITUDES
C ALSO THE COMMON FIELDS /MEDIUM,FINDIF,WAVE/ ARE COMPUTED IN INDATA
C AND IN SOLTN.
C
C OUTPUT:
C ES,HS ARE ARRAYS CONTAINING THE FIELD VALUES
C AT THE OBSERVATION POINTS.
C
C          REAL ANG(MNJUN,MNJFACE),VSJUN(3),H(3)
C          INTEGER NJFACE(MNJUN,MNJFACE),MIFACE(MNJUN),NBJUN(MNJUN),
C          &          NWJUN(MNJUN),WIRSUM(NWNOD)
C
C          COMPLEX CV(100)
C          DIMENSION WNODE(3,NWNOD),MULTW(NWNOD),NSEGC(2,NWSEG),
C          >          WSEGH(3,NWSEG),RAD(NWSEG),INSEG(MXWMLT+1,NWNOD)
C

```

```

    INTEGER NCONN(3,NEDGES),NBOUND(3,NFACES),IEDGF(MULT1,NEDGES),
    $IS(3),INDSUM(NEDGES),NS(3)
    DIMENSION DATNOD(3,NNODES),TMAT(3,3),RK(3,3),RMK(3),RFLD(3,5),
    $DK(3)
    COMPLEX ETH(100,100),EPH(100,100),HSQR,RCS(10,10),SIGMA
    $,ETHSQR(10,10),EPHSQR(10,10)
    COMPLEX ES(3),HS(3),CJ(NUNKNT),CH,CAXJ,CAYJ,CAZJ,CA(3),CI(3),
    $CVPB,CSPB,C,CVEC(3),CAX,CAY,CAZ,CSPOT,CFLAG,
    $CTEMP,CSPOTJ,CP(3)
    COMPLEX HTHETA,HPhi,ETHETA,EPhi
    REAL LAMBDA,K,MU,IMP
    REAL MAGH(3,3),LMINK,LMAXK,HHAT(3,3),DN(3,3)
    REAL L(3,3),LHAT(3,3),NHAT(3),MAGL(3)
    COMMON/PARA/L,MAGL,AREAX2,H
    COMMON/WAVE/OMEGA,LAMBDA,K
    COMMON/FINDIF/NNFLD,DX,DY,DZ
    COMMON/MEDIUM/DEG2RAD,EPSLON,MU,IMP,SL,PI
    COMMON/PAT/PHI1,PHI2,NPhi,THETA1,THETA2,NTHETA
    COMMON/INC/HTHETA,HPhi,ETHETA,EPhi
C SET CONSTANTS.
    CVPB=CMPLX(0.,IMP/(4.*PI))
    WRITE(10,101)
101  FORMAT(X,/25X,'FAR FIELD PATTERN',/)
    WRITE(10,102)
102  FORMAT(X,'ITHETA',2X,'THETA',2X,'IPHI',2X,'PHI',3X,
    &'ETH(ITHETA,IPHI)',5X,'EPH(ITHETA,IPHI)',/)
C LOOP OVER FIELD OBSERVATION POINTS.
    IF(NPhi.EQ.1)THEN
        DPhi=0
    ELSE
        DPhi=(PHI2-PHI1)/FLOAT(NPhi-1)
    ENDIF
    IF(NTHETA.EQ.1)THEN
        DTHETA=0
    ELSE
        DTHETA=(THETA2-THETA1)/FLOAT(NTHETA-1)
    ENDIF
    DO 999 ITHETA=1,NTHETA
        THETA=THETA1+(ITHETA-1)*DTHETA
    DO 998 IPHI=1,NPhi
        PHI=PHI1+(IPHI-1)*DPhi
        RMK(1)=SIND(THETA)*COSD(PHI)
        RMK(2)=SIND(THETA)*SIND(PHI)
        RMK(3)=COSD(THETA)
        ETH(ITHETA,IPHI)=(0.,0.)

```

```

      EPH(ITHETA,IPHI)=(0.,0.)
      CALL BPATTN(IPAT,DATNOD,NCONN,NBOUND,IEDGF,MULT1,NNODES,NEDGES,
$          NFACES,NUNKNT,CJ,CV,INDSUM,NJCT,MNJUN,MNJFACE,NWNOD,WIRSUM,
$          ANG,NJFACE,MIFACE,NWJUN,NBJUN,RMK,ETH,EPH,ITHETA,IPHI,THETA,
$          PHI)
C-----
      IF(NJCT.GE.0)THEN
      CALL WPATTN(IPAT,MXWMLT,NWNOD,NWSEG,NUNKNB,NUNKNT,WNODE,MULTW,
$NSEGC,WSEGH,RAD,INSEG,CJ,CV,WIRSUM,NWJUN,MNJUN,RMK,ETH,EPH,ITHETA,
$      IPHI,THETA,PHI)
      ENDIF
C-----
C COMPUTE RADAR CROSS SECTION
C
C      ETHSQR(ITHETA,IPHI)=ETH(ITHETA,IPHI)*CONJG(ETH(ITHETA,IPHI))
C      EPHSQR(ITHETA,IPHI)=EPH(ITHETA,IPHI)*CONJG(EPH(ITHETA,IPHI))
C      SIGMA=4.*PI*(ETHSQR(ITHETA,IPHI)+EPHSQR(ITHETA,IPHI))/
C      1      (IMP*IMP)
C      HSQR=HTHETA*CONJG(HTHETA)+HPHI*CONJG(HPHI)
C      RCS(ITHETA,IPHI)=SIGMA/HSQR
C
C      WRITE(15,*)RCS(ITHETA,IPHI)
C 1001  FORMAT(1X,2I6,2F10.5)
998  CONTINUE
999  CONTINUE
      RETURN
      END

```

A.46 BPATTN

```

C=====
      SUBROUTINE BPATTN(IPAT,DATNOD,NCONN,NBOUND,IEDGF,MULT1,NNODES,
$NEDGES,NFACES,NUNKNT,CJ,CV,INDSUM,NJCT,MNJUN,MNJFACE,NWNOD,WIRSUM,
$      ANG,NJFACE,MIFACE,NWJUN,NBJUN,RMK,ETH,EPH,ITHETA,IPHI,THETA,
$      PHI)
C-----
      COMPLEX ETH(100,100),EPH(100,100),CAXJ,CAYJ,CAZJ
      REAL ANG(MNJUN,MNJFACE),VMJUN(3),VSJUN(3),H(3)
      INTEGER NJFACE(MNJUN,MNJFACE),MIFACE(MNJUN),NBUN(MNJUN),
&      NWJUN(MNJUN),WIRSUM(NWNOD)
C
      INTEGER NCONN(3,NEDGES),NBOUND(3,NFACES),IEDGF(MULT1,NEDGES),
$      IM(3),IS(3),NS(3),NM(3),IGNDP(3)

```

```

DIMENSION INDSUM(NEDGES),DN(3,3),DM(3,3),DMC(3),DLM(3),AL(3)
$      ,SN(3),DLN(3),DATNOD(3,NNODES),TMAT(3,3),RK(3,3),RMK(3),
$      SM(3),DL(3,3)
COMPLEX CJ(NUNKNT),CV(NUNKNT),CVPB,C,CI(3),CA(3),CFLAG,
$CAJ(3),CAJT(3)
C
C VARIABLES ASSOCIATED WITH IMAGE PATCH
$ ,CAA(3,3)
C
REAL LAMBDA,K,MU,IMP
COMMON/PARA/DL,DET,H,AL
COMMON/WAVE/OMEGA,LAMBDA,K
COMMON/MEDIUM/DEG2RAD,EPSLON,MU,IMP,SL,PI
COMMON/INC/HTHETA,HPHI,ETHETA,EPHI
COMMON/GPLANE/NGNDP,IGNDP
SIZE(X,Y,Z)=SQRT(X*X+Y*Y+Z*Z)
CVPB=CMPLX(0.,IMP/(4.*PI))
C
DO 999 IFS=1,NFACES
IIS=1
C OBTAIN THE EDGES OF THE SOURCE TRIANGLE
CALL FACEDG(NFACES,NBOUND,IFS,IS)
C OBTAIN THE VERTICES OF THE SOURCE TRIANGLE
CALL FACVTX(NCONN,NEDGES,IS,NS)
C OBTAIN THE COORDINATES OF THE SOURCE TRIANGLE'S VERTICIES
CALL VTXCRD(DATNOD,NNODES,NS,DN)
DO 2 I=1,3
DO 2 J=1,3
2   RK(I,J)=K*DN(I,J)
DO 88 I=1,3
IP1=MOD(I,3)+1
IM1=MOD(I+1,3)+1
DO 77 J=1,3
77  DLN(J)=DN(IM1,J)-DN(IP1,J)
88  SN(I)=SIZE(DLN(1),DLN(2),DLN(3))
C IF NJCT=0, NO JUNCTION CASE
C IF NJCT=1, CHECK IF ANY JUNCTION IN THIS SOURCE TRIANGLE
C IF JSF=0, NO JUNCTION IN THIS SOURCE TRIANGLE
C IF JSF=N, N=1,2,3, FIND OUT ASSOCIATED PARAMETERS
C
IF(NJCT.EQ.1) THEN
C TO FIND PARAMATERS ASSOCIATE WITH THIS JUNCTION
CALL JUNPAR(JSF,DN,NS,IFS,ANGS,VSJUN,JSCOL,
&      WIRSUM,MNJUN,MNJFACE,NWNOD,ANG,NJFACE,MIFACE,NWJUN,NBJUN)
C ANGS IS THE ANGULAR DISTRIBUTION COEFFICIENCY OF SOURCE TRIANGLE

```

```

C SN(JMF) IS THE LENGTH OF OPPOSITE EDGE
  IF(JSF.NE.0)      ANGS=ANGS/SN(JSF)
  ELSE
  JSF=0
  ENDIF

C
C
* COMPUTE QUANTITIES FOR CALCULATION OF THE VECTOR AND SCALAR POTENTIALS
C
C CAA(I)= INTEGRAL OF ( XSI(I)*(EXP(-JKR)-XSING)/R D (XSI(I))
C D(XSI(I+1))) IF XSING=1 THEN +1/(2*AREA)*INTEGRAL OF XSI(I)/R
C D(S') XSI(I),I=1,3 DENOTE ZETA, XSI AND ETA, XSING=0 OR 1
C DL(I,J) IS THE VECTOR FROM THE I+1TH VERTEX TO THE ITH VERTEX
C AL(I) IS THE LENGTH OF THE DL(I,J)
C
      CALL POTBOD(JSF,RK,RMK,C,CAA,CAJT,IPAT)
      IF(JSF.NE.0) THEN
      ISL=JSF
C SET (+) TO CA (-) TO CAJT (+) TO CSJPOT
      FLAG=AL(JSF)
      CFLAG=CVPB*FLAG

C
C CAA(J),J=1,3 ARE THE X,Y,Z COMPONENTS OF THE VECTOR PONTETIAL
C DUE TO THE TRIANGULAR BASIS FUNCTION
C CAA=(SGN)*MU*AL(I)/4*PI*(CI(I+1)*DL(I-1,J)-CI(I-1)*DL(I+1,J))
C CAJT(J) IS THE VECTOR POTENTIAL DUE TO THE JUNCTION PART OF
C THE JUNCTION BASIS FUNCTION
C CAJ(J) IS THE VECTOR POTENTIAL DUE TO THE JUNCTION
C BASIS FUNCTION
C SET (+) TO CA (-) TO CAJT (+) TO CSJPOT
C
      DO 29 J=1,3
29      CAJ(J)=(CFLAG*CAA(JSF,J)-CVPB*CAJT(J))
C
      ENDIF
      IV=0
      DO 460 ISL=1,3
      ISS=IS(ISL)
      MULTS=NCONN(3,ISS)
      IF(MULTS.NE.0)THEN
      IF(ISL.EQ.1)THEN
      FLAG=AL(1)
      IF(NCONN(1,IS(1)).EQ.NS(3))FLAG=-FLAG
      ELSEIF(ISL.EQ.2)THEN
      FLAG=AL(2)

```



```

                IF(NCONN(1,IS(2)).EQ.NS(1))FLAG=-FLAG
            ELSE
                FLAG=AL(3)
                IF(NCONN(1,IS(3)).EQ.NS(2))FLAG=-FLAG
            ENDIF
            CFLAG=CVPB*FLAG
            IP1=MOD(ISL,3)+1
            IM1=MOD(IP1,3)+1
C
C CA(J),J=1,3 ARE THE X,Y,Z COMPONENTS OF THE VECTOR PONTETIAL
C CA=(SGN)*MU*AL(I)/4*PI*(CI(I+1)*DL(I-1,J)-CI(I-1)*DL(I+1,J))
C
            DO 9 J=1,3
9             CA(J)=CFLAG*CAA(ISL,J)
C
C FOR AN EDGE WITH MULTIPLICITY MULT, THE LOWEST NUMBERED FACE
C CONTRIBUTES TO MULT BASIS FUNCTIONS ASSOCIATED WITH THAT EDGE,
C WHILE EACH REMAINING FACE CONTRIBUTES TO ONE BASIS FUNCTION ASSOCIATED
C WITH THAT EDGE.
                IF(MULTS.NE.0)THEN
                    CALL EDGMUL(IEDGF,MULT1,NEDGES,MULTS,ISS,IFS,JS1,JS2)
                    INDS=INDSUM(ISS)
                    DO 450 JS=JS1,JS2
                        IV=IV+1
                        ICOL=INDS+JS
C
C
                        CAXJ=CA(1)*CJ(ICOL)/K
                        CAYJ=CA(2)*CJ(ICOL)/K
                        CAZJ=CA(3)*CJ(ICOL)/K
                        CAZJ=CA(3)*CJ(ICOL)/K
C
C IT HAS JUNCTION IN THIS SOURCE TRIANGLE,
                IF(JSF.NE.0.AND.IV.LE.1) THEN
                    CAXJ=CAXJ+CAJ(1)*CJ(JSCOL)/K
                    CAYJ=CAYJ+CAJ(2)*CJ(JSCOL)/K
                    CAZJ=CAZJ+CAJ(3)*CJ(JSCOL)/K
                ENDIF
                ETH(ITHETA,IPHI)=ETH(ITHETA,IPHI)
1                 -CAXJ*COSD(THETA)*COSD(PHI)
2                 -CAYJ*COSD(THETA)*SIND(PHI)
3                 +CAZJ*SIND(THETA)
                EPH(ITHETA,IPHI)=EPH(ITHETA,IPHI)
1                 +CAXJ*SIND(PHI)
2                 -CAYJ*COSD(PHI)
C

```

```

450          CONTINUE
          ENDIF
          ENDIF
460          CONTINUE
999 CONTINUE
      IF(NJCT.LT.0)THEN
      WRITE(10,1000)ITHETA,THETA,IPHI,PHI,ETH(ITHETA,IPHI),EPH(ITHETA,
1IPHI)
1000 FORMAT(I5,F8.2,I5,F8.2,4(1PE10.2))
          ENDIF
      RETURN
      END

```

A.47 WPATTN

```

C-----
      SUBROUTINE WPATTN(IPAT,MXWMLT,NWNOD,NWSEG,NUNKNB,NUNKNT,WNCDE,
      $MULTW,NSEGC,WSEGH,RAD,INSEG,CJ,CV,WIRSUM,NWJUN,MNJUN,RM,ETH,EPH,
      $ITHETA,IPHI,THETA,PHI)
C-----
      COMPLEX ETH(100,100),EPH(100,100),CAXJ,CAYJ,CAZJ
      DIMENSION WNODE(3,NWNOD),MULTW(NWNOD),NSEGC(2,NWSEG),
      >          WSEGH(3,NWSEG),RAD(NWSEG),INSEG(MXWMLT+1,NWNOD),
      >          RMK(3),SH(3),RSK(3),T(3),RM(3)
      COMPLEX CJ(NUNKNT),CV(NUNKNT),
      >          CSPW,HTHETA,HPHI,ETHETA,EPHI,EX,EY,EZ,ANS(3),
      >          SPOT,VAP(3),VAM(3),SSPOT,VA(3)
      >          ,CKRED,CKTOT,CKSELF,CANS
      INTEGER WIRSUM(NWNOD),NWJUN(MNJUN)
      REAL LAMBDA,K,MU,IMP
      EXTERNAL CKRED,CKTOT,CKSELF
      LOGICAL LOWS,LOWM,PLUS
      SAVE /WAVE/,/PARAM/,/INC/,/MEDIUM/,/WKERNL/,/WIRE/,/WKER/
      COMMON/POT/NM,NS,DEL,DELS,DELH,DELRH,RVPW,CSPW,RADMKS
      COMMON/WAVE/OMEGA,LAMBDA,K
      COMMON/MEDIUM/DEG2RAD,EPSLON,MU,IMP,SL,PI
      COMMON/INC/HTHETA,HPHI,ETHETA,EPHI
      COMMON/WKERNL/RSK,SH,RMK,RADSK,RADSKS
      COMMON/WKER/DPAR,RHO,RHOPR,RHOPRS,RHOMRS
      COMMON/WIRE/IQUADW
      COMMON/WIRSLF/IQWS
C
* SET CONSTANTS.

```

```

C
  IQUAD=IQUADW
  IQUAWS=IQWS
  PI4=4.*PI
  RVPW=MU/PI4
  CSPW=CMPLX(0.,1./((OMEGA*EPSLON*PI4)))
  DO 5 J=1,3
    RMK(J)=RM(J)
5
C
* LOOP OVER THE SOURCE SEGMENTS.
C
  DO 1000 NS=1,NWSEG
    RADSK=K*RAD(NS)
    RADSKS=RADSK*RADSK
    DELRH=15.*RADSK
C
* OBTAIN K* THE COORDINATES OF THE SOURCE SEGMENT CENTROID
C
    NSF=NSEGC(1,NS)
    NST=NSEGC(2,NS)
    DO 2 J=1,3
      RSK(J)=K*WSEGH(J,NS)
      SH(J)=K*(WNODE(J,NST)-WNODE(J,NSF))
2
    CONTINUE
    DEL=SQRT(SH(1)*SH(1)+SH(2)*SH(2)+SH(3)*SH(3))
    DELS=DEL*DEL
    DELH=.5*DEL
    DO 4 J=1,3
      SH(J)=SH(J)/DEL
4
    CONTINUE
C
* LOOP OVER THE MATCH SEGMENTS.
C
* COMPUTE QUANTITIES FOR CALCULATION OF THE VECTOR AND SCALAR POTENTIALS
C
    CALL POTWIR(SSPOT,VAP,VAM,IPAT)
    IV=0
C
* LOOP OVER NODES ATTACHED TO THE SOURCE SEGMENT.
C
    DO 400 JNS=1,2
      NODES=NSEGC(JNS,NS)
      MULTS=MULTW(NODES)
      IF(MULTS.GT.0)THEN
C

```

```

* COMPUTE COLUMN INDEX FOR SOURCE SEGMENT.
C
      INDC=WIRSUM(NODES)
C
* DETERMINE WHETHER SEGMENT NS IS THE LOWEST SEGMENT ATTACHED TO NODE
* NODES. LOOP OVER THE NUMBER OF SEGMENTS ATTACHED TO NS AT NODE NODES.
C
      CALL NODMUL(INSEG,MXWMLT,NWNOD,NODES,MULTS,NS,JS1,JS2,
>      LOWS)
      IF(LOWS.AND.NODES.EQ.NWJUN(MNJUN)) THEN
        JS1=1
        JS2=1
        LOWS=.FALSE.
      ENDIF
      DO 300 J=JS1,JS2
        ICOL=INDC+J
        IV=IV+1
        IF(NODES.EQ.NSEGC(2,NS))THEN
          PLUS=.TRUE.
        ELSE
          PLUS=.FALSE.
        ENDIF
        IF(LOWS)THEN
C
* OBTAIN THE ATTACHED SEGMENT.
C
          NSEGAS=INSEG(J+1,NODES)
C
          * SGN=+1 IF SEGMENT NS & ATTACHED SEGMENT RUN IN THE SAME DIRECTION,
          * SGN=-1 IF SEGMENT NS & ATTACHED SEGMENT RUN IN THE OPPOSITE DIRECTION
C
          IF(NSEGC(2,NS).EQ.NSEGC(1,NSEGAS).OR.
>          NSEGC(1,NS).EQ.NSEGC(2,NSEGAS))THEN
            SGN=1.
          ELSE
            SGN=-1.
          ENDIF
          ELSE
            SGN=1.
          ENDIF
C
C
* COMPUTE APPROPRIATE VECTOR AND SCALAR POTENTIALS.
C
      IF(PLUS)THEN

```

```

C
C FOR PLUS SOURCE SEGMENT SET SIGN= (-1) TO VECTOR AND SCALAR POTENTIALS
C
  IF(NODES.EQ.NWJUN(MNJUN)) SGN=-1.
    DO 30 JJ=1,3
      VA(JJ)=VAP(JJ)*SGN
30    CONTINUE
      ELSE
C
C FOR MINUS SOURCE SEGMENT SET SIGN= (+1) TO VECTOR POTENTIAL AND
C (-1) TO SCALAR POTENTIALS
C
  IF(NODES.EQ.NWJUN(MNJUN)) SGN=1.
    DO 40 JJ=1,3
      VA(JJ)=VAM(JJ)*SGN
40    CONTINUE
      ENDIF
C
      CAXJ=VA(1)*CJ(ICOL)/K*CMPLX(C.,OMEGA)
      CAYJ=VA(2)*CJ(ICOL)/K*CMPLX(O.,OMEGA)
      CAZJ=VA(3)*CJ(ICOL)/K*CMPLX(O.,OMEGA)
      ETH(ITHETA,IPHI)=ETH(ITHETA,IPHI)
1      -CAXJ*COSD(THETA)*COSD(PHI)
2      -CAYJ*COSD(THETA)*SIND(PHI)
3      +CAZJ*SIND(THETA)
      EPH(ITHETA,IPHI)=EPH(ITHETA,IPHI)
1      +CAXJ*SIND(PHI)
2      -CAYJ*COSD(PHI)
300    CONTINUE
      ENDIF
400    CONTINUE
1000 CONTINUE
      WRITE(10,1100) ITHETA, THETA, IPHI, PHI, ETH(ITHETA, IPHI), EPH(ITHETA,
1100 1IPHI)
1100 FORMAT(I5,F8.2,I5,F8.2,4(1PE10.2))
C 1100 FORMAT(I5,F10.3,I5,F10.3,4(1PE15.5))
      RETURN
      END

```

A.48 CHARGE

```

C=====
SUBROUTINE CHARGE(NJCT,CI,DATNOD,NCONN,NBOUND,NNODES,NEDGES,

```

```

$ NFACES,NUNKNT,NWNOD,NWSEG,NUNKNB,WNODE,WIRSUM,NSEGC,MULTW,
$ MNJUN,MNJFACE,ANG,NJFACE,MIFACE,NWJUN,NBJUN,INDSUM,
$ IEDGF,MULT1,INSEG,MXWMLT)

```

```

C-----
C
C THIS SUBROUTINE COMPUTES THE CHARGE DISTRIBUTION ON THE
C BODY.THE CHARGE DENSITY IS COMPUTED AT THE CENTROID OF
C EACH TRIANGLE.
C
      IMPLICIT COMPLEX (C)
      COMPLEX CI(NUNKNT)
      REAL ANG(MNJUN,MNJFACE),VSJUN(3)
      INTEGER NJFACE(MNJUN,MNJFACE),MIFACE(MNJUN),NBUN(MNJUN),
&          NWJUN(MNJUN),INSEG(MXWMLT+1,NWNOD)
      DIMENSION DATNOD(3,NNODES)
      INTEGER NCONN(3,NEDGES),NBOUND(3,NFACES),INDSUM(NEDGES),
$WIRSUM(NWNOD),NSEGC(2,NWSEG),MULTW(NWNOD)
      REAL WNODE(3,NWNOD)
C
C COMPUTE CHARGE DENSITY ON THE BODIES
C
      CALL QBODY(NJCT,CI,DATNOD,NCONN,NBOUND,NNODES,NEDGES,
$NFACES,NUNKNT,NUNKNB,INDSUM,
$ WIRSUM,MNJUN,MNJFACE,NWNOD,ANG,NJFACE,MIFACE,NWJUN,NBJUN,
$IEDGF,MULT1)
C
C COMPUTE CHARGE DENSITY ON THE WIRES
C
      IF(NJCT.GE.0)THEN
          CALL QWIRE(CI,NWNOD,NWSEG,NUNKNB,NUNKNT,WNODE,WIRSUM,NSEGC,
$ MULTW,MNJUN,NWJUN,INSEG,MXWMLT)
      ENDIF
C
      RETURN
      END

```

A.49 QBODY

```

C=====
      SUBROUTINE QBODY(NJCT,CI,DATNOD,NCONN,NBOUND,NNODES,NEDGES,
$NFACES,NUNKNT,NUNKNB,INDSUM,
$ WIRSUM,MNJUN,MNJFACE,NWNOD,ANG,NJFACE,MIFACE,NWJUN,NBJUN,
$IEDGF,MULT1)

```

```

C-----
C
C THIS SUBROUTINE COMPUTES THE CHARGE DISTRIBUTION ON THE
C BODY. THE CHARGE DENSITY IS COMPUTED AT THE CENTROID OF
C EACH TRIANGLE.
C
      IMPLICIT COMPLEX (C)
      COMPLEX CI(NUNKNT),CS(3)
C
      REAL ANG(MNJUN,MNJFACE),VSJUN(3)
      INTEGER NJFACE(MNJUN,MNJFACE),MIFACE(MNJUN),NBJUN(MNJUN),
&          NWJUN(MNJUN),WIRSUM(NWNOD)
C
      DIMENSION DATNOD(3,NNODES),DN(3,3),IS(3),NS(3),DLM(3),SM(3)
      INTEGER NCONN(3,NEDGES),NBOUND(3,NFACES),INDSUM(NEDGES)
      REAL LAMBDA,K,MU,IMP
      COMMON/MEDIUM/DEG2RAD,EPSLON,MU,IMP,SL,PI
      COMMON/WAVE/OMEGA,LAMBDA,K
      SIZE(X,Y,Z)=SQRT(X*X+Y*Y+Z*Z)
      CONST1=CMPLX(0.0,1.0/OMEGA)
      CHARGE=CMPLX(0.0,0.0)
      WRITE(11,101)
101  FORMAT(X,/25X,'SURFACE CHARGE',/)
      WRITE(11,102)
102  FORMAT(1X,'FACE NUMBER',10X,'CHARGE DENSITY (COULOMBS/SQ.METER)')
      WRITE(11,103)
103  FORMAT(/20X,'REAL',11X,'IMAGINARY',8X,'MAGNITUDE',10X,'PHASE')
      DO 999 IFS=1,NFACES
C
C OBTAIN THE EDGES OF THE TRIANGLE.
C
      CALL FACEDG(NFACES,NBOUND,IFS,IS)
C
C OBTAIN THE VERTICES CONNECTED TO THESE EDGES.
C
      CALL FACVTX(NCONN,NEDGES,IS,NS)
C
C COMPUTE THE COORDINATES OF EACH VERTEX.
C
      CALL VTXCRD(DATNOD,NNODES,NS,DN)
C
      IF(NJCT.EQ.1)THEN
C
C TO FIND PARAMETERS ASSOCIATE WITH THIS JUNCTION
C

```

```

      CALL JUNPAR(JSF, DN, NS, IFS, ANGS, VSJUN, JSCOL,
&      WIRSUM, MNJUN, MNJFACE, NWNOD, ANG, NJFACE, MIFACE, NWJUN, NBJUN)
C
C ANGM IS THE ANGULAR DISTRIBUTION COEFFICIENCY OF MATCH TRIANGLE
C
      ELSE
      JSF=0
      ENDIF
C
C CALCULATE THE AREA OF THE TRIANGLE.
C
      AR1=(DN(2,2)-DN(1,2))*(DN(3,3)-DN(1,3))-(DN(3,2)-DN(1,2))*
& (DN(2,3)-DN(1,3))
      AR2=(DN(2,3)-DN(1,3))*(DN(3,1)-DN(1,1))-(DN(3,3)-DN(1,3))*
& (DN(2,1)-DN(1,1))
      AR3=(DN(2,1)-DN(1,1))*(DN(3,2)-DN(1,2))-(DN(3,1)-DN(1,1))*
& (DN(2,2)-DN(1,2))
      AREA=SQRT(AR1**2+AR2**2+AR3**2)/2.0
C
C CALCULATE THE LENGTHS OF EACH SIDE.
C
      DO 8 I=1,3
      IP1=MOD(I,3)+1
      IM1=MOD(I+1,3)+1
      DO 7 J=1,3
      DLM(J)=DN(IM1,J)-DN(IP1,J)
      SM(I)=SIZE(DLM(1),DLM(2),DLM(3))
      C
C COMPUTE THE CHARGE DENSITY ON THE TRIANGLE.
C
      CSUM=CMPLX(0.0,0.0)
      DO 460 ISL=1,3
      ISS=IS(ISL)
      MULTS=NCONN(3,ISS)
      IF(MULTS.NE.0)THEN
      IF(ISL.EQ.1)THEN
      FLAG=SM(1)
      IF(NCONN(1,IS(1)).EQ.NS(3))FLAG=-FLAG
      ELSEIF(ISL.EQ.2)THEN
      FLAG=SM(2)
      IF(NCONN(1,IS(2)).EQ.NS(1))FLAG=-FLAG
      ELSE
      FLAG=SM(3)
      IF(NCONN(1,IS(3)).EQ.NS(2))FLAG=-FLAG
      ENDIF

```



```

C FOR AN EDGE WITH MULTIPLICITY MULT, THE LOWEST NUMBERED FACE
C CONTRIBUTES TO MULT BASIS FUNCTIONS ASSOCIATED WITH THAT EDGE,
C WHILE EACH REMAINING FACE CONTRIBUTES TO ONE BASIS FUNCTION ASSOCIATED
C WITH THAT EDGE.

```

```

          CALL EDGMUL(IEDGF,MULT1,NEDGES,MULTS,ISS,IFS,JS1,JS2)
          INDS=INDSUM(ISS)
          DO 450 JS=JS1,JS2
            IV=IV+1
            ICOL=INDS+JS
450         CSUM=CSUM+FLAG*CONST1*CI(ICOL)
            ENDIF
460         CONTINUE
            IF(JSF.NE.0)THEN
              CSUM=CSUM+CONST1*CI(JSCOL)*ANGS
            ENDIF
          CHDEN=CSUM/CMPLX(AREA,0.0)
          RA1=REAL(CHDEN)
          RA2=AIMAG(CHDEN)
          RA3=CABS(CHDEN)
          EPS=1.E-7
          IF(RA2.EQ.0.)THEN
            RA4=0.
          ELSEIF(ABS(RA1/RA2).LT.EPS)THEN
            RA4=90.
          ELSE
            RA4=ATAN2(RA2,RA1)/DEG2RAD
          ENDIF
          WRITE(11,501) IFS,RA1,RA2,RA3,RA4
501        FORMAT(2X,I4,5X,1E,2X,1E,2X,1E,2X,1E)
          CHARGE=CHARGE+CSUM
999        CONTINUE
          WRITE(11,502) CHARGE
502        FORMAT(/10X,'TOTAL CHARGE ON THE BODY= (' ,2E,1X,') COULOMBS')
          RETURN
          END

```

A.50 QWIRE

```

C
C=====
          SUBROUTINE QWIRE(CI,NWNOD,NWSEG,NUNKNB,NUNKNT,WNODE,WIRSUM,NSEGC,
          $MULTW,MNJUN,NWJUN,INSEG,MXWMLT)
C-----

```

```

C THIS SUBROUTINE COMPUTES THE CHARGE DISTRIBUTION ON THE
C WIRE. THE CHARGE DENSITY IS COMPUTED AT THE CENTROID OF
C EACH SEGMENT.
C
      IMPLICIT COMPLEX (C)
C
      INTEGER WIRSUM(NWNOD),NSEGC(2,NWSEG),MULTW(NWNOD),NWJUN(MNJUN)
&,INSEG(MXWMLT+1,NWNOD)
      REAL WNODE(3,NWNOD),SH(3)
      COMPLEX CI(NUNKNT)
      REAL LAMBDA,K,MU,IMP
      LOGICAL LOWS,LOWM,PLUS
      COMMON/MEDIUM/DEG2RAD,EPSLON,MU,IMP,SL,PI
      COMMON/WAVE/OMEGA,LAMBDA,K
      SIZE(X,Y,Z)=SQRT(X*X+Y*Y+Z*Z)
      CONST1=CMPLX(0.0,1.0/OMEGA)
      CHARGE=CMPLX(0.0,0.0)
      WRITE(11,101)
101  FORMAT(X,/25X,'CHARGE DENSITY ON WIRE',/)
      WRITE(11,102)
102  FORMAT(1X,'SEGMENT NUMBER',10X,'CHARGE DENSITY
& (COULOMBS/SQ.METER)')
      WRITE(11,103)
103  FORMAT(/20X,'REAL',11X,'IMAGINARY',9X,'MAGNITUDE',10X,'PHASE')
C
* LOOP OVER THE SOURCE SEGMENTS.
C
      DO 1000 NS=1,NWSEG
      CSUM=CMPLX(0.0,0.0)
          NSF=NSEGC(1,NS)
          NST=NSEGC(2,NS)
          DO 2 J=1,3
              SH(J)=(WNODE(J,NST)-WNODE(J,NSF))
          2  CONTINUE
          DEL=SQRT(SH(1)*SH(1)+SH(2)*SH(2)+SH(3)*SH(3))
C
* LOOP OVER NODES ATTACHED TO THE SEGMENT.
C
          DO 400 JNS=1,2
              NODES=NSEGC(JNS,NS)
              MULTS=MULTW(NODES)
              IF(MULTS.GT.0)THEN
C
* COMPUTE COLUMN INDEX FOR SOURCE SEGMENT.
C

```

```

      INDC=WIRSUM(NODES)
C
* DETERMINE WHETHER SEGMENT NS IS THE LOWEST SEGMENT ATTACHED TO NODE
* NODES. LOOP OVER THE NUMBER OF SEGMENTS ATTACHED TO NS AT NODE NODES.
C
      CALL NODMUL(INSEG,MXWMLT,NWNOD,NODES,MULTS,NS,JS1,JS2,
>         LOWS)
      IF(LOWS.AND.NODES.EQ.NWJUN(MNJUN)) THEN
        JS1=1
        JS2=1
        LOWS=.FALSE.
      ENDIF
      DO 300 J=JS1,JS2
        ICOL=INDC+J
        IV=IV+1
        IF(NODES.EQ.NSEGC(2,NS))THEN
          PLUS=.TRUE.
        ELSE
          PLUS=.FALSE.
        ENDIF
        IF(LOWS)THEN
C
* OBTAIN THE ATTACHED SEGMENT.
C
          NSEGAS=INSEG(J+1,NODES)
C
* SGN=+1 IF SEGMENT NS & ATTACHED SEGMENT RUN IN THE SAME DIRECTION,
* SGN=-1 IF SEGMENT NS & ATTACHED SEGMENT RUN IN THE OPPOSITE DIRECTION
C
          IF(NSEGC(2,NS).EQ.NSEGC(1,NSEGAS).OR.
>         NSEGC(1,NS).EQ.NSEGC(2,NSEGAS))THEN
            SGN=1.
          ELSE
            SGN=-1.
          ENDIF
          ELSE
            SGN=1.
          ENDIF
C
C
* COMPUTE APPROPRIATE VECTOR AND SCALAR POTENTIALS.
C
      IF(PLUS)THEN
C
C FOR PLUS SOURCE SEGMENT SET SIGN= (-1) TO VECTOR AND SCALAR POTENTIALS

```

```

C
      IF(NODES.EQ.NWJUN(MNJUN)) SGN=-1.
          ELSE
C
C FOR MINUS SOURCE SEGMENT SET SIGN= (+1) TO VECTOR POTENTIAL AND
C (-1) TO SCALAR POTENTIALS
C
      IF(NODES.EQ.NWJUN(MNJUN)) SGN=1.
          ENDIF
          IF(JNS.EQ.2)SGN=-SGN
          CSUM=CSUM-CONST1*CI(ICOL)*SGN
300      CONTINUE
          ENDIF
400      CONTINUE
      CHDEN=CSUM/DEL
      RA1=REAL(CHDEN)
      RA2=AIMAG(CHDEN)
      RA3=CABS(CHDEN)
          EPS=1.E-7
          IF(RA2.EQ.0.)THEN
              RA4=0.
          ELSEIF(ABS(RA1/RA2).LT.EPS)THEN
              RA4=90.
          ELSE
              RA4=ATAN2(RA2,RA1)/DEG2RAD
          ENDIF
      WRITE(11,501) NS,RA1,RA2,RA3,RA4
501  FORMAT(2X,I4,5X,1E,2X,1E,2X,1E,2X,1E)
      CHARGE=CHARGE+CSUM
1000  CONTINUE
      WRITE(11,502) CHARGE
502  FORMAT(/10X,'TOTAL CHARGE ON THE WIRE= (',2E,1X,') COULOMBS')
      RETURN
      END

```

A.51 CPATT

```

C-----
      COMPLEX FUNCTION CPATT(SP)
C-----
* INPUT:
* RMK(J),J=1,2,3=K* THE X,Y,Z COMPONENTS OF THE MATCH POINT.
* SH(J) J=1,2,3 = THE X,Y,Z COMPONENTS OF THE UNIT VECTOR POINTING IN

```

```

* THE SAME DIRECTION AS THE SOURCE SEGMENT.
* RSK(J)= K* THE X,Y,Z COORDINATES OF THE THE SOURCE SEGMENT CENTROID
* RADSK=K * THE SOURCE SEGMENT RADIUS.
* RADSKS=RADSK*RADSK
* SP IS K* THE DISTANCE ALONG THE SOURCE SEGMENT THAT RPRIME IS FROM
* THE SOURCE SEGMENT CENTROID. A POSITIVE DISTANCE IS TOWARDS THE
* ENDPOINT OF THE SOURCE SEGMENT, A NEGATIVE DISTANCE IS TOWARDS THE
* INITIAL POINT OF THE SOURCE SEGMENT.
* OUTPUT:
* CKMN=THE REDUCED KERNEL EVALUATED AT R=THE MATCH POINT AND
* RPRIME(J)=RSK(J)-SP*SH(J).

```

```

C-----
  DIMENSION RMK(3),SH(3),RSK(3),D(3)
  COMMON/WKERNL/RSK,SH,RMK
C
  DO 5 J=1,3
    D(J)=RSK(J)-SP*SH(J)
5  CONTINUE
  R=D(1)*RMK(1)+D(2)*RMK(2)+D(3)*RMK(3)
  CPATT=CEXP(CMPLX(0.,R))
  RETURN
  END

```

A.52 SGQADS

```

C=====
  SUBROUTINE SGQADS(FCT,XL,XU,IQUAD,ANS)
C-----
C GAUSSIAN QUADRATURE INTEGRAL OVER WIRE SEGMENT FOR SELF TERM
* INPUTS:
* FCT IS A COMPLEX FUNCTION.
* IQUAD IS THE NUMBER OF INTEGRATION POINTS( IQUAD MUST BE 4,8 OR 16)
* XL IS THE LOWER INTEGRATION LIMIT
* XU IS THE UPPER INTEGRATION LIMIT
* OUTPUT:
* ANS IS THE INTEGRAL OF FCT(X) FROM X=XL TO XU.
C-----
  COMPLEX FCT,ANS,FP,FM,CSUM
  REAL AC(2),AD(2),AE(4),AF(4),AG(8),AH(8)
  EXTERNAL FCT
  DATA AC(1),AC(2),AD(1),AD(2) /
  > .339981043584856,.861136311594053,
  > .652145154862546,.347854845137454/

```

```

DATA AE(1),AE(2),AE(3),AE(4),AF(1),AF(2),AF(3),AF(4)/
> .183434642495650, .525532409916329,
> .796666477413627, .960289856497536,
> .362683783378362, .313706645877887,
> .222381034453374, .101228536290376/
DATA AG/.095012509837637, .281603550779258,
> .4508016777657227, .617876244402643, .755404408355003,
> .865631202387831, .944575023073232, .989400934991649/
DATA AH/.189450610455068, .182603415044923,
> .169156519395002, .149595988816576, .124628971255533,
> .095158511682492, .062253523938647, .027152459411754/

```

```

C
  CSUM=(0.,0.)
  DEL=.5*(XU-XL)
  XC=.5*(XL+XU)
  IF(IQUAD.EQ.4)THEN
C
  DO 20 J=1,2
    C=AC(J)*DEL
    FP=FCT(XC+C)
    FM=FCT(XC-C)
    CSUM=CSUM+AD(J)*(FP+FM)
20  CONTINUE
  ELSEIF(IQUAD.EQ.8)THEN
C
  DO 200 J=1,4
    C=AE(J)*DEL
    FP=FCT(XC+C)
    FM=FCT(XC-C)
    CSUM=CSUM+AF(J)*(FP+FM)
200 CONTINUE
  ELSEIF(IQUAD.EQ.16)THEN
    DO 300 J=1,4
      C=AE(J)*DEL
      FP=FCT(XC+C)
      FM=FCT(XC-C)
      CSUM=CSUM+AF(J)*(FP+FM)
300 CONTINUE
  ELSE
    WRITE(4,10)IQUAD
10  FORMAT(1X,'WARNING IN QG IQUAD OUT OF RANGE,IQUAD=',I5)
    STOP
  ENDIF
  ANS=CSUM*DEL

```

RETURN
END

A.53 SEGQAD

```
C=====
      SUBROUTINE SEGQAD(FCT,DEL,IQUAD,ANS)
C-----
C GAUSSIAN QUADRATURE INTEGRAL OVER WIRE SEGMENT
* INPUTS:
* FCT IS A COMPLEX FUNCTION.
* DEL IS AN INTEGRATION LIMIT.
* IQUAD IS THE NUMBER OF INTEGRATION POINTS(FOR NOW IQUAD MUST BE 4)
* OUTPUT:
* ANS(J)=1./DEL * THE INTEGRAL FROM SP=-DEL/2 TO DEL/2
*       FCT(SP)*VEC(J) D(SP)
* WHERE VEC(1)=1,VEC(2)=SP,VEC(3)=DEL-SP
C-----
      COMPLEX FCT,ANS(3),FP,FM
      REAL AC(2),AD(2),AE(4),AF(4)
      EXTERNAL FCT
      DATA AC(1),AC(2),AD(1),AD(2) /
>          .339981043584856,.861136311594053,
>          .652145154862546,.347854845137454/
      DATA AE(1),AE(2),AE(3),AE(4),AF(1),AF(2),AF(3),AF(4)/
>          .183434642495650,.525532409916329,
>          .796666477413627,.960289856497536,
>          .362683783378362,.313706645877887,
>          .222381034453374,.101228536290376/
C
      DEL2=DEL/2.
      DO 5 J=1,3
        ANS(J)=(0.,0.)
5      CONTINUE
      IF(IQUAD.EQ.4)THEN
        DO 40 J=1,2
          C=AC(J)*DEL2
          FP=FCT(C)
          FM=FCT(-C)
          ANS(1)=ANS(1)+AD(J)*(FP+FM)
          ANS(2)=ANS(2)+AD(J)*(FP*C-FM*C)
40       CONTINUE
          ANS(1)=DEL2*ANS(1)
```

```

        ANS(2)=.5*(ANS(2)+ANS(1))
        ANS(3)=ANS(1)-ANS(2)
    ELSEIF(IQUAD.EQ.8)THEN
        DO 400 J=1,4
            C=AE(J)*DEL2
            FP=FCT(C)
            FM=FCT(-C)
            ANS(1)=ANS(1)+AF(J)*(FP+FM)
            ANS(2)=ANS(2)+AF(J)*(FP-C-FM*C)
400     CONTINUE
        ANS(1)=DEL2*ANS(1)
        ANS(2)=.5*(ANS(2)+ANS(1))
        ANS(3)=ANS(1)-ANS(2)
    ELSE
        WRITE(4,10)IQUAD
10     FORMAT(1X,'WARNING IN QUAD IQUAD OUT OF RANGE,IQUAD=',I5)
        STOP
    ENDIF
    RETURN
    END

```

A.54 WCUMUL

```

-----
      SUBROUTINE WCUMUL(NWNOD,NUNKNB,MULTW,WIRSUM)
-----
C CUMULATE MULTIPLICITY OF WIRE NODE
* INPUT:
* NWNOD=THE NUMBER OF WIRE UNKNOWNNS.
* NUNKNB=THE NUMBER OF BODY UNKNOWNNS.
* MULTW(N)=THE MULTIPLICITY OF THE NTH WIRE NODE.
* OUTPUT:
* WIRSUM(I)=NUNKNB+THE SUM OF WIRE NODE MULTIPLICITIES UP TO
* (BUT NOT INCLUDING)THE NODE I.
-----
      INTEGER MULTW(NWNOD),WIRSUM(NWNOD)
      WIRSUM(1)=0
      DO 10 I=2,NWNOD
          WIRSUM(I)=MULTW(I-1)+WIRSUM(I-1)
10     CONTINUE
      DO 20 I=1,NWNOD
          WIRSUM(I)=WIRSUM(I)+NUNKNB
20     CONTINUE

```



```
RETURN
END
```

A.55 NODMUL

```
C=====
      SUBROUTINE NODMUL(INSEG,MXWMLT,NWNOD,NODE,MULTN,NSEG,JN1,JN2,LOW)
C-----
C MAPPING FROM NODE TO MULTIPLICITY
* INPUT:
* INSEG(M,N) M=1,...,MULTW(N)+1 CONTAIN THE WIRE SEGMENTS ATTACHED
* TO THE NTH NODE.
* MXWMLT=THE MAXIMUM MULTIPLICITY OF ANY WIRE NODE.
* NWNOD THE NUMBER OF WIRE NODES.
* NODE= THE WIRE NODE NUMBER.
* MULTN=THE MULTIPLICITY OF THE NODE 'NODE'
* NSEG= THE SEGMENT NUMBER
* OUTPUT: JN1 JN2
* IF NSEG IS THE LOWEST NUMBERED SEGMENT ATTACHED TO NODE 'NODE'
* THEN JN1=1 AND JN2=MULTN
* ELSE IF NSEG IS THE ITH SEGMENT(>FIRST) ATTACHED TO NODE 'NODE'
* THEN JN1=JN2=I-1
* ENDIF
C-----
      INTEGER INSEG(MXWMLT+1,NWNOD)
      LOGICAL LOW
C
      IF(INSEG(1,NODE).EQ.NSEG)THEN
        LOW=.TRUE.
        JN1=1
        JN2=MULTN
      ELSE
        LOW=.FALSE.
        M1=MULTN+1
        DO 10 I=2,M1
          IF(NSEG.EQ.INSEG(I,NODE))THEN
            I1=I-1
            JN1=I1
            JN2=I1
            GO TO 11
          ENDIF
        CONTINUE
      10 CONTINUE
      11 CONTINUE
```

```
ENDIF
RETURN
END
```

A.56 WIROUT

```
C=====
      SUBROUTINE WIROUT(CV,NUNKNT,NUNKNB,MULTW,NWNOD)
C-----
C THIS SUBROUTINE PRINTS OUTPUT DATA ASSOCIATED WITH WIRES
C
      COMPLEX CV(NUNKNT)
      INTEGER MULTW(NWNOD)
      COMMON/F/FREQ
      DEG2RAD=3.14159265358979/180.
C
* WRITE THE CURRENT DENSITY TABLE.
C
      WRITE(9,22)
22      FORMAT(//28X,'SURFACE CURRENTS'/)
      WRITE(9,23)
23      FORMAT(1X,'EDGE NUMBER ',13X,'CURRENT DENSITY (AMPS/METER)')
      WRITE(9,24)
24      FORMAT
>      (14X,'REAL',9X,'IMAGINARY',7X,'MAGNITUDE',7X,'PHASE(DEG)')
      CO=(0.,0.)
      K1=0+NUNKNB
      DO 50 I50=1,NWNOD
      IF(MULTW(I50).EQ.0)THEN
      WRITE(9,101)I50,CO,0.
      WRITE(4,101)I50,CO,0.
      A=0.
      ELSE
      DO 35 I35=1,MULTW(I50)
      K1=K1+1
      RA1=REAL(CV(K1))
      RA2=AIMAG(CV(K1))
      RA3=CABS(CV(K1))
      IF(ABS(RA1).LT.1.E-10)THEN
      RA4=90.
      ELSE
      RA4=ATAN2(RA2,RA1)/DEG2RAD
      ENDIF
```

```

        WRITE(9,101) I50,RA1,RA2,RA3,RA4
        WRITE(4,101) I50,RA1,RA2,RA3,RA4
35      CONTINUE
        ENDIF
50      CONTINUE
101     FORMAT(2X,I4,2X,3(2X,E12.5,2X),F12.3)
C
C PRINT OUT CURRENT ON THE JUNCTION NODE
C IT IS INPUT CONDUCTANCE IF SET RIGHT HAND SIDE CV(NUNKNT)=1
C
C      WRITE(11,*)FQ,CV(NUNKNT)
C      WRITE(11,*)FQ,CV(NUNKNB+1)
C102    FORMAT(X,F6.4,',',E15.5,',',E15.5,',')
        RETURN
        END

```

A.57 FNDJUN

```

C=====
      SUBROUTINE FNDJUN(WNODE,NWNOD,DATNOD,NNODES,MNJUN,NWJUN,NBJUN)
C-----
C PURPOSE:
C FOR BODY AND WIRE, FIND OUT WHICH NODE IS JUNCTION
C
C INPUT:
C MNJUN: NUMBER OF JUNCTION
C OUTPUT:
C NWJUN(I): WIRE NODE NUMBER OF THE ITH JUNCTION I=1,MNJUN
C NBJUN(I): BODY NODE NUMBER OF THE ITH JUNCTION I=1,MNJUN
C-----
      REAL WNODE(3,NWNOD),DATNOD(3,NNODES)
      INTEGER NWJUN(MNJUN),NBJUN(MNJUN)
      NJ=0
C
C LOOP OVER EACH WIRE NODE AND BODY NODE
C
      EPS=1.E-4
      DO 10 NW=1,NWNOD
      DO 10 N=1,NNODES
C
C IF WIRE NODE AND BODY NODE AT THE SAME POINT IT IS A JUNCTION POINT
C
      E1=ABS(WNODE(1,NW)-DATNOD(1,N))

```

```

E2=ABS(WNODE(2,NW)-DATNOD(2,N))
E3=ABS(WNODE(3,NW)-DATNOD(3,N))
IF(E1.GT.EPS.OR.E2.GT.EPS.OR.E3.GT.EPS) GO TO 10
NJ=NJ+1
NWJUN(NJ)=NW
NBUN(NJ)=N
10 CONTINUE
WRITE(3,*)' NUMBER OF JUNCTION = ',NJ
DO 20 NN=1,NJ
20 WRITE(3,*)' ON WIRE NODE = ',NWJUN(NN)

RETURN
END

```

A.58 JUNFAC

```

C=====
SUBROUTINE JUNFAC(NBJUN,MNJUN,NBOUND,NFACES,NCONN,NEDGES,
& NJFACE,MNJFACE,MIFACE)
C-----
C PURPOSE:
C FOR EACH JUNCTION, FIND OUT THE PATCHES ATTACH TO?
C
C INPUT:
C NBJUN(I):NODE NUMBER OF THE ITH JUNCTION POINT. I=1,MNJUN
C MNJFACE:MAXIMUN NUMBER OF FACE ATTACHING TO JUNCTION POINT
C OUTPUT:
C NJFACE(I,J):FACE NUMBER OF THE JTH FACE ATTACHING TO THE ITH JUNCTION
C POINT. I=1,MNJUN, J=1,MIFACE(I)
C MIFACE(I):MAXIMUN NUMBER OF FACE ATTACHING TO THE ITH JUNCTION POINT.
C-----
INTEGER NBJUN(MNJUN),NCONN(3,NEDGES),NBOUND(3,NFACES)
& ,NJFACE(MNJUN,MNJFACE),MIFACE(MNJUN),IE(3),NV(3)
C
C LOOP OVER EACH JUNCTION POINT
C
EPS=1.E-3
DO 10 NJ=1,MNJUN
I=0
C
C CHECK IF THIS FACE ATTACHING TO JUNCTION POINT
C
DO 20 N=1,NFACES

```

```

      CALL FACEDG(NFACES,NBOUND,N,IE)
      CALL FACVTX(NCONN,NEDGES,IE,NV)
      E1=ABS(NV(1)-NBUN(NJ))
      E2=ABS(NV(2)-NBUN(NJ))
      E3=ABS(NV(3)-NBUN(NJ))
      IF(E1.GT.EPS.AND.E2.GT.EPS.AND.E3.GT.EPS) GO TO 20
      I=I+1
      NJFACE(NJ,I)=N
20    CONTINUE
      MIFACE(NJ)=I
10    CONTINUE
      RETURN
      END

```

A.59 JANGLE

```

C=====
      SUBROUTINE JANGLE(DATNOD,NNODES,NBUN,MNUN,NBOUND,NFACES,NCONN,
&      NEDGES,NJFACE,MNJFACE,MIFACE,ANG)
C-----
C PURPOSE: FOR EACH JUNCTION, COMPUTE VERTEX ANGLE OF EACH ATTACHED
C TRIANGLES.
C OUTPUT:
C ANG(I,J): ANGLE FACTOR OF THE JTH PATCH ATTACHING TO THE ITH JUNCTION
C POINT
C SANG(I): THE SUM OF THE VERTEX ANGLE OF THE ITH JUNCTION POINT
C-----
      REAL DATNOD(3,NNODES),SANG(40),ANG(MNUN,MNJFACE),DN(3,3)
&      ,VP(3),VM(3)
      INTEGER NBUN(MNUN),NCONN(3,NEDGES),NBOUND(3,NFACES)
&      ,NJFACE(MNUN,MNJFACE),MIFACE(MNUN),NV(3),IE(3)
C
      SIZE(A,B,C)=SQRT(A*A+B*B+C*C)
C
C LOOP OVER EACH JUNCTION POINT
C
      DO 10 I=1,MNUN
      SANG(I)=0.
C
C LOOP OVER EACH ATTACHED FACE
C
      DO 10 J=1,MIFACE(I)
      IJ=NJFACE(I,J)

```

```

      CALL FACEDG(NFACES,NBOUND,IJ,IE)
C
C FROM THESE EDGES, OBTAIN THE VERTICES OF THE TRIANGLE.
C
      CALL FACVTX(NCONN,NEDGES,IE,NV)
C
C CALCULATE THE COORDINATES OF EACH VERTEX.
C
      CALL VTXCRD(DATNOD,NNODES,NV,DN)
      DO 20 II=1,3
20    IF(NV(II).EQ.NBJUN(I)) GO TO 30
30    IP1=MOD(II,3)+1
      IM1=MOD(IP1,3)+1
      DO 40 JJ=1,3
      VP(JJ)=DN(IP1,JJ)-DN(II,JJ)
40    VM(JJ)=DN(IM1,JJ)-DN(II,JJ)
      EP=SIZE(VP(1),VP(2),VP(3))
      EM=SIZE(VM(1),VM(2),VM(3))
      DOT=VP(1)*VM(1)+VP(2)*VM(2)+VP(3)*VM(3)
C
C A DOT B = LAL LBL COS(ANG)
C
      ANG(I,J)=ACOS(DOT/EP/EM)
      SANG(I)=SANG(I)+ANG(I,J)
10    CONTINUE
      DO 100 I=1,MNJUN
      DO 100 J=1,MIFACE(I)
100   ANG(I,J)=ANG(I,J)/SANG(I)
      RETURN
      END

```

A.60 JUNPAR

```

C=====
      SUBROUTINE JUNPAR(JFLAG,DATNOD,NODE,IFACE,AANGLE,VETJUN,INDEX,
      & WIRSUM,MNJUN,MNJFACE,NWNOD,ANG,NJFACE,MIFACE,NWJUN,NBJUN)
C-----
C PURPOSE:
C FIND OUT GEOMETRY PARAMETERS ASSOCIATE WITH THIS JUNCTION PATCH
C
C INPUT:
C DATNOD(I,J):J=1,3 ARE THE X,Y,Z COORDINATES OF THE ITH VERTEX OF THIS
C PATCH

```

```

C NODE(I):I=1,3 NODE NUMBER OF THE ITH VERTEX OF THIS PATCH
C IFACE:FACE NUMBER OF THIS PATCH
C OUTPUT:
C ANGLE:VERTEX ANGLE FACTOR OF THIS PATCH ATTACHING TO THE JUNCTION
C POINT
C VETJUN(J):J=1,3 TESTING VECTOR RUNNING FROM THE CENTER TO THE
C JUNCTION VERTEX

```

```

-----
      REAL ANG(MNJUN,MNJFACE),DATNOD(3,3),DATNODC(3),VETJUN(3)
      INTEGER NJFACE(MNJUN,MNJFACE),MIFACE(MNJUN),NBJUN(MNJUN),
&         NWJUN(MNJUN),NODE(3),WIRSUM(NWNOD)
C
C FIND JUNCTION VERTEX AND CORRESPONDING JUNCTION NUMBER
C
      DO 10 INODE=1,3
      DO 10 JUNNO=1,MNJUN
      IF(NODE(INODE).EQ.NBJUN(JUNNO)) GO TO 20
10    CONTINUE
C
C SET JFLAG=0, THERE IS NOT ANY JUNCTION ON THIS PATCH
C
      JFLAG=0
      GO TO 100
C
C SET JFLAG= WHICH VERTEX OF THIS JUNCTION PATCH,
C IF THERE IS ANY JUNCTION ON THIS PATCH
C
20    JFLAG=INODE
C
C DETERMINE ROW(MATCHING) OR COLUMN(SOURCE) LOCATION IN THE MATRIX
C
      INDEX=WIRSUM(NWJUN(JUNNO))+1
C
C FIND THE VERTEX ANGLE OF THE JUNCTION PATCH
C
      DO 30 M=1,MIFACE(JUNNO)
      IF(NJFACE(JUNNO,M).EQ.IFACE) GO TO 40
30    CONTINUE
      WRITE(6,*)' ERROR : CAN NOT FIND JUNCTION FACE '
40    AANGLE=ANG(JUNNO,M)
C
C DATNODC(J),J=1,3 IS THE CENTER OF THE PATCH
C
      DO 50 JJ=1,3
C

```

```

C COMPUTE THE VECTOR OF TESTING PATH
C
      DATNODC(JJ)=(DATNOD(1,JJ)+DATNOD(2,JJ)+DATNOD(3,JJ))/3.
50   VETJUN(JJ)=DATNOD(INODE,JJ)-DATNODC(JJ)
100  RETURN
      END

```

A.61 FACPAR

```

C=====
      SUBROUTINE FACPAR(JSF,RK,RMK)
C-----
C THIS SUBROUTINE COMPUTES MOST PARAMETERS FOR COMPUTING VECTOR AND SCALAR
C POTENTIALS
C JSF = 1,2 OR 3 IS JUNCTION VERTEX OF THE TRIANGLE
C RK(I,J) IS THE VERTEX COORDINATE OF THE SOURCE TRIANGLE
C RMK(J) IS THE MATCHING POINT OR CENTER OF THE MATCHING TRIANGLE
C-----
      IMPLICIT COMPLEX (C)
      DIMENSION RMKO(3),RI(3,3),P(3,3),PH(3),RIL(3),URI(3,3),RR2(3)
      $      ,VH(3,3),RR(3),RRR(3)
      DIMENSION H(3),RK(3,3),RMK(3),UN(3),UL(3,3),UH(3,3),AL(3),DL(3,3)
      COMMON/CONJ/XSING2
      COMMON/MINMAX/RLMINK,RLMAXK
      COMMON/JUN/RI,P,PH,RR,RR2,RIL,URI,VH
      COMMON/PARA/DL,DET,H,AL,UL,UH,RMKO,UN,DPERPK
      SIZE(X,Y,Z)=SQRT(X*X+Y*Y+Z*Z)
C
C HAT N = [(VET R(2)-VET R(1)) X (VET R(3)-VET R(1))] /
C ABS[(VET R(2)-VET R(1)) X (VET R(3)- VET R(1))]
C
      PK=JSF
      DO 10 I=1,3
      IP1=MOD(I,3)+1
      IM1=MOD(I+1,3)+1
      DO 20 J=1,3
20   DL(I,J)=RK(IM1,J)-RK(IP1,J)
      AL(I)=SIZE(DL(I,1),DL(I,2),DL(I,3))
      DO 30 J=1,3
30   UL(I,J)=DL(I,J)/AL(I)
10   CONTINUE
C
      RLMINK=AMIN1(AL(1),AL(2),AL(3))

```



```

      RLMAXK=AMAX1(AL(1),AL(2),AL(3))
C
      DO 40 J=1,3
      JP1=MOD(J,3)+1
      JM1=MOD(J+1,3)+1
40    UN(J)=DL(3,JP1)*DL(1,JM1)-DL(3,JM1)*DL(1,JP1)
      DET=SIZE(UN(1),UN(2),UN(3))
      DO 50 J=1,3
      UN(J)=UN(J)/DET
50    H(J)=DET/AL(J)
C
C   VH(I,J)= THE VECTOR OF THE HEIGHT H(I)
C
      DO 70 I=1,3
      DO 70 J=1,3
      JP1=MOD(J,3)+1
      JM1=MOD(J+1,3)+1
      UH(I,J)=UL(I,JP1)*UN(JM1)-UL(I,JM1)*UN(JP1)
70    VH(I,J)=H(I)*UH(I,J)
      DPERPK=0.
      DO 80 J=1,3
80    DPERPK=DPERPK+UN(J)*(RMK(J)-RK(1,J))
      DO 90 J=1,3
90    RMKO(J)=RMK(J)-DPERPK*UN(J)
      DPERPK=ABS(DPERPK)
      DO 100 I=1,3
      DO 110 J=1,3
110   RI(I,J)=RMK(J)-RK(I,J)
      RIL(1)=SIZE(RI(I,1),RI(I,2),RI(I,3))
      RIH=0.
      DO 120 J=1,3
      URI(I,J)=RI(I,J)/RIL(I)
120   RIH=RIH+URI(I,J)*UH(I,J)
      RRR(I)=RIH
      RR(I)=RIL(I)*RIH
      RR2(I)=RR(I)*RR(I)
22    FORMAT(3X,3E15.5)
      DO 130 J=1,3
130   P(I,J)=RMKO(J)-RK(I,J)
      PDH=0.
      DO 140 J=1,3
140   PDH=PDH+P(I,J)*UH(I,J)
      PH(I)=PDH
100   CONTINUE
C

```

```

C IF RI IS NEAR PERPENDICULAR TO HI, SET XSING2=1.
C
  IF(JSF.NE.0)THEN
  IF(RRR(JSF).LT.1.E-6) THEN
  XSING2=1.
  ELSE
  XSING2=0.
  ENDIF
  ENDIF
  RETURN
  END

```

A.62 ADWMUL

```

C=====
  SUBROUTINE ADWMUL(NWNOD,WNODE,MULTW,NWUNKS)
C-----
C ADJUST MULTIPLICITY OF WIRE NODE
C INPUT:
C NWNOD=THE NUMBER OF WIRES NODES.
C NWUNKS=THE NUMBER OF WIRES UNKNOWN BEFORE CONSIDERING THE
C GROUND PLANE ATTACHMENTS.
C WNODE(I,N)=THE X,Y,Z COMPONENTS(I=1,2,3) OF THE NTH NODE N=1,NWNOD.
C MULTW(N)= THE MULTIPLICITY OF THE NTH NODE N=1,NWNOD(BEFORE ANY GROUND
C PLANE ATTACHMENTS ARE CONSIDERED).
C
C OUTPUT:
C FOR EACH NODE IN THAT IS CONNECTED TO A P.E.C. GROUND PLANE AND IS
C NOT CONNECTED TO A P.M.C. GROUND PLANE, ITS MULTIPLICITY(MULTW(IN))
C AND THE NUMBER OF WIRES UNKNOWN(NWUNKS) ARE INCRIMENTED BY 1.
C THE NODE CONNECTION LIST WITH MULTIPLICITIES IS OUTPUTTED
C AFTER ACCOUNTING FOR ALL GROUND PLANE ATTACHMENTS.
C-----
  DIMENSION WNODE(3,NWNOD),MULTW(NWNOD),IGNDP(3),DM(3)
  COMMON/GPLANE/NGNDP,IGNDP
  IF(NGNDP.GT.0)THEN
  GLIMIT=1.E-4
  DO 100 IN=1,NWNOD
  DO 2 I=1,3
2  DM(I)=ABS(WNODE(I,IN))
  IX=0
  IY=0
  IZ=0

```

```

IF(DM(1).LE.GLIMT)IX=IGNDP(1)
IF(IX.NE.1)THEN
  IF(DM(2).LE.GLIMT)IY=IGNDP(2)
  IF(IY.NE.1.AND.DM(3).LE.GLIMT)IZ=IGNDP(3)
ENDIF
IMAX=AMAXO(IX,IY,IZ)
IF(IMAX.NE.1)THEN
  IMIN=AMINO(IX,IY,IZ)
  NWUNKS=NWUNKS-IMIN
  MULTW(IN)=MULTW(IN)-IMIN
ENDIF
100 CONTINUE
ENDIF
RETURN
END

```

A.63 POTBOD

```

C=====
C      SUBROUTINE FOTBOD(JSF,RK,RMK,C,CA,CAJ,IPAT)
C-----
C COMPUTE POTENTIALS ASSOCIATED WITH BODY FACE
C   INPUT:
C   RK(J,I) I=1,2,3 DENOTES K*THE X,Y,Z COORDINATES OF THE JTH
C   VERTEX OF THE SOURCE TRIANGLE.(K=THE WAVE NUMBER)
C   RMK(I) I=1,2,3 DENOTE THE X,Y,Z COORDINATES OF THE MATCH POINT.
C   CONSTANTS PASSED THROUGH COMMON/GPLANE/
C   NGNDP=0,1,2,OR 3 THE NUMBER OF IMAGE PLANES.
C   IGNDP(I)=0 NO GROUND PLANES;
C             -1 A P.E.C. GROUND PLANE;
C             1 A P.M.C. GROUND PLANE.
C             I=1,2,3 DENOTES THE X=0,Y=0,ANDZ=0 GROUND PLANES.
C   OUTPUT:
C   AL(I),I=1,3 DENOTE K* THE LENGTHS OF THE SOURCE TRIANGLES SIDES
C   OPPOSITE THE 1 2 3 LOCAL VERTICIES RESPECTIVELY.
C   CI(I),I=1,3 IS
C   C=CI(1)+CI(2)+CI(3)
C   SCALAR POTENTIAL =-AL(I)*C/(2.*PI*OMEGA*EPSO*I) I=SQRT(-1.)
C   AL(I) IS THE LENGTH OF THE ITH SIDE OF THE SOURCE TRIANGLE *K
C   VECTOR POTENTIAL(I)=(MUO*AL(I)/(4*PI*K))*(CI(IP1)*DL(IM1,J)-
C   CI(IM1)*DL(IP1,J))
C-----
C      COMPLEX CI(3),C,CPP,CMM,CA(3,3),CAJ(3)

```

```

C
C VARIABLES ASSOCIATED WITH IMAGE PATCH
$ CII(3),CPPI,CMMI
C
  DIMENSION RK(3,3),RMK(3),IGNDP(3)
  COMMON/SPANE/NGNDP,IGNDF
  CALL FACPAR(JSF,RK,RMK)
  CALL PD3FAC(JSF,RK,RMK,C,CA,CAJ,IPAT)
  IF(NGNDP.GT.0)THEN
    DO 20 I=1,3
      IF(IGNDP(I).NE.0)CALL BIMAGE(JSF,RK,RMK,C,CA,CAJ,I,IPAT)
20    CONTINUE
      IF(NGNDP.GE.2)THEN
        IF(IGNDP(1).EQ.0)THEN
          CALL BIMAGE(JSF,RK,RMK,C,CA,CAJ,2,IPAT)
        ELSEIF(IGNDP(2).EQ.0)THEN
          CALL BIMAGE(JSF,RK,RMK,C,CA,CAJ,3,IPAT)
        ELSE
          CALL BIMAGE(JSF,RK,RMK,C,CA,CAJ,1,IPAT)
        IF(IGNDP(3).NE.0)THEN
          CALL BIMAGE(JSF,RK,RMK,C,CA,CAJ,2,IPAT)
          CALL BIMAGE(JSF,RK,RMK,C,CA,CAJ,3,IPAT)
        ENDIF
      ENDIF
      IF(NGNDP.GE.3)CALL BIMAGE(
$         JSF,RK,RMK,C,CA,CAJ,3,2,1,IPAT)
    ENDIF
  ENDIF
  RETURN
  END

```

A.64 PD3FAC

```

C=====
  SUBROUTINE PD3FAC(JSF,RK,RMK,C,CA,CAJ,IPAT)
C-----
C POTENTIAL INTEGRAL OF DYNAMIC 3-D GREEN'S FUNCTION OVER FACE
C INPUT:
C RK(J,I) I=1,2,3 DENOTES K* THE X,Y,Z COORDINATES OF THE JTH VERTEX
C OF THE SOURCE TRIANGLE.
C RMK(I) I=1,2,3 DENOTES K* THE X,Y,Z COORDINATES OF THE MATCH POINT.
C K= THE WAVE NUMBER.
C RATIO1,RATIO2,RATIO3 ARE TEST PARAMETERS THAT DETERMINE THE METHOD OF

```

```

C INTEGRATION.
C TO EXPLAIN THIS TEST DEFINE :
C RAT=(THE DISTANCE TO THE SOURCE TRIANGLE'S CENTROID FROM THE
C MATCH POINT)/THE SOURCE TRIANGLE'S MAXIMUM EDGE LENGTH.
C RATIO=RAT*RAT.
C IF RATIO.LE.RATIO1, 7 POINT QUADRATURE AND ANALYTICAL TREATMENT OF
C SINGULARITIES IS PERFORMED.
C IF RATIO1.LT.RATIO.LE.RATIO2 7 POINT QUADRATURE IS USED.
C IF RATIO2.LT.RATIO.LE.RATIO3 3 POINT QUADRATURE IS USED.
C IF RATIO3.LT.RATIO 1 POINT QUADRATURE IS USED.
C OUTPUT:
C TO DEFINE THE REMAINING OUTPUTS WE INTRODUCE THE NORMALIZED
C AREA COORDINATES (ZETA,XSI,ETA) FOR THE SOURCE TRIANGLE.
C (ZETA,XSI,ETA)=(1,0,0) CORRESPONDS TO VERTEX 1.
C (ZETA,XSI,ETA)=(0,1,0) CORRESPONDS TO VERTEX 2.
C (ZETA,XSI,ETA)=(0,0,1) CORRESPONDS TO VERTEX 3.
C ZETA=1-XSI-ETA.
C THE NORMALIZED SOURCE TRIANGLE IS DEFINED BY: (XSI,ETA) WHERE
C XSI VARIES FROM 0 TO 1-ETA AND ETA VARIES FROM 0 TO 1.
C THE REMAINING OUTPUTS ARE C AND CVEC WHERE
C C= THE DOUBLE INTEGRAL OVER THE NORMALIZED SOURCE TRIANGLE OF
C CEXP(-I*DK)/DK D(XSI) D(ETA)
C WHERE DK= K* THE DISTANCE FROM THE SOURCE POINT TO THE MATCH POINT.
C CVEC(I)=RK(1,I)*(C-CXSI-CETA)+RK(2,I)*CXSI+RK(3,I)*CETA I=1,2,3
C WHERE CXSI AND CETA ARE THE INTEGRALS OVER THE NORMALIZED SOURCE
C TRIANGLE OF:
C CEXP(-I*DK)/DK TIMES XSI AND ETA RESPECTIVELY D(XSI) D(ETA).
C
C THE VECTOR AND SCALAR POTENTIALS DUE TO A BASIS FUNCTION THAT
C FLOWS OUTWARD FROM THE NTH EDGE OF THE SOURCE TRIANGLE ARE GIVEN
C BY VEC POTENTIAL(I)=(MUO*RLK'SUB'N/(4*PI*K)*(CVEC(I)-RK(N,I)*C)
C I=1,2,3.
C SCALAR POTENTIAL=-RLK'SUB'N*C/(2*PI*OMEGA*EPSO*I) I=SQRT(-1.)
C RLK'SUB'N IS K* THE LENGTH OF THE NTH SIDE OF THE SOURCE TRIANGLE.
C-----
      IMPLICIT COMPLEX (C)
      DIMENSION RK(3,3),RMK(3),RCK(3),SINGV(3),RMC(3),DL(3,3)
      &           ,SINGT(3),SP2(3),SM2(3),CI(3),CA(3,3),CAJ(3)
      &           ,H(3),V(3),AL(3)
      COMMON/MINMAX/RLMINK,RLMAXK
      COMMON/TEST/RATIO1,RATIO2,RATIO3
      COMMON/PARA/DL,DET,H,AL
      DATA RATIO1,RATIO2,RATIO3/1.,9.,100./
C
      CRX(X)=CMPLX(X,0.)

```

```

C      JI=JI+1
C      IF(IPAT.GT.0)THEN
C      XSING=0.
C      IF(IPAT.EQ.1)IQUAD=3
C      IF(IPAT.EQ.2)IQUAD=1
C      ELSE
DO 2 J=1,3
RCK(J)=(RK(1,J)+RK(2,J)+RK(3,J))/3.
2      RMC(J)=RMK(J)-RCK(J)
RATIO=(RMC(1)*RMC(1)+RMC(2)*RMC(2)+RMC(3)*RMC(3))/(RLMAXK*RLMAXK)
IF(RATIO.LE.RATIO1)THEN
C      XSING=1.
C      IQUAD=7
ELSE
C      XSING=0.
C      IF(RATIO.LE.RATIO2)THEN
C      IQUAD=7
C      ELSEIF(RATIO.LE.RATIO3)THEN
C      IQUAD=3
C      ELSE
C      IQUAD=1
C      ENDIF
ENDIF
ENDIF
C
C      CALL FACQAD(JSF,IQUAD,XSING,RK,RMF,RLMIN,C,CIXSI,CIETA,CRX,CMM
&,IPAT)
C
C      CI(1)=C-CIXSI-CIETA
C      CI(2)=CIXSI
C      CI(3)=CIETA
C      IF(XSING.EQ.1.)THEN
DO 111 I=1,3
111      SP2(I)=0.
C      SM2(I)=0.
C
C      CALL PS3FAC(JSF,RK,SINGT,SP2,SM2)
C
DO 6 I=1,3
6      CI(I)=CI(I)+SINGT(I)
IF(JSF.NE.0)THEN
C      CPP=CPP+CRX(SP2(JSF))
C      CMM=CMM+CRX(SM2(JSF))
ENDIF

```

```

      ENDIF
      C=CI(1)+CI(2)+CI(3)
      DO 8 I=1,3
      IP1=MOD(I,3)+1
      IM1=MOD(IP1,3)+1
      DO 8 J=1,3
S      CA(I,J)=CI(IP1)*DL(IM1,J)-CI(IM1)*DL(IP1,J)
      IF(JSF.NE.0)THEN
      IP1=MOD(JSF,3)+1
      IM1=MOD(IP1,3)+1
      DO 20 J=1,3
C1     CAJ(J)=DET*H(JSF)*COP*DL(IM1,J)-CMM*DL(IP1,J)
      ENDIF
      RETURN
      END

```

A.65 BINAGE

```

-----
SUBROUTINE BINAGE (JSF,RK,RM,K,C,CA,CAD,I1,I2,I3,IPAT)
-----
C INPUT:
C   RK(I), I=1,2,3 DENOTES K*THE X,Y,Z COORDINATE OF THE ITH
C   VERTEX OF THE SOURCE TRIANGLE.
C   RM(I), I=1,2,3 DENOTE K* THE X,Y,Z COORDINATES OF THE MATCH POINT
C   RKL1,RKL2,RKL3 DENOTE K*THE LENGTHS OF THE EDGES
C   OPPOSITE THE 1,2,3 VERTICES OF THE SOURCE TRIANGLE.
C   IJ=1,2,3 DENOTES I) IMAGE PLANES IF IJ=0
C   THE X=0 GROUND PLANE IF IJ=1
C   THE Y=0 GROUND PLANE IF IJ=2
C   THE Z=0 GROUND PLANE IF IJ=3
C   I1>0
C   EITHER I1>I2>I3 OR I1>I2=0=I3
C OUTPUT:
C   THE SOURCE TRIANGLE IS IMAGED ABOUT:
C   (CASE I1>0 I2=I3=0) THE I1 IMAGE PLANE.
C   (CASE I1>I2>I3=0) THE I1 IMAGE PLANE, THEN REFLECTED
C   ABOUT THE I2 IMAGE PLANE.
C   (CASE I1>I2>I3) THE I1 IMAGE PLANE, THEN REFLECTED
C   ABOUT THE I2 AND I3 IMAGE PLANES.
C THIS NEW IMAGE TRIANGLE'S CONTRIBUTIONS TO CI ARE
C COMPUTED AND THE RESULTS ARE ADDED TO CI

```

```

C-----
      DIMENSION RK(3,3),RMK(3),IGNDP(3)
      COMPLEX CI,C,CPP,CMM,CII(3),CPPI,CMMI,
&          CA(3,3),CAJ(3),CAI(3,3),CAJI(3)
      COMMON/GPLANE/NGNDP,IGNDP
      CALL BSWTCH(RK,I1,I2,I3)
      CALL FACPAR(JSF,RK,RMK)
      CALL PD3FAC(JSF,RK,RMK,CI,CAI,CAJI,IPAT)
      CALL BSWTCH(RK,I1,I2,I3)
      CALL FACPAR(JSF,RK,RMK)
      SGN=IGNDP(I1)
      IF(I2.GT.0)THEN
        SGN=SGN*IGNDP(I2)
        IF(I3.GT.0)SGN=SGN*IGNDP(I3)
      ENDIF
      C=C+SGN*CI
      DO 10 I=1,3
      DO 10 J=1,3
10      CAI(I,J)=CAI(I,J)+SGN*CAI(I,J)
      IF(JSF.NE.0)THEN
      DO 20 J=1,3
20      CAJ(J)=CAJ(J)+SGN*CAJI(J)
      ENDIF
      RETURN
      END

```

A.66 BSWTCH

```

C=====
      SUBROUTINE BSWTCH(RK,I1,I2,I3)
C-----
C INPUT:
C   RK(J,I) I=1,2,3 ARE K* THE X,Y,Z COORDINATES OF THE JTH VERTEX
C   OF THE SOURCE TRIANGLE WHERE K=THE WAVE NUMBER.
C   I1>0 AND EITHER I1>I2>I3 OR I1>I2=I3=0.
C   I1,I2,I3 ARE TAKEN FROM THE SET: (0,1,2,3)
C
C OUTPUT:
C   FOR EACH VERTEX J=1,2,3 RK(J,I) IS SET TO -RK(J,I) FOR I RUNNING
C   OVER THE POSITIVE INTEGERS IN THE SET (I1,I2,I3).
C-----
      DIMENSION RK(3,3)
      DO 10 J=1,3

```



```

    RK(J,I1)=-RK(J,I1)
10  CONTINUE
    IF(I2.GT.0)THEN
        DO 20 J=1,3
            RK(J,I2)=-RK(J,I2)
20  CONTINUE
    IF(I3.GT.0)THEN
        DO 30 J=1,3
            RK(J,I3)=-RK(J,I3)
30  CONTINUE
    ENDIF
    RETURN
    END

```

A.67 FACQAD

```

C=====
C      SUBROUTINE FACQAD(JSF,IQUAD,XSING,RK,RMK,RLMINK,CI,CIXSI,CIETA,
C      & CPP,CMM,IPAT)
C-----
C GAUSSIAN QUADRATURE INTEGRAL OVER FACE
C INPUT:
C IQUAD EQUALS 1,3,OR 7 THE NUMBER OF INTEGRATION POINTS.
C XSING=0.(NO SINGULARITY EXTRACTION) XSING=1.(SINGULARITY EXTRACTION)
C RK(J,I) I=1,2,3 DENOTES K* THE X,Y,Z COORDINATES OF THE JTH VERTEX
C OF THE SOURCE TRIANGLE.
C RMK(I) I=1,2,3 DENOTE K* THE X,Y,Z COORDINATE OF THE MATCH POINT.
C RLMINK=K* THE LENGTH OF THE SHORTEST SIDE OF THE SOURCE TRIANGLE.
C OUTPUT:
C THE NORMALIZED AREA COORDINATES (XSI,ETA) AND THE NORMALIZED
C SOURCE TRIANGLE ARE DEFINED IN TRISC.
C DK=THE MAGNITUDE OF THE 3 DIMENSIONAL VECTOR.
C V(I)=(1-XSI-ETA)*RK(1,I)+XSI*RK(2,I)+ETA*RK(3,I)-RMK(I) I=1,2,3.
C CI,CIXSI,AND CIETA ARE THE INTEGRALS OVER THE NORMALIZED SOURCE
C TRIANGLE C OF 1.,XSI,ETA (RESPECTIVELY) TIMES
C CEXP(-I*DK)/DK D(XSI) D(ETA) WHERE I=SQRT(-1.).
C GAUSSIAN WEIGHTS FOR THE SOURCE TRIANGLE FOR IQUAD=3,7 ARE IN STRANG
C AND FIX, 'AN ANALYSIS OF THE FINITE ELEMENT METHOD', P.184;
C HOWEVER, THEIR FIRST WEIGHT IS WRONG.(7PT) OURS HAS BEEN CORRECTED
C SO THE INTEGRATION OVER A CONSTANT AND 1ST ORDER POLY IN ETA OR
C XSI IS EXACT.
C-----

```

```

IMPLICIT COMPLEX (C)
DIMENSION RK(3,3),RMK(3),XSI3(3),XSI7(7),ETA3(3),ETA7(7)
$      .WGHT7(7),CFP(3),CFM(3),CSP1(3),CSM1(3),CFP1(3),CFM1(3)
DATA XSI1,ETA1,WGHT1/.3333333333333333,.3333333333333333,.5/
DATA XSI3/.6666666666666667,.1666666666666667,.1666666666666667/
DATA ETA3/.1666666666666667,.6666666666666667,.1666666666666667/
DATA WGHT3/.1666666666666667/
DATA XSI7/.3333333333333333,.797426985353087,.101286507323456,
1 101286507323456,.470142064105115,.470142064105115,
2 .059715871789770/
DATA ETA7/.3333333333333333,.101286507323456,.797426985353087,
1 .101286507323456,.470142064105115,.059715871789770,
2 .470142064105115/
DATA WGHT7/.1125,.062969590272413,.062969590272413,
1 .062969590272413,.066197076394253,.066197076394253,
2 .066197076394253/
RLIMIT=1.E-5*RLMINK
IF(IQUAD.EQ.1)THEN
CALL INTGRN(JSF,RK,RMK,RLIMIT,XSING,XSI1,ETA1,
&      WGHT1,CI,CIXSI,CIETA,CFP,CFM,CSP1,CSM1,IPAT)
ELSE
IF(JSF.NE.0)THEN
DO 5 J=1,3
CFP1(J)=(0.,0.)
CFM1(J)=(0.,0.)
CFP(J)=(0.,0.)
5 CFM(J)=(0.,0.)
ENDIF
CI=(0.,0.)
CIXSI=(0.,0.)
CIETA=(0.,0.)
IF(IQUAD.EQ.3)THEN
DO 10 I=1,3
CALL INTGRN(JSF,RK,RMK,RLIMIT,XSING,XSI3(I),ETA3(I),
&      WGHT3,CC,CCXSI,CCETA,CFP1,CFM1,CSP1,CSM1,IPAT)
IF(JSF.NE.0)THEN
DO 15 J=1,3
CFP(J)=CFP(J)+CFP1(J)
15 CFM(J)=CFM(J)+CFM1(J)
ENDIF
CI=CI+CC
CIXSI=CIXSI+CCXSI
CIETA=CIETA+CCETA
10 CONTINUE
ELSE

```

```

DO 20 I=1,7
  CALL INTGRN(JSF,RK,RMK,RLIMIT,XSING,XSI7(I),ETA7(I),
&          WGHT7(I),CC,CCXSI,CCETA,CFP1,CFM1,CSP1,CSM1,IPAT)
  IF(JSF.NE.0)THEN
    DO 25 J=1,3
      CFP(J)=CFP(J)+CFP1(J)
25    CFM(J)=CFM(J)+CFM1(J)
    ENDIF
      CI=CI+CC
      CIXSI=CIXSI+CCXSI
      CIETA=CIETA+CCETA
20    CONTINUE
    ENDIF
  ENDIF
  IF(JSF.NE.0)THEN
    CPP=CFP(JSF)+CSP1(JSF)
    CMM=CFM(JSF)+CSM1(JSF)
  ENDIF
  RETURN
END

```

A.68 INTGRN

```

C=====
C          SUBROUTINE INTGRN(JSF,RK,RMK,RLIMIT,XSING,XSI,ETA,WGHT,CC,
&          CCXSI,CCETA,CFP,CFM,CSP1,CSM1,IPAT)
C-----
C INTEGRAND OF POTENTIAL INTEGRAL
C INPUT:
C RK(J,I) I=1,2,3 DENOTES K* THE X,Y,Z COORDINATES OF THE JTH VERTEX
C OF THE SOURCE TRIANGLE.
C RMK(I) I=1,2,3 DENOTES K* THE X,Y,Z COORDINATES OF THE MATCH POINT.
C RLIMIT=A SMALL CONSTANT * K* THE LENGTH OF THE SOURCE TRIANGLE'S
C SHORTEST SIDE.
C XSING=0. NO SINGULARITY EXTRACTION.
C XSING=1. SINGULARITY EXTRACTION.
C XSING MUST BE 1. IF MATCH POINT IS ON THE SOURCE TRIANGLE.
C XSI,ETA ARE NORMALIZED AREA COORDINATES OF THE SOURCE TRIANGLE.
C WGHT IS A WEIGHT FACTOR.
C OUTPUT:
C CC,CCXSI,CCETA= WGHT*(CEXP(-I*DK)-XSING)/DK *(1,XSI,ETA,RESPECTIVELY)
C WHERE I=SQRT(-1.).
C DK=THE MAGNITUDE OF THE VECTOR.

```

```

C VX=(1-XSI-ETA)*RK(1,1)+XSI*RK(2,1)+ETA*RK(3,1)-RMK(1)
C VY=(1-XSI-ETA)*RK(1,2)+XSI*RK(2,2)+ETA*RK(3,2)-RMK(2)
C VZ=(1-XSI-ETA)*RK(1,3)+XSI*RK(2,3)+ETA*RK(3,3)-RMK(3)
C-----
      IMPLICIT COMPLEX (C)
      DIMENSION RK(3,3),RMK(3),V(3),CFP(3),CFM(3),CSP1(3),CSM1(3)
      SIZE(X,Y,Z)=SQRT(X*X+Y*Y+Z*Z)
      ZETA=1.-XSI-ETA
      IF(IPAT.GT.0)THEN
        DO 2 J=1,3
2       V(J)=- (ZETA*RK(1,J)+XSI*RK(2,J)+ETA*RK(3,J))
          DK=V(1)*RMK(1)+V(2)*RMK(2)+V(3)*RMK(3)
          ELSE
            DO 3 J=1,3
3           V(J)=RMK(J)- (ZETA*RK(1,J)+XSI*RK(2,J)+ETA*RK(3,J))
              DK=SIZE(V(1),V(2),V(3))
              ENDIF
              C=CMPLX(0.,-DK)
              CDK=DEXP(C)
              IF(IPAT.GT.0)THEN
                CE=WGHT*CDK
              ELSEIF(DK.GT.RLIMIT)THEN
                CE=WGHT*(CDK-XSING)/DK
              ELSEIF(XSING.EQ.1.)THEN
                CE=(0.,-1.)*WGHT
              ELSE
                WRITE(4,99)
29          FORMAT(1X,'WARNING IN QUAD SINGULARITY WITH NO EXTRACTION')
                STOP
              ENDIF
C
C IF THERE ARE ANY JUNCTIONS, CALL INTJUN TO COMPUTE INTEGRATION DUE TO
C JUNCTION BASIS FUNCTION
C
      IF(JSF.NE.0.)THEN
        IF(IPAT.EQ.0)THEN
C
          CALL INTJUN(JSF,RK,RMK,V,DK,CDK,XSING,XSI,ETA,WGHT,
            & CFP,CFM,CSP1,CSM1,IPAT)
          ELSE
C COMPUTE VECTOR POTENTIAL FOR FAR FIELD
          CALL JKFPAT(JSF,RK,RMK,CDK,XSING,XSI,ETA,WGHT,
            & CFP,CFM,CSP1,CSM1)
C
        ENDIF

```

```

ENDIF
CC=CE
CCXSI=CE*XSI
CCETA=CE*ETA
RETURN
END

```

A.69 PS3FAC

```

C=====
C      SUBROUTINE PS3FAC(JSF,R,SINGT,SP2,SM2)
C-----
C POTENTIAL INTEGRAL OF SCALAR 3-D GREEN'S FUNCTION OVER FACE
C INPUT:
C EACH VECTOR SAY V,V(I) I=1,2,3 CORRESPONDS TO THE X,Y,AND Z
C COMPONENTS OF V RESPECTIVELY.
C R(J,I) J=1,2,3 CORRESPONDS TO VECTORS FROM THE ORIGIN TO THE
C 1,2,3 VERTICIES RESPECTIVELY OF THE SOURCE TRIANGLE.
C RM IS THE VECTOR FROM THE ORIGIN TO THE MATCH POINT.
C RMO IS THE PROJECTION OF RM INTO THE PLANE OF THE SOURCE TRIANGLE.
C UN IS THE UNIT NORMAL TO THE PLANE OF THE SOURCE TRIANGLE.
C IT IS DETERMINED BY THE RIGHT HAND RULE APPLIED TO VERTICIES 1,2,3.
C DPERP IS THE DISTANCE OF RM TO RMO (DPERP=GE.G).
C OUTPUT:
C SING=THE INTEGRAL OVER THE SOURCE TRIANGLE OF
C 1./ABS(VECTOR(RP-RM)) D(AREAP).
C SINGV(I) I=1,2,3 = THE X,Y,Z COMPONENTS OF THE INTEGRAL OVER THE
C SOURCE TRIANGLE OF VECTOR(RP-RMO)/ABS(VECTOR(RM-RP)) D(AREAP).
C RP=THE SOURCE VECTOR LOCATION.
C REFERENCE:'POTENTIAL INTEGRALS FOR UNIFORM AND LINEAR SOURCE
C DISTRIBUTIONS IN POLYHEDRAL DOMAINS'
C D.R.WILTON ET. AL. TO APPEAR IN IEEE. AP MARCH 1984.
C SING=THE SUM OVER THE SOURCE TRIANGLE EDGES OF:
C      (PO)HAT DOT (UU)HAT [PO*LN((RRP+ALP)/(RRM+ALM))
C      -D*(ATAN(PO*ALP/(PO**2+D**2+D*RRP))-ATAN(PO*ALM/(PO**2+D**2+D*
C      RRM)))]
C SINGV(I)(I=1,2,3)=THE X,Y,Z COMPONENTS OF THE SUM OVER THE EDGES
C OF THE SOURCE TRIANGLE OF
C .5*(UU)HAT *[(D**2+PO**2)*LN((RRP+ALP)/(RRM+ALM))+ALP*RRP-ALM*RRM]
C PO=THE PERPENDICULAR DISTANCE FROM RMO TO THE EDGE OF THE
C SOURCE TRIANGLE.
C (PO)HAT IS THE UNIT VECTOR THAT POINTS FROM RMO TOWARDS THE EDGE.
C (UU)HAT IS THE OUTWARD POINTING UNIT NORMAL OF THE SOURCE TRIANGLE

```

```

C THAT LIES IN THE PLANE OF THE SOURCE TRINAGLE.
C D=DPERP.
C ALP=THE DIRECTED DISTANCE BETWEEN THE PROJECTION OF RM ONTO THE
C EDGE AND THE (PLUS) VERTEX.
C ALM=THE DIRECTED DISTANCE BETWEEN THE PROJECTION OF RM ONTO THE
C EDGE AND THE (MINUS) VERTEX.
C FOR AN EDGE RUNNING FROM VERTEX IVTX TO VERTEX IVTX1,
C THE PLUS VERTEX IS IVTX1 AND THE MINUS VERTEX IS IVTX.
C REP=SQRT(D**2+PO**2+ALP**2).
C REM=SQRT(D**2+PO**2+ALM**2).
-----
C IMPLICIT COMPLEX (C)
C
C DIMENSION R(3,3),RMO(3),UN(3),SINGV(3),VL(3,3),UH(3,3),RI(3,3)
$,H(3),SINGT(3),VPO(3,3),VH(3,3),DL(3,3),AL(3),F(3,3),SXI(3),
$
$,PH(3),RR2(3),RR(3),SPR(3),SML(3),HPS(3)
COMMON/JUN/RI,P,PH,RR,RR2,RIL,URI,VH
COMMON/S2/HPR,HMR
COMMON/CONJ/XSING2
COMMON/FARA/DL,DET,H,AL,UL,UH,RV,UN,DPERP
DPERP=DPERP
ISM=JSF
SING=0.
DO 2 J=1,3
SINGV(J)=0.
DO 10 I=1,3
IP1=MOD(I,3)+1
IM1=MOD(I+1,3)+1
PO=UH(I,1)*(R(IP1,1)-RMO(1))+UH(I,2)*(R(IP1,2)-RMO(2))
$
+UH(I,3)*(R(IP1,3)-RMO(3))
DO 4 J=1,3
4 VPO(I,J)=PO*UH(I,J)
SGN=1.
IF(PO.LT.0.)THEN
SGN=-1.
PO=-PO
ENDIF
ALP=UL(I,1)*(R(IM1,1)-RMO(1))+UL(I,2)*(R(IM1,2)-RMO(2))
$
+UL(I,3)*(R(IM1,3)-RMO(3))
ALM=UL(I,1)*(R(IP1,1)-RMO(1))+UL(I,2)*(R(IP1,2)-RMO(2))
$
+UL(I,3)*(R(IP1,3)-RMO(3))
C
C CALL CAS(PO,DPERP,ALP,ALM,VALA,VALL)
C
SING=SING+SGN*VALA

```

```

        DC 6 J=1,3
6       SINGV(J)=SINGV(J)+UH(I,J)*VALL
10      CONTINUE
C
C
        DO 8 I=1,3
          HPS(I)=0.
          DC 9 J=1,3
9         HPS(I)=HPS(I)+(UH(I,J)*(VPO(I,J)*SING-SINGV(J)))
8         SINGT(I)=HPS(I)/DET/H(I)
C
C THE INTEGRATION OF THE TAYLOR SERIES EXPANSION OF THE INTEGRAND
C OF THE JUNCTION POTENTIAL INTEGRATION AT THE JUNCTION VERTEX
C SINGULARITY
C
C UHS= UNIT VET H DOT VET IO
C SINGV(J)= VET IO = INTEGRAL OF
C
        IF(JSF.NE.0.AND.XSING2.NE.1.) THEN
          I=JSF
          IP1=MOD(I,3)+1
          IM1=MOD(I+1,3)+1
          SXI(IP1)=HPS(IP1)/H(IP1)
          SXI(IM1)=HPS(IM1)/H(IM1)
          HLP=0.
          HLM=0.
          DO 24 J=1,3
            HLP=HLP+UH(I,J)*DL(IP1,J)
            HLM=HLM+UH(I,J)*DL(IM1,J)
24         CONTINUE
          SX=2./RR(I)*(RR(I)*SING-HLM*SXI(IP1)+HLP*SXI(IM1))
          RCON=1./RR2(I)/DET
          SP2(I)=RCON*(SXI(IP1)-HPR*SX)
          SM2(I)=RCON*(SXI(IM1)-HMR*SX)
          ELSE
            DO 30 II=1,3
              SP2(II)=0.
30             SM2(II)=0.
          ENDIF
          RETURN
        END

```

A.70 CAS

```

C=====
C      SUBROUTINE CAS(PO,DPERP,ALP,ALM,VALA,VALL)
C-----
C      . . . CONSTANTS -- RRO IS DIST. TO LINE? RRP AND RRM, TO VERTICES . . .
C INPUT:
C PO=THE PERPENDICULAR DISTANCE FROM RRO TO THE EDGE OF THE SOURCE
C TRIANGLE UNDER CONSIDERATION. DPERT IS THE PERPENDICULAR DISTANCE OF
C RRM TO RRO AS DEFINED IN ISING. ALP IS THE DIRECTED DISTANCE FROM
C THE PROJECTION OF RRM ONTO THE EDGE OF THE SOURCE TRIANGLE.
C VALA=[PO*LN(RRP+ALP)/(RRM+ALM)]
C      -DPERP*(ATAN(PO*ALP/(PO**2+DPERP**2+DPERP*RRP)
C      -ATAN(PO*ALM/(PO**2+DPERP**2+DPERP*RRM))))]
C CVALL=.5*[(DPERP**2+PO**2)*LN((RRP+ALP)/(RRM+ALP)+ALP*RRP-ALM*RRM)]
C THE NOTATION IN THIS ROUTINE IS THE SAME AS IN CALLING ROUTINE ISING.
C-----
      RROSQ=DPERP*DPERP+PO*PO
      RRO=SQRT(RROSQ)
      RRP=SQRT(RROSQ+ALP*ALP)
      RRM=SQRT(RROSQ+ALM*ALM)
      AL=ALP-ALM
      RATIO=RRO/AL
      IF (RATIO .GT. 1E-4) THEN
        PERPD=ABS(DPERT)
        ALGTRM=ALOG((RRP+ALP)/(RRM+ALM)
        ARGTRM=PO*ALP/(RROSQ+PERPD*RRP)
        ARGTRM=PO*ALM/(RROSQ+PERPD*RRM)
        VALA=PO*ALGTRM-PERPD*(ATAN(ARGTRM)-ATAN(ARGTRM))
        VALL=.5*(RROSQ*ALGTRM+ALP*RRP-ALM*RRM)
      ELSE
        VALA=0.
        VALL=.5*(ALP*RRP-ALM*RRM)
      ENDIF
      RETURN
      END

```

A.71 JKFPAT

```

C=====
C      SUBROUTINE JKFPAT(JSF,RK,RMK,CDK,XSING,XSI,ETA,WGHT,
C      & CFP,CFM,CSP,CSM)
C-----

```



```

* COMPUTE QUANTITIES FOR CALCULATION OF THE VECTOR POTENTIAL
C ASSOCIATE WITH JUNCTION BASIS FUNCTION BY USING FAR FIELD KERNEL
C RK(I,J) J=1,2,3 DENOTES K* THE X,Y,Z COORDINATES OF THE ITH VERTEX OF
C THE SOURCE TRIANGLE. (K=THE WAVE NUMBER)
C RMK(I) I=1,2,3 DENOTES THE X,Y,Z COORDINATES OF THE MATCH POINT.
C H(I) IS THE LENGTH OF THE HEIGHT VECTOR.
C DL(I,J) IS THE VECTOR FROM THE I+1TH VERTEX TO THE ITH VERTEX.

```

```

-----
      IMPLICIT COMPLEX (C)
      DIMENSION RK(3,3),RMK(3),XI(3),RI(3,3),H(3),
&DL(3,3),CFP(3),CFM(3),CSP(3),CSM(3)
      COMMON/JUN/RI
      COMMON/PARA/DL,DET,H
      CRX(X)=CMPLX(X,0.)
      CIX(X)=CMPLX(0.,X)
      SIZE(X,Y,Z)=SQRT(X*X+Y*Y+Z*Z)
      ZETA=1.-XSI-ETA
      XI(1)=ZETA
      XI(2)=XSI
      XI(3)=ETA
      DO 20 I=1,3
      CFP(I)=(0.,0.)
20    CFM(I)=(0.,0.)
      I=JSF
      DO 10 I=1,3
      IP1=MOD(I,3)+1
      IM1=MOD(I+1,3)+1
      C HS2=(H(I)*(1-XI(I)))**2
      C
      RRI=0.
      DO 30 J=1,3
30    RRI=RRI+RMK(J)*RI(I,J)
      HS=H(I)*(1.-XI(I))
      HS2=HS*HS
      C
      CF=1./HS2*(CDK-CRR)
      CFP(I)=XI(IP1)*CF*WGHT
      CFM(I)=XI(IM1)*CF*WGHT
      C
      C CDK=EXP(-JR)
      C
      C THE INTEGRATION OF THE TAYLOR SERIES EXPANSION OF THE INTEGRAND
      C AT THE JUNCTION VERTEX SINGULARITY
      C
      CSP(I)=1./H(I)/H(I)*CRR*0.5

```

```

      CSM(I)=CSP(I)
10    CONTINUE
      RETURN

```

A.72 INTJUN

```

-----
      SUBROUTINE INTJUN(JSF,RK,RMK,V,DK,CDK,XSING,XSI,ETA,WGHT,
&                      CFP,CFM,CSP1,CSM1,IPAT)
-----
* COMPUTE QUANTITIES FOR CALCULATION OF THE VECTOR AND SCALAR POTENTIALS
& ASSOCIATE WITH JUNCTION BASIS FUNCTION
& RK(I,J) J=1,2,3 DENOTES K* THE X,Y,Z COORDINATES OF THE ITH VERTEX OF
& THE SOURCE TRIANGLE. (K=THE WAVE NUMBER)
& RMK(I) I=1,2,3 DENOTES THE X,Y,Z COORDINATES OF THE MATCH POINT.
& UN(I) I=1,2,3 DENOTES THE X,Y,Z COORDINATES OF THE UNIT NORMAL VECTOR
& TO THE PLANE OF THE SOURCE TRIANGLE.
& UL(I,J) IS THE UNIT VECTOR FROM THE I+1TH VERTEX TO THE ITH VERTEX
& UH(I,J) IS THE UNIT HEIGHT VECTOR WHICH IS NORMAL TO UL(I,J)
& UH = UL X UN
& H(I) IS THE LENGTH OF THE HEIGHT VECTOR.
& DL(I,J) IS THE VECTOR FROM THE I+1TH VERTEX TO THE ITH VERTEX.
& AL(I) IS THE LENGTH OF THE DL(I,J)
& RMKO= THE PROJECTION OF RMK ONTO THE PLANE OF THE SOURCE TRIANGLE
& DPERFK= K* THE PERPENDICULAR DISTANCE FROM THE MATCH POINT TO THE
& PLANE OF THE TRIANGLES
-----
      IMPLICIT COMPLEX (C)
      DIMENSION RK(3,3),RMK(3),V(3),XI(3),UV(3),RI(3,3),URI(3,3),
&AL(3),UL(3,3),UH(3,3),H(3),P(3,3),VH(3,3),CT(3),CS(3),
&DL(3,3),RMKO(3),PH(3),RR(3),RR2(3),CFP(3),CFM(3),RIL(3),VR(3),
&CSP1(3),CSM1(3)
      COMMON/CONJ/XSING2
      COMMON/S2/HPR,HMR
      COMMON/JUN/RI,P,PH,RR,RR2,RIL,URI,VH
      COMMON/PARA/DL,DET,H,AL,UL,UH,RMKO
      CRX(X)=CMPLX(X,0.)
      CIX(X)=CMPLX(0.,X)
      SIZE(X,Y,Z)=SQRT(X*X+Y*Y+Z*Z)
      ZETA=1.-XSI-ETA
      XI(1)=ZETA
      XI(2)=XSI
      XI(3)=ETA

```

```

      DO 20 I=1,3
      CFP(I)=(0.,0.)
20    CFM(I)=(0.,0.)
      I=JSF
C     DO 10 I=1,3
      IP1=MCD(I,3)+1
      IM1=MCD(I+1,3)+1
C
C     RI(I,J) ARE THE VECTOR FROM THE MATCH POINT TO THE ITH VERTEX
C     OF THE SOURCE TRIANGLE.
C     RIL IS THE LENGTH OF RI
C     URI(I,J) ARE THE UNIT VECTOR OF RI(I,J)
C     RIH= UNIT VECTOR OF RI DOT UNIT VECTOR OF HEIGHT
C     RR(I)= RI*(UNIT VET RI DOT UNIT VET H)
C
C     DRP= VET L(I+1) DOT UNIT VET RI
C     DRM= VET L(I-1) DOT UNIT VET RI
C     HS2=(H(I)*(1-XI(I)))**2
C
      DRP=0.
      DRM=0.
      DO 30 J=1,3
      DRP=DRP+DL(IP1,J)*URI(I,J)
30    DRM=DRM+DL(IM1,J)*URI(I,J)
      HS=H(I)*(1-XI(I))
      HS2=HS*HS
C
C     CRK=EXP(-JRI)/RI
C     CR2=(1+JRI)/RI
C
      CRK=CEXP(CIX(-RIL(I)))/RIL(I)
      CR2=(CRX(1.)+CIX(RIL(I)))/RIL(I)
      WH=WGHT
C
C     CDK=EXP(-JR)
C     CF IS THE FIRST TERM OF THE NUMERICAL INTEGRAL OF THE POTENTIAL
C     INTEGRAL EQUATION
C
      RATIO=DK/AL(I)
C
C     IF R=0 TAKE THE LIMIT VALUE OF THE INTEGRAL FUNCTION
C     TO AVOID OVERFLOW
C
      XDK=0.
      IF(DK.LT.1.E-7) THEN

```


AD-A200 315

JUNCTION CODE USER'S MANUAL ELECTROMAGNETIC SCATTERING
AND RADIATION BY A. (U) HOUSTON UNIV ELECTROMAGNETICS
LAB D R WILTON ET AL. AUG 88 TR-87-18 NOSC-ID-1324
NG6001-85-D-0203 F/G 28/14

3/3

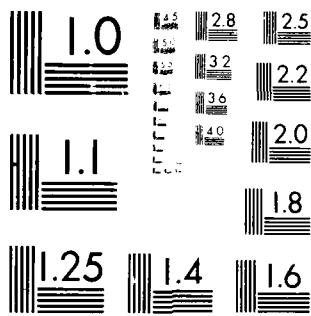
UNCLASSIFIED

NL

END

DATE

12/88



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

```

HLM=0.
HPR=0.
HMR=0.
GPM=0.
DO 70 J=1,3
GPM=GPM+UH(I,J)*V(J)
HPR=HPR+UH(IP1,J)*RI(I,J)
70 HMR=HMR+UH(IM1,J)*RI(I,J)
HPR=HPR/H(IP1)
HMR=HMR/H(IM1)
C
C RR(I)= RI*(UNIT VET RI DOT UNIT VET H)
C
GPM=2./RR(I)*GPM
YP=HPR*GPM
YM=HMR*GPM
GP=(XI(IP1)-YP)/RR2(I)/DK
GM=(XI(IM1)-YM)/RR2(I)/DK
C1=XI(IP1)*CF
C2=XI(IM1)*CF
C
C CFP,CFM ARE THE NUMERICAL INTEGRATIONS
C IN WHICH THE SINGULARITIES AT JUNCTION VERTEX AND R=0
C HAVE BE SUBTRACTED OUT
C
CFP(I)=(C1-CRX(GP))*WH
CFM(I)=(C2-CRX(GM))*WH
ENDIF
10 CONTINUE
RETURN
END
END

```

A.73 POTWIR

```

C=====
SUBROUTINE POTWIR(SSPOT,VAP,VAM,IPAT)
C-----
C POTENTIAL INTEGRAL OVER WIRE SEGMENT
C INPUT:
C RSK(J) ARE K* THE X,Y,Z COORDINATES OF THE CENTER POINT
C OF THE SOURCE SEGMENT WHERE K=THE WAVE NUMBER.
C RMK(I) I=1,2,3 DENOTE THE X,Y,Z COORDINATES OF THE MATCH POINT.

```

```

C      CONSTANTS PASSED THROUGH COMMON/GPLANE/
C      NGNDP=0,1,2,OR 3 THE NUMBER OF WIMAGE PLANES.
C      IGNDP(1)=0 NO GROUND PLANES;
C              -1 A P.E.C. GROUND PLANE;
C              1 A P.M.C. GROUND PLANE.
C              I=1,2,3 DENOTES THE X=0,Y=0,ANDZ=0 GROUND PLANES.
C      OUTPUT:
C      SSPOT: SCALAR POTENTIAL
C      VAP,VAM: VECTOR POTENTIALS
C-----
COMPLEX CSPW,VAP(3),VAM(3),SSPOT
INTEGER IGNDP(3)
COMMON/GPLANE/NGNDP,IGNDP
CALL PDWSEG(SSPOT,VAP,VAM,IPAT)
IF(NGNDP.GT.C)THEN
  DO 20 I=1,3
    IF(IGNDP(I).NE.0)CALL WIMAGE(SSPOT,VAP,VAM,I
    $      ,0,0,IPAT)
20  CONTINUE
IF(NGNDP.LE.2)THEN
  IF(IGNDP(1).EQ.0)THEN
    CALL WIMAGE(SSPOT,VAP,VAM,3,1,0,IPAT)
  ELSEIF(IGNDP(2).EQ.0)THEN
    CALL WIMAGE(SSPOT,VAP,VAM,3,1,0,IPAT)
  ELSE
    CALL WIMAGE(SSPOT,VAP,VAM,2,1,0,IPAT)
  IF(IGNDP(3).NE.0)THEN
    CALL WIMAGE(SSPOT,VAP,VAM,3,2,0,IPAT)
    CALL WIMAGE(SSPOT,VAP,VAM,3,1,0,IPAT)
  ENDIF
ENDIF
IF(NGNDP.GE.3)CALL WIMAGE(SSPOT,VAP,VAM,3,2,1,IPAT)
ENDIF
ENDIF
RETURN
END

```

A.74 WIMAGE

```

C=====
C      SUBROUTINE WIMAGE(SSPOT,VAP,VAM,I1,I2,I3,IPAT)
C-----
C INPUT:

```



```

C   RSK(J) ARE K* THE X,Y,Z COORDINATES OF THE CENTER POINT
C   OF THE SOURCE SEGMENT WHERE K=THE WAVE NUMBER.
C   RMK(I) I=1,2,3 DENOTE K* THE X,Y,Z COORDINATES OF THE MATCH POINT.
C   IJ J=1,2,3 DENOTES : 0 IMAGE PLANES IF IJ=0
C                       THE X=0 GROUND PLANE IF IJ=1
C                       THE Y=0 GROUND PLANE IF IJ=2
C                       THE Z=0 GROUND PLANE IF IJ=3
C   I1>0
C   EITHER I1>I2>I3 OR I1>I2=0=I3
C
C   OUTPUT:
C   THE SOURCE SEGMENT IS IMAGED ABOUT:
C   (CASE I1>0 I2=I3=0) THE I1 IMAGE PLANF
C   (CASE I1>I2>I3=0)  THE I1 IMAGE PLANE, THEN REFLECTED
C                       ABOUT THE I2 IMAGE PLANE.
C   (CASE I1>I2>I3)   THE I1 IMAGE PLANE, THEN REFLECTED
C                       ABOUT THE I2 AND I3 IMAGE PLANES.
C   THIS NEW IMAGE SEGMENT'S CONTRIBUTIONS TO POTENTIALS ARE
C   COMPUTED AND THE RESULTS ARE ADDED TO THEM
C-----
C   COMPLEX  CSPW,VAP(3),VAM(3),SSPOT,VAPI(3),VAMI(3),SSPOTI
C   INTEGER  IGNDP(3)
C   COMMON/GPLANE/NGNDP,IGNDP
C   CALL WSWTCH(I1,I2,I3)
C   CALL PDWSEG(SSPOTI,VAPI,VAMI,IPAT)
C   CALL WSWTCH(I1,I2,I3)
C   SGN=IGNDP(I1)
C   IF(I2.GT.0)THEN
C     SGN=SGN*IGNDP(I2)
C     IF (I3.GT.0)SGN=SGN*IGNDP(I3)
C   ENDIF
C     DO 5 J=1,3
C       VAP(J)=SGN*VAPI(J)+VAP(J)
C       VAM(J)=SGN*VAMI(J)+VAM(J)
5   CONTINUE
C     SSPOT=SGN*SSPOTI+SSPOT
C   RETURN
C   END

```

A.75 WSWTCH

```

C=====
C   SUBROUTINE WSWTCH(I1,I2,I3)

```

```

-----
C INPUT:
C RSK(J) ARE K* THE X,Y,Z COORDINATES OF THE CENTER POINT
C   OF THE SOURCE SEGMENT WHERE K=THE WAVE NUMBER.
C SH(J) ARE K* THE X,Y,Z COORDINATES OF THE UNIT VECTOR
C   OF THE SOURCE SEGMENT WHERE K=THE WAVE NUMBER.
C I1>0 AND EITHER I1>I2>I3 OR I1>I2=I3=0.
C I1,I2,I3 ARE TAKEN FROM THE SET: (0,1,2,3)
C
C OUTPUT:
C FOR J=1,2,3 RSK(J) IS SET TO -RSK(J) FOR I RUNNING
C OVER THE POSITIVE INTEGERS IN THE SET (I1,I2,I3).
-----
      DIMENSION SH(3),RSK(3),RMK(3)
      COMMON/WKERNL/RSK,SH,RMK,RADSK,RADSKS
      RSK(I1)=-RSK(I1)
      SH(I1)=-SH(I1)
      IF(I2.GT.0)THEN
         RSK(I2)=-RSK(I2)
      SH(I2)=-SH(I2)
      IF(I3.GT.0)THEN
         RSK(I3)=-RSK(I3)
      SH(I3)=-SH(I3)
      ENDIF
      ENDIF
      RETURN
      END

```

A.76 PDWSEG

```

=====
      SUBROUTINE PDWSEG(SSPOT,VAP,VAM,IPAT)
-----
C POTENTIAL INTEGRAL OF DYNAMIC WIRE GREEN'S FUNCTION OVER SEGMENT
      COMPLEX VAP(3),VAM(3),SSFOT,CSPW,ANS(3)
      > ,CKRED,CKTOT,CKSELF,CANS,CPATT
      DIMENSION IGNDP(3),RMK(3),SH(3),RSK(3)
      REAL LAMBDA,K,MU,IMP
      COMMON/MEDIUM/DEG2RAD,EPSLON,MU,IMP,SL,PI
      COMMON/POT/NM,NS,DEL,DELS,DELH,DELRH,RVPW,CSPW,RADMKK
      COMMON/WAVE/OMEGA,LAMBDA,K
      COMMON/WKERNL/RSK,SH,RMK,RADSK,RADSKS
      COMMON/WKER/DPAR,RHO,RHOPR,RHOPRS,RHUMRS

```

```

COMMON/WIRE/IQUADW
COMMON/WIRSLF/IQWS
COMMON/GPLANE/NGNDP,IGNDP
EXTERNAL CKRED,CKTOT,CKSELF,CPATT
IQUAD=IQUADW
IJUAWS=IQWS
DELI=DEL
RADMKS=RADMKK
C
* COMPUTE QUANTITIES FOR CALCULATION OF THE VECTOR AND SCALAR POTENTIALS
C
  IF(IPAT.GT.0)THEN
    ROS=100.
    IQUAD=4
    CALL SEGQAD(CPATT,DELI,IQUAD,ANS)
  ELSE
    ROS=(RSK(1)-RMK(1))*2+(RSK(2)-RMK(2))*2+(RSK(3)-RMK(3))*2
    IF(IPAT.EQ.0.AND.ROS.LE.3.9*DELS)THEN
C
* SINGULAR TERM TREATED ANALYTICALLY.
C
      DPAR=(RSK(1)-RMK(1))*SH(1)+(RSK(2)-RMK(2))*SH(2)+
      (RSK(3)-RMK(3))*SH(3)
      DPERPS=ROS-DPAR*DPAR
      RHO=SQRT(DPERPS+RADMKS)
      RHOPR=RHO+RADSK
      RHOMR=RHO-RADSK
      RHOPRS=RHOPR*RHOPR
      RHOMRS=RHOMR*RHOMR
      IF(NM.EQ.NS)THEN
        DELRHI=DELRH
        DELHI=DELH
        IF(DELH.GT.DELRH)THEN
          CALL SGQADS(CKTOT,DELRHI,DELHI,IQUAWS,CANS)
          ANS(1)=CANS
          XU=DELRHI
        ELSE
          XU=DELHI
          ANS(1)=(0.,0.)
        ENDIF
        CALL SGQADS(CKSELF,0.,XU,IQUAWS,CANS)
        XUS=XU*XU
        SIN1=XU*ALOG((XUS+RHOMRS)/(XUS+RHOPRS))
        IF(RHOMR.EQ.0.)THEN
          SINGT=SIN1-2.*RHOPR*ATAN(XU/RHOPR)

```

```

ELSE
  SINGT=SIN1
  > +2.*(RHOMR*ATAN(XU/RHOMR)-RHOPR*ATAN(XU/RHOPR))
ENDIF
ANS(1)=2.*(ANS(1)+CANS-SINGT/(RHOPR*PI))
C BY SYMMETRY
ANS(2)=.5*ANS(1)
ANS(3)=ANS(2)
ELSE
  CALL SEQQAD(CKTOT,DELI,IQUAD,ANS)
ENDIF
ELSE
  CALL SEQQAD(CHRED,DELI,IQUAD,ANS)
ENDIF
ENDIF
)
* FILL WIRE PORTION OF THE IMPEDANCE MATRIX
* COMPUTE VECTOR AND SCALAR POTENTIAL CONTRIBUTIONS
* DUE TO THE SOURCE SEGMENT.
)
SERDOT=RSPW*ANS(1)/(DELI/K)
ANS(2)=RVPW*ANS(2)
ANS(3)=RVPW*ANS(3)
001 FORMAT(X,'AN',I4,GE11.3)
DO 10 J=1,3
  VAP(J)=ANS(2)*SH(J)
  VAM(J)=ANS(3)*SH(J)
10 CONTINUE
RETURN
END

```

A.77 CKTOT

```

=====
      COMPLEX FUNCTION CKTOT(SP)
C-----
* INPUT:
* SP=K*THE DISTANCE OF THE SOURCE POINT FROM THE CENTROID OF THE SOURCE
* SEGMENT. SP IS POSITIVE IF THE DISTANCE IS TOWARDS THE SOURCE
* SEGMENT'S ENDPOINT AND NEGATIVE IF TOWARDS THE INITIAL POINT.
* RADSK =K*THE SOURCE SEGMENT RADIUS.
* RADSKS=RADSK*RADSK
* DPAR=(RSK(1)-RMK(1))*SH(1)+(RSK(2)-RMK(2))*SH(2)+

```



```

* RHO=SQRT(DPAR**2+RADMKS),WHERE RADMKS=(K* THE RADIUS OF
* THE MATCH SEGMENT)**2
* RHOPRS=(RHO+RADSK)**2
* RHOMRS=(RHO-RADSK)**2
* OUTPUT:
* CKSELF IS THE TOTAL KERNEL MINUS THE SINGULAR PART OF THE ELLIPTIC
* INTEGRAL CONTRIBUTION.
* A REDUCED KERNEL APPROXIMATION IS USED FOR THE SMOOTH
* TERM (CEXP((O.,-R)-1.)/R).

```

```

-----
C
  DIMENSION RMK(3),SH(3),RSK(3)
  COMPLEX CBNDR
  COMMON /WKERL/RSK,SH,RMK,RADSK ,RADSKG
  COMMON/WKER/DPAR,RHO,RHOPR,RHOPRS,RHOMRS
  DATA PIDTWO /1.57079632679489/
C
  SPS=SP*SP
  R=SQRT(SPS+RADSKS)
C
* CBNDR IS THAT PORTION OF THE KERNEL TO WHICH THE REDUCED KERNEL
* APPROXIMATION IS APPLIED.
C
  CBNDR=(CEXP(CMPLX(O.,-R))-1.)/R
  TERM=SPS
  DENOMS=TERM+RHOPRS
  DENOM=SQRT(DENOMS)
  BETA1=(TERM+RHOMRS)/DENOMS
  ELIPT=(ELIC1K(BETA1)/DENOM+.5*ALOG(BETA1)/RHOPR)/PIDTWO
  CKSELF=CBNDR+ELIPT
  RETURN
  END

```

A.79 CKRED

```

=====
  COMPLEX FUNCTION CKRED(SP)
-----
C
* INPUT:
* RMK(J),J=1,2,3=K* THE X,Y,Z COMPONENTS OF THE MATCH POINT.
* SH(J) J=1,2,3 = THE X,Y,Z COMPONENTS OF THE UNIT VECTOR POINTING IN
* THE SAME DIRECTION AS THE SOURCE SEGMENT.
* RSK(J)= K* THE X,Y,Z COORDINATES OF THE THE SOURCE SEGMENT CENTROID
* RADSK=K * THE SOURCE SEGMENT RADIUS.

```

```

* RADSKS=RADSK*RADSK
* SP IS K* THE DISTANCE ALONG THE SOURCE SEGMENT THAT RPRIME IS FROM
* THE SOURCE SEGMENT CENTROID. A POSITIVE DISTANCE IS TOWARDS THE
* ENDPOINT OF THE SOURCE SEGMENT, A NEGATIVE DISTANCE IS TOWARDS THE
* INITIAL POINT OF THE SOURCE SEGMENT.
* OUTPUT:
* CKMN=THE REDUCED KERNEL EVALUATED AT R=THE MATCH POINT AND
* RPRIME(J)=RSK(J)-SP*SH(J).

```

```

-----
C      DIMENSION RMK(3),SH(3),RSK(3),D(3)
      COMMON/WKERNL/RSK,SH,RMK,RADSK,RADSKS
C
      DO 5 J=1,3
        D(J)=RMK(J)-RSK(J)-SP*SH(J)
      CONTINUE
      R=SQRT(D(1)*D(1)+D(2)*D(2)+D(3)*D(3)+RADSKS)
      CKRED=CEXP(CMPLX(0.,-R))/R
      RETURN
      END

```

A.80 ELIC1K

```

-----
C      FUNCTION ELIC1K(AM1)
-----
* COMPLETE ELLIPTIC INTEGRAL OF THE FIRST KIND K(M), AS DEFINED IN JOHN
* THE REFERENCE BELOW, WHERE AM1=1-M
*
* REFERENCE: HANDBOOK OF MATHEMATICAL FUNCTIONS
* ABRAMPWITZ AND STEGUN
*
* EQUATION 17.3.34
-----
C      DATA AO,A1,A2,A3,A4,B0,B1,B2,B3,B4/
      >1.38629436112,.09666344259,.03590092393,.03742563713,.01451196212,
      >.500000000000,.12498593597,.06880248576,.03328355346,.00441787012/
C
      IF(AM1 .LT. 1.E-18)THEN
        A=AM1*A1+AO
        B=AM1*B1+BO
      ELSEIF(AM1 .LT. 1.E-12)THEN
        A=AM1*(AM1*A2+A1)+AO
        B=AM1*(AM1*B2+B1)+BO

```

```
ELSEIF(AM1 .LT. 1.E-9)THEN
  A=AM1*(AM1*(AM1*A3+A2)+A1)+A0
  B=AM1*(AM1*(AM1*B3+B2)+B1)+B0
ELSE
  A=AM1*(AM1*(AM1*(AM1*A4+A3)+A2)+A1)+A0
  B=AM1*(AM1*(AM1*(AM1*B4+B3)+B2)+B1)+B0
ENDIF
ELIC1K=A-B*ALOG(AM1)
RETURN
END
```


INITIAL DISTRIBUTION

U.S. NAVY

Office of Naval Research
Arlington, VA 22203-5000
ONNR-211

Naval Weapons Center
China Lake, CA 93555
Code 35203

David Taylor Research Center
Bethesda, MD 20084-5000
Code 0119

Pacific Missile Test Center
Point Mugu, CA 93042
Code 3242

Naval Postgraduate School
Monterey, CA 93940
Code 62AB

U.S. ARMY

U.S. Army Communications Electronics Command
Fort Monmouth, NJ 07703-5202
AMSEL-COM-TN-4

U.S. Army Information Systems Engineering Installation Activity
Fort Huachuca, AZ 85613-7300
ASBH-SET-P

U.S. AIR FORCE

Rome Air Development Center
Hanscom Air Force Base, MA 01731

OTHER

Lawrence Livermore Laboratory
Livermore, CA 94550

Defense Technical Information Center
Alexandria, VA 22314

DATE
L MED
88