UCID- 18834

NUMERICAL ELECTROMAGNETICS CODE (NEC) - METHOD OF MOMENTS

PART II: PROGRAM DESCRIPTION - CODE

G. J. Burke
A. J. Poggio

January 1981

Lawrence Livermore Laboratory

This is an informal report intended primarily for internal or limited external distribution. The opinions and conclusions stated are those of the author and may or may not be those of the Laboratory.

Work performed under the auspices of the U.S. Department of Energy by the Lawrence Livermore Laboratory under Contract W-7405-Eng-48.

DISCLAIMER (From the original NEC document)

==

The NEC2 documentation is composed of three sections

Part I: Program Description - Theory
Part II: Program Listing
Part III: User's guide

And may be obtained from the
National Technical Information Service (U.S. Department of Commerce)
5285 Port Royal Road
Springfield, VA 22161

Fax: +1 (703) 321 8547
Tel: +1 (703) 487 4679
http://www.ntis.gov

Ordering codes: AD-A075 209, AD-A075 289 and AD-A075 460 (not sure which is which)

==

April 1998

This document (Part II) was scanned at a resolution of 196x204DPI and placed into a CCITT/G4 encoded PDF. Only minor cosmetic changes were made. The original page US Letter page size was preserved, use the Acrobat "shrink to fit" print menu option for output to the DIN A4 format. Also remember to select double sided printing, if your can - there are 441 pages!

I make no representation whasoever as to the usefulness or exactitude of this file. (Imagine here whatever lawyerese you think is appropriate.) Please let me know your impressions on this file.

Have fun!

Alexandre Kampouris, Montréal E-Mail: ak@Radio-BIP.qc.ca

Preface

The Numerical Electromagnetics Code (NEC) has been developed at the Lawrence Livermore Laboratory, Livermore, California, under the sponsorship of the Naval Ocean Systems Center and the Air Force Weapons Laboratory. It is an advanced version of the Antenna Modeling Program (AMP) developed in the early 1970's by MBAssociates for the Naval Research Laboratory, Naval Ship Engineering Center, U.S. Army ECOM/Communications Systems, U.S. Army Strategic Communications Command, and Rome Air Development Center under Office of Naval Research Contract N00014-71-C-0187. The present version of NEC is the result of efforts by G. J. Burke and A. J. Poggio of Lawrence Livermore Laboratory.

The documentaton for NEC consists of three parts:

Part I: NEC Program Description - Theory

Part II: NEC Program Description - Code

Part III: NEC User's Guide

The documentation has been prepared by using the AMP documents as foundations and by modifying those as needed. In some cases this led to minor changes in the original documents while in many cases major modifications were required.

Over the years many individuals have been contributors to AMP and NEC and are acknowledged here as follows:

R. W. Adams	R. J. Lytle
J. N. Brittingham	E. K. Miller
G. J. Burke	J. B. Morton
F. J. Deadrick	G. M. Pjerrou
K. K. Hazard	A. J. Poggio
D. L. Knepp	E. S. Selden
D. L. Lager	

The support for the development of NEC-2 at the Lawrence Livermore Laboratory has been provided by the Naval Ocean Systems Center under MIPR-N0095376MP. Cognizant individuals under whom this project was carried out include: J. Rockway and J. Logan. Previous development of NEC also included the support of the Air Force Weapons Laboratory (Project Order 76-090) and was monitored by J. Castillo and TSgt. H. Goodwin.

Work was performed under the auspices of the U.S. Department of Energy under contract No. W-7405-Eng-48. Reference to a company or product name

does not imply approval or recommendation of the product by the University of California or the U.S. Department of Energy to the exclusion of others that may be suitable.

Contents

Section	Page

List of Illustrations

Abstract

The Numerical Electromagnetics Code (NEC-2) is a computer code for analyzing the electromagnetic response of an arbitrary structure consisting of wires and surfaces in free space or over a ground plane. The analysis is accomplished by the numerical solution of integral equations for induced currents. The excitation may be an incident plane wave or a voltage source on a wire, while the output may include current and charge density, electric or magnetic field in the vicinity of the structure, and radiated fields. Hence, the code may be used for antenna analysis or scattering and EMP studies.

This document is Part II of a three-part report. It contains a detailed description of the Fortran coding, including the definitions of variables and constants, and a listing of the code. The other two documents cover the equations and numerical methods (Part I) and instructions for use of the code (Part III).

KEY WORDS FOR DD FORM 1473:

EM scattering

EMP

Wire Model

Method of moments

Section I
Introduction

The Numerical Electromagnetics Code (NEC-2)* is a user-oriented computer code for the analysis of the electromagnetic response of antennas and other metal structures. It is built around the numerical solution of integral equations for the currents induced on the structure by sources or incident fields. This approach avoids many of the simplifying assumptions required by other solution methods and provides a highly accurate and versatile tool for electromagnetic analysis.

The code combines an integral equation for smooth surfaces with one specialized to wires to provide for convenient and accurate modeling of a wide range of structures. A model may include nonradiating networks and transmission lines connecting parts of the structure, perfect or imperfect conductors, and lumped-element loading. A structure may also be modeled over a ground plane that may be either a perfect or imperfect conductor.

The excitation may be either voltage sources on the structure or an incident plane wave of linear or elliptic polarization. The output may include induced currents and charges, near electric or magnetic fields, and radiated fields. Hence, the program is suited to either antenna analysis or scattering and EMP studies.

This document is Vol. II of a three-part report on NEC. It contains a detailed description of the Fortran coding. Section II contains for each routine: (1) a statement of purpose, (2) a narrative description of the methodology, (3) definitions of variables and constants, and (4) a listing of the code. The remaining sections cover the common blocks, system library functions, array dimension limitations, and subroutine linkage. The information in Vol. II will be of use mainly to persons attempting to modify the code or to use it on a computer system with which the delivered deck is not compatible.

Vol. I describes the equations and numerical methods used in NEC and Vol. III contains instructions for using the code, including preparation of input data and interpretation of output. Persons attempting to use NEC for the first time should start by reading Vol. III. Vol. I will help the new user to understand the capabilities and limitations of NEC.

*NEC-2 will be abbreviated to NEC elsewhere in this volume.

Section II
Code Description

In this section, each routine in NEC is described in detail. The main program is described first and is followed by the subroutines in alphabetical order. For each routine, there is a brief statement of its purpose, a description of the code, an alphabetized listing and definition of important variables and constants, and a listing of the code. Variables that are in common blocks, and hence occur in several routines, are usually omitted from the lists for individual routines. They are defined in Section III under their common block labels.

Following line MA 495 in the main program, all quantities of length have been normalized to wavelength. Current is normalized to wavelength throughout the solution. This changes the appearance of many of the equations. In particular the wave number, $k = 2\pi/\lambda$, usually appears as 2π.

MAIN

PURPOSE

To handle input and output and to call the appropriate subroutines.

METHOD

The structure of MAIN is shown in the flow charts of Figures 1 and 2 where Figure 1 represents the first half of the code to about line MA 459. Comment cards are read and printed after line MA 72 and subroutine DATAGN is called at MA 90 to read and process structure data. If a Numerical Green's Function (NGF) file was read in DATAGN then subroutine FBNFG is called to determine whether file storage is needed for the matrix and to allocate core storage. When a NGF has not been read the mode of matrix storage cannot be determined until line MA 464 since it depends on whether a NFG file is to be written.

The box labeled "Read data card" in Figure 1 refers to the READ statement at MA 139. Any of the types of data cards in Table 1 may be read at this point to set parameters or to request execution of the solution part of the code.

The integer variables IGØ and IFLOW are keys to the operation of the code. IGØ indicates the stage of completion of the solution as listed in Table 2. When a card requesting execution is read (NE, NH, RP, WG, or XQ) the solution part of the code (Figure 2) is entered at the point determined by IGØ (see MA 385, MA 420, MA 429, and MA 457). After the current has been computed IGØ is given the value five. If subsequent data cards change parameters, the value of IGØ is reduced to the value in Table 1 to indicate the point beyond which the solution must be repeated. For example, when an EX card is read IGØ is set equal to three if it was greater than three but is not changed if it was less than three. For cards that request execution "ex." is shown in Table 1.

IFLOW is used to indicate the type of the previous data card. When several cards of the same type can be used together (CP, LD, NT, TL, and EX for voltage sources) a counter is incremented and data is added to arrays if the card is the same as the previous card as indicated by IFLOW. If the previous card was different the counter is initialized and previous data in the arrays is destroyed. IFLOW is also used to indicate what type of card

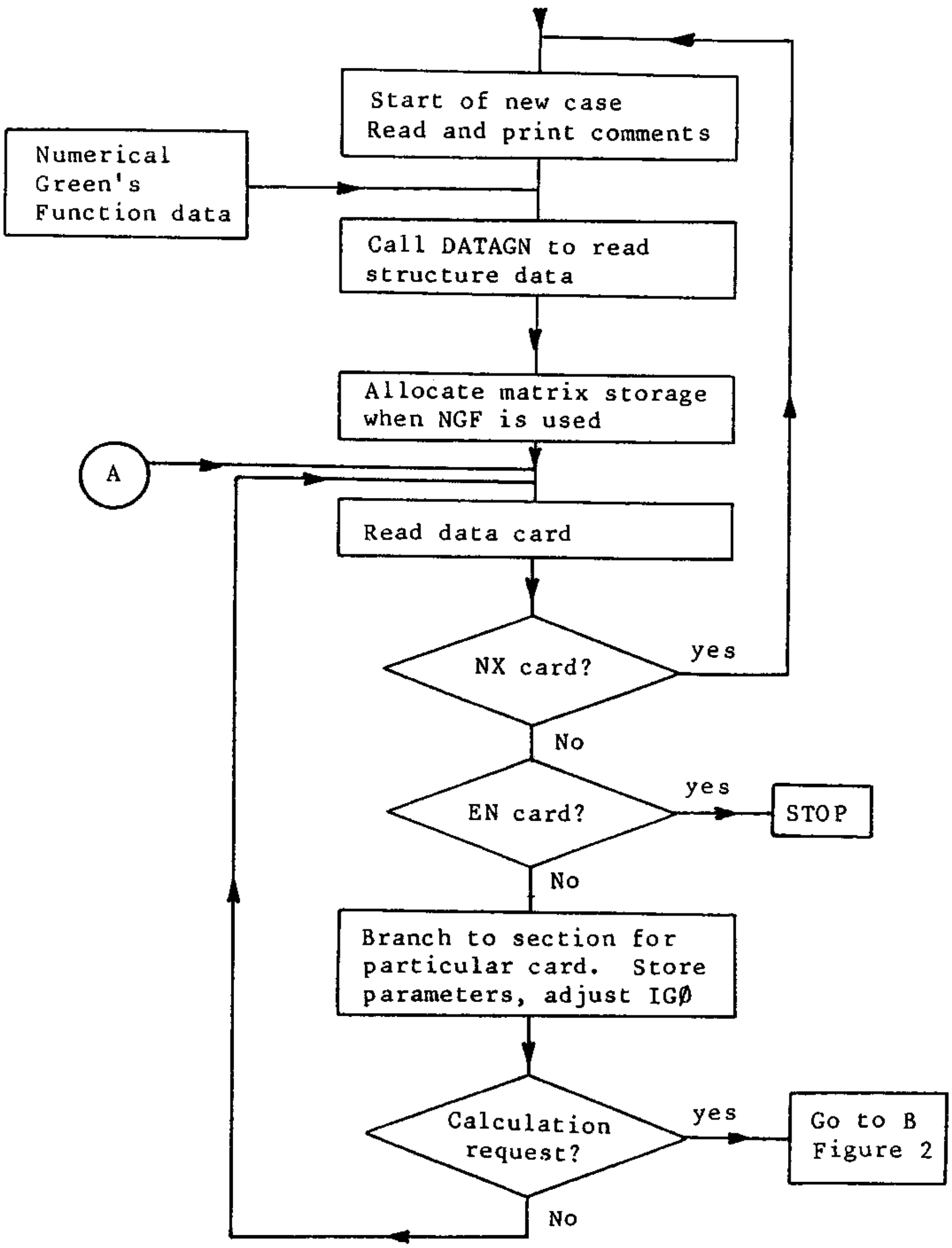


Figure 1. Flow Diagram of Main Program Input Section

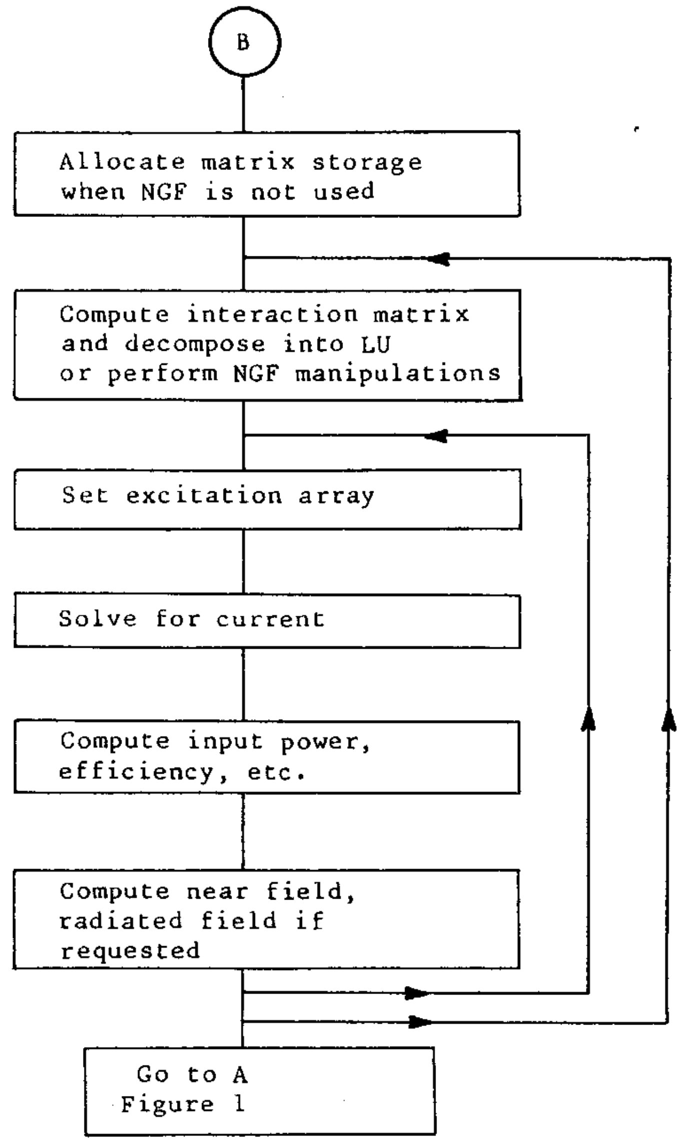


Figure 2. Flow Diagram of Main Program Computation Section

TABLE 1

	I	AIN(I)	GO TO	Line	IGØ	IFLOW
1	21	CP	304	202	-	2
2	19	EK	320	194	2	1
3	13	EN	STØP	166	-	-
4	5	EX	24	275	3	5
5	2	FR	16	172	1	1
6	9	GD	34	389	-	9
7	4	GN	21	245	2	4
8	16	KH	305	187	2	1
9	3	LD	17	221	2	3
10	8	NE	32	370	ex.*	8
11	17	NH	208	368	ex.*	8
12	6	NT	28	321	3	6
13	12	NX	1	69	1	1
14	18	PQ	319	358	-	-
15	15	PT	31	348	-	-
16	10	RP	36	398	ex.	10
17	14	TL	28	321	3	6
18	20	WG	322	424	ex.	12
19	7	XQ	37	433	ex.	7 or 11

* NE and NH do not cause execution when multiple frequencies have been requested on the FR card. This allows computation of both near fields and radiated fields in a frequency loop.

Table 2

IGO	Completion Point
1	Start
2	Frequency has been set and geometry scaled to wavelength
3	Interaction matrix filled and factored
4,5	Current computed and printed

requested the solution (NE, RP, etc.). Cards such as RP may be stacked together but are not stored since they are acted upon as they are encountered.

The solution part of the code contains a loop over frequency starting at MA 463 and a loop over incident field direction starting at MA 562. FBLOCK is called at MA 465 to determine whether file storage is required for the matrix. From MA 466 to MA 493 the structure data are scaled from units of meters to wavelength or from one wavelength to the next when frequency is changed. Subroutine LOAD is called at MA 497 to fill array ZARRAY for the given frequency. At MA 520 the Sommerfeld interpolation tables are read from file TAPE21 if this option is used. NXA(1) is set to zero at MA 67 so the test ensures that the tape is read only once.

When the NGF option is not in use the matrix is filled by subroutine CMSET at MA 537 and factored by subroutine FACTRS at MA 540. When the NGF is used the equivalent steps are performed by CMNGF and FACGF. If a NGF file is to be written, subroutine GFOUT is called at MA 557 to write TAPE20.

Subroutine ETMNS, called at MA 582, fills the excitation array and the current is computed in subroutine NETWK called at MA 611. If transmission lines or two port networks are used NETWK combines the network equations with driving-point interaction equations derived from the primary interaction matrix. Otherwise the current is computed directly from the primary matrix.

The remainder of MAIN prints the currents and calls subroutines for near fields, radiated fields or coupling.

SYMBOL DICTIONARY:

AIN	= mnemonic from data card
ATST	= array of possible data card mnemonics
CMAG	= magnitude of the current in amperes
COM	= array to store text from comment cards
CURI	= current on segment I in amperes
CVEL	= (velocity of light) (10^{-6}) in meters/second
DELFRQ	= frequency increment (additive or multiplicative)
DPH	= far-field φ angle increment in degrees (input quantity)
DTH	= far-field θ angle increment in degrees (input quantity)

DXNR, DYNR, DZNR	= near-field observation point increments (input quantities with multiple meanings -- see NE card)
EPH	= current component in direction $\hat{t}_2$ on patch
EPHA	= phase angle of EPH
EPHM	= magnitude of EPH
EPSC	= complex dielectric constant of ground $\epsilon_c = \epsilon_r - j\sigma/\omega\epsilon_0$.)
EPSCF	= ϵ_c read from file TAPE21
EPSR	= ϵ_r
EPSR2	= ϵ_r for outer ground region
ETH	= current component in direction $\hat{t}_1$ on patch
ETHA	= phase angle of ETH
ETHM	= magnitude of ETH
EX	= $\hat{x}$ component of current on a patch
EXTIM	= time at start of run (seconds)
EY	= $\hat{y}$ component of current on a patch
EZ	= $\hat{z}$ component of current on a patch
FJ	= $\sqrt{-1}$
FMHZ	= frequency in MHz
FMHZS	= frequency in MHz
FNORM	= multiply used array; stores impedances for printing of the normalized impedance or stores currents in the receiving pattern case for printing normalized receiving pattern
FR	= (next frequency)/(present frequency)
FR2	= (FR)(FR)
GNOR	= if non-zero, equals gain normalization factor (dB) from RP card
HPOL	= array containing polarization types (Hollerith)
IAVP	= input integer flag used in average gain logic (RP card)
IAX	= input integer flag specifying gain type (RP card)
IB11	= location in array CM for start of storage of submatrix B when NGF is used
IC11	= location in array CM for start of storage of submatrix C when NGF is used

ID11	=	location in CM for submatrix D
IEXK	=	flag to select the extended thin-wire kernel
IFAR	=	input integer flag specifying type of field calculation and type of ground system in far field (RP card)
IFLOW	=	integer flag used to distinguish various input sections
IFRQ	=	input integer flag specifying type of frequency stepping (FR card)
IGO	=	integer to indicate stage of completion of the solution
INC	=	incident field loop index
INOR	=	input integer flag used for normalized gain request (RP card)
IPD	=	input integer flag selects gain type for normalization (RP card)
IPED	=	input integer flag used for impedance normalization request (EX card)
IPTAG	=	input integer for print control equal to segment tag number (PT card)
IPTAGF	=	input integer for print control specifying segment placement in a set of equal tags (PT card)
IPTAGT	=	same function as IPTAGF (input, PT card)
IPTFLG	=	input integer flag specifying type of print control (PT card)
IPTAQ IPTAQF IPTAQT IPTFLQ	=	same as above four variables but for PQ card
IRESRV	=	length of array CM in complex numbers
IRNGF	=	storage in array CM that is reserved for later use when a NGF file is written
ISANT	=	array of segment numbers for voltage sources
ISAVE	=	segment number for normalized receiving pattern calculation

ISEG1 (I) ISEG2 (I)	=	segment numbers of end 1 and end 2 of the i^{th} network connection
ITMP1 to ITMP5	=	temporary storage
IX	=	array for matrix pivot element information
IX11	=	location in CM of the start of an array in the NGF solution
IXTYP	=	excitation type from EX card
KCOM	=	number of comment cards read
LDTAG	=	tag number of loaded segment
LDTAGF	=	number of first loaded segment in set of segments having given tag
LDTAGT	=	last loaded segment
LDTYP	=	loading type
LOADMX	=	maximum number of loading cards
MASYM	=	flag to request matrix asymmetry calculation
MHZ	=	frequency loop index
MPCNT	=	counter for data cards
NCOUP	=	number of excitation points for coupling calculation
NCSEG NCTAG	=	excitation segment for coupling calculation
NEAR	=	increment option for near field points
NEQ	=	order of the primary interaction matrix
NEQ2	=	number of new unknowns in NGF mode
NETMX	=	maximum number of network data cards
NFEH	=	0 for near E field, 1 for near H
NFRQ	=	number of frequency steps
NONET	=	number of network data cards
NORMF	=	dimension of FNORM
NPHI	=	number of phi steps in incident field
NPHIC	=	loop index for phi in incident field
NPRINT	=	print control flag for subroutine NETWK
NRX NRY NRZ	=	number of steps in near field evaluation loops
NSANT	=	number of voltage sources
NSMAX	=	maximum number of voltage sources

MAIN

NTHI	=	number of theta steps in incident field
NTHIC	=	loop index for theta in incident field
PH	=	phase angle of current or charge (degrees)
PHISS	=	initial ϕ value for incident field
PIN	=	P_{in} = total power supplied to a structure by all voltage sources (Σ Re(VI*)/2). For a Hertzian dipole source $P_{in} = \eta(\pi/3)\|I\ell/\lambda\|^2$.
PLOSS	=	power lost in distributed and point structure loads in watts
PNET	=	array contains Hollerith transmission line type
RFLD	=	if non-zero, equal to input far-field observation distance in meters
RKH	=	minimum separation for use of approximate interaction equations
SCRWLT	=	input length of radials in radial wire screen (GN card) in meters
SCRWRT	=	radius of wires in radial wire ground screen in meters
SIG	=	conductivity of ground (σ in mhos/meter on GN card)
SIG2	=	conductivity of second medium in mhos/meter (GN and GD card)
TA	=	$\pi/180$
THETIS	=	initial θ for incident field
THETS	=	initial θ for radiated field
TIM	=	matrix computation time (seconds)
TMP1 to TMP6	=	temporary input variables
XPR1 to XPR6	=	input quantities for incident field or Hertzian dipole illumination
ZLC, ZLI, ZLR	=	input quantities for loading
ZPNORM	=	impedance normalization quantity

CONSTANTS

1.E-20	=	used as small value test

1.745329252	=	$\pi/180$
2367.067	=	$2\pi\eta_0$
59.96	=	$1/(2\pi c\varepsilon_0)$
299.8	=	$c/10^6$

```
      PROGRAM NEC(INPUT,TAPE5=INPUT,OUTPUT,TAPE11,TAPE12,TAPE13,TAPE14,  MA   1
     1TAPE15,TAPE16,TAPE20,TAPE21)                                       MA   2
C                                                                        MA   3
C     NUMERICAL ELECTROMAGNETICS CODE (NEC2)  DEVELOPED AT LAWRENCE      MA   4
C     LIVERMORE LAB., LIVERMORE, CA.  (CONTACT G. BURKE, 415-422-8414)   MA   5
C     FILE CREATED 4/11/80.                                              MA   6
C                                                                        MA   7
C                 ***********NOTICE**********                            MA   8
C     THIS COMPUTER CODE MATERIAL WAS PREPARED AS AN ACCOUNT OF WORK     MA   9
C     SPONSORED BY THE UNITED STATES GOVERNMENT.  NEITHER THE UNITED     MA  10
C     STATES NOR THE UNITED STATES DEPARTMENT OF ENERGY, NOR ANY OF      MA  11
C     THEIR EMPLOYEES, NOR ANY OF THEIR CONTRACTORS, SUBCONTRACTORS, OR  MA  12
C     THEIR EMPLOYEES, MAKES ANY WARRANTY, EXPRESS OR IMPLIED, OR        MA  13
C     ASSUMES ANY LEGAL LIABILITY OR RESPONSIBILITY FOR THE ACCURACY,    MA  14
C     COMPLETENESS OR USEFULNESS OF ANY INFORMATION, APPARATUS, PRODUCT  MA  15
C     OR PROCESS DISCLOSED, OR REPRESENTS THAT ITS USE WOULD NOT         MA  16
C     INFRINGE PRIVATELY-OWNED RIGHTS.                                   MA  17
C                                                                        MA  18
      INTEGER  AIN,ATST,PNET,HPOL                                        MA  19
      COMPLEX  CM,FJ,VSANT,ETH,EPH,ZRATI,CUR,CURI,ZARRAY,ZRATI2          MA  20
      COMPLEX  EX,EY,EZ,ZPED,VQD,VQDS,T1,Y11A,Y12A,EPSC,U,U2,XX1,XX2     MA  21
      COMPLEX  AR1,AR2,AR3,EPSCF,FRATI                                   MA  22
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),       MA  23
     1SI(300),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),           MA  24
     2ITAG(300),ICONX(300),WLAM,IPSYM                                    MA  25
      COMMON /CMB/CM(4000)                                               MA  26
      COMMON /MATPAR/ ICASE,NBLOKS,NPBLK,NLAST,NBLSYM,NPSYM,NLSYM,IMAT,  MA  27
     1ICASX,NBBX,NPBX,NLBX,NBBL,NPBL,NLBL                                MA  28
      COMMON/SAVE/IP(600),KCOM,COM(13,5),EPSR,SIG,SCRWLT,SCRWRT,FMHZ     MA  29
      COMMON /CRNT/ AIR(300),AII(300),BIR(300),BII(300),CIR(300),        MA  30
     1 CII(300),CUR(900)                                                 MA  31
      COMMON /GND/ZRATI,ZRATI2,FRATI,CL,CH,SCRWL,SCRWR,NRADL,KSYMP,IFAR, MA  32
     1IPERF,T1,T2                                                        MA  33
      COMMON /ZLOAD/ ZARRAY(300),NLOAD,NLODF                             MA  34
      COMMON/YPARM/NCOUP,ICOUP,NCTAG(5),NCSEG(5),Y11A(5),Y12A(20)        MA  35
      COMMON /SEGJ/ AX(30),BX(30),CX(30),JCO(30),JSNO,ISCON(50),NSCON,   MA  36
     1IPCON(10),NPCON                                                    MA  37
      COMMON/VSORC/VQD(30),VSANT(30),VQDS(30),IVQD(30),ISANT(30),        MA  38
     1IQDS(30),NVQD,NSANT,NQDS                                           MA  39
      COMMON/NETCX/ZPED,PIN,PNLS,NEQ,NPEQ,NEQ2,NONET,NTSOL,NPRINT,       MA  40
     1MASYM,ISEG1(30),ISEG2(30),X11R(30),X11I(30),X12R(30),X12I(30),     MA  41
     1X22R(30),X22I(30),NTYP(30)                                         MA  42
      COMMON/FPAT/NTH,NPH,IPD,IAVP,INOR,IAX,THETS,PHIS,DTH,DPH,          MA  43
     1RFLD,GNOR,CLT,CHT,EPSR2,SIG2,IXTYP,XPR6,PINR,PNLR,PLOSS,           MA  44
     1NEAR,NFEH,NRX,NRY,NRZ,XNR,YNR,ZNR,DXNR,DYNR,DZNR                   MA  45
      COMMON /GGRID/ AR1(11,10,4),AR2(17,5,4),AR3(9,8,4),EPSCF,DXA(3),   MA  46
     1DYA(3),XSA(3),YSA(3),NXA(3),NYA(3)                                 MA  47
      COMMON/GWAV/U,U2,XX1,XX2,R1,R2,ZMH,ZPH                             MA  48
      DIMENSION CAB(1),SAB(1),X2(1),Y2(1),Z2(1)                          MA  49
      DIMENSION LDTYP(30),LDTAG(30),LDTAGF(30),LDTAGT(30),ZLR(30),       MA  50
     1ZLI(30),ZLC(30)                                                    MA  51
      DIMENSION ATST(21),PNET(6),HPOL(3),IX(600)                         MA  52
      DIMENSION FNORM(200)                                               MA  53
      DIMENSION T1X(1),T1Y(1),T1Z(1),T2X(1),T2Y(1),T2Z(1)                MA  54
      EQUIVALENCE (CAB,ALP),(SAB,BET),(X2,SI),(Y2,ALP),(Z2,BET)          MA  55
      EQUIVALENCE (T1X,SI),(T1Y,ALP),(T1Z,BET),(T2X,ICON1),(T2Y,ICON2),  MA  56
     1 (T2Z,ITAG)                                                        MA  57
      DATA ATST/2HCE,2HFR,2HLD,2HGN,2HEX,2HNT,2HXQ,2HNE,2HGD,2HRP,2HCM,  MA  58
     1 2HNX,2HEN,2HTL,2HPT,2HKH,2HNH,2HPQ,2HEK,2HWG,2HCP/                MA  59
      DATA HPOL/6HLINEAR,5HRIGHT,4HLEFT/                                 MA  60
      DATA PNET/6H      ,2H  ,6HSTRAIG,2HHT,6HCROSSE,1HD/                MA  61
      DATA TA/1.745329252E-02/,CVEL/299.8/                               MA  62
      DATA LOADMX,NSMAX,NETMX/30,30,30/,NORMF/200/                       MA  63
```

```
      CALL SECOND(EXTIM)
      FJ=(0.,1.)
      LD=300
      NXA(1)=0
      IRESRV=4000
1     KCOM=0
2     KCOM=KCOM+1
      IF (KCOM.GT.5) KCOM=5
      READ(5,125)AIN,(COM(I,KCOM),I=1,13)
      IF(KCOM.GT.1)GO TO 3
      PRINT 126
      PRINT 127
      PRINT 128
3     PRINT 129, (COM(I,KCOM),I=1,13)
      IF (AIN.EQ.ATST(11)) GO TO 2
      IF (AIN.EQ.ATST(1)) GO TO 4
      PRINT 130
      STOP
4     CONTINUE
      DO 5 I=1,LD
5     ZARRAY(I)=(0.,0.)
      MPCNT=0
      IMAT=0
C
C     SET UP GEOMETRY DATA IN SUBROUTINE DATAGN
C
      CALL  DATAGN
      IFLOW=1
      IF(IMAT.EQ.0)GO TO 326
C
C     CORE ALLOCATION FOR ARRAYS B, C, AND D FOR N.G.F. SOLUTION
C
      NEQ=N1+2*M1
      NEQ2=N-N1+2*(M-M1)+NSCON+2*NPCON
      CALL FBNGF(NEQ,NEQ2,IRESRV,IB11,IC11,ID11,IX11)
      GO TO 6
326   NEQ=N+2*M
      NEQ2=0
      IB11=1
      IC11=1
      ID11=1
      IX11=1
      ICASX=0
6     NPEQ=NP+2*MP
      PRINT 135
C
C     DEFAULT VALUES FOR INPUT PARAMETERS AND FLAGS
C
      IGO=1
      FMHZS=CVEL
      NFRQ=1
      RKH=1.
      IEXK=0
      IXTYP=0
      NLOAD=0
      NONET=0
      NEAR=-1
      IPTFLG=-2
      IPTFLQ=-1
      IFAR=-1
      ZRATI=(1.,0.)
      IPED=0
      IRNGF=0
      NCOUP=0
```

```
      ICOUP=0
      IF(ICASX.GT.0)GO TO 14
      FMHZ=CVEL
      NLODF=0
      KSYMP=1
      NRADL=0
      IPERF=0
C
C     MAIN INPUT SECTION - STANDARD READ STATEMENT - JUMPS TO APPRO-
C     PRIATE SECTION FOR SPECIFIC PARAMETER SET UP
C
14    READ(5,136)AIN,ITMP1,ITMP2,ITMP3,ITMP4,TMP1,TMP2,TMP3,TMP4,TMP5,
     1TMP6
      MPCNT=MPCNT+1
      PRINT 137, MPCNT,AIN,ITMP1,ITMP2,ITMP3,ITMP4,TMP1,TMP2,TMP3,TMP4,
     1TMP5,TMP6
      IF (AIN.EQ.ATST(2)) GO TO 16
      IF (AIN.EQ.ATST(3)) GO TO 17
      IF (AIN.EQ.ATST(4)) GO TO 21
      IF (AIN.EQ.ATST(5)) GO TO 24
      IF (AIN.EQ.ATST(6)) GO TO 28
      IF (AIN.EQ.ATST(14)) GO TO 28
      IF (AIN.EQ.ATST(15)) GO TO 31
      IF (AIN.EQ.ATST(18)) GO TO 319
      IF (AIN.EQ.ATST(7)) GO TO 37
      IF (AIN.EQ.ATST(8)) GO TO 32
      IF (AIN.EQ.ATST(17)) GO TO 208
      IF (AIN.EQ.ATST(9)) GO TO 34
      IF (AIN.EQ.ATST(10)) GO TO 36
      IF (AIN.EQ.ATST(16)) GO TO 305
      IF (AIN.EQ.ATST(19)) GO TO 320
      IF (AIN.EQ.ATST(12)) GO TO 1
      IF (AIN.EQ.ATST(20)) GO TO 322
      IF (AIN.EQ.ATST(21)) GO TO 304
      IF (AIN.NE.ATST(13)) GO TO 15
      CALL SECOND(TMP1)
      TMP1=TMP1-EXTIM
      PRINT 201,TMP1
      STOP
15    PRINT 138
      STOP
C
C     FREQUENCY PARAMETERS
C
16    IFRQ=ITMP1
      IF(ICASX.EQ.0)GO TO 8
      PRINT 303,AIN
      STOP
8     NFRQ=ITMP2
      IF (NFRQ.EQ.0) NFRQ=1
      FMHZ=TMP1
      DELFRQ=TMP2
      IF(IPED.EQ.1)ZPNORM=0.
      IGO=1
      IFLOW=1
      GO TO 14
C
C     MATRIX INTEGRATION LIMIT
C
305   RKH=TMP1
      IF(IGO.GT.2)IGO=2
      IFLOW=1
      GO TO 14
C
```

```
C     EXTENDED THIN WIRE KERNEL OPTION
C
  320 IEXK=1
      IF(ITMP1.EQ.-1)IEXK=0
      IF(IGO.GT.2)IGO=2
      IFLOW=1
      GO TO 14
C
C     MAXIMUM COUPLING BETWEEN ANTENNAS
C
  304 IF(IFLOW.NE.2)NCOUP=0
      ICOUP=0
      IFLOW=2
      IF(ITMP2.EQ.0)GO TO 14
      NCOUP=NCOUP+1
      IF(NCOUP.GT.5)GO TO 312
      NCTAG(NCOUP)=ITMP1
      NCSEG(NCOUP)=ITMP2
      IF(ITMP4.EQ.0)GO TO 14
      NCOUP=NCOUP+1
      IF(NCOUP.GT.5)GO TO 312
      NCTAG(NCOUP)=ITMP3
      NCSEG(NCOUP)=ITMP4
      GO TO 14
  312 PRINT 313
      STOP
C
C     LOADING PARAMETERS
C
   17 IF (IFLOW.EQ.3) GO TO 18
      NLOAD=0
      IFLOW=3
      IF (IGO.GT.2) IGO=2
      IF (ITMP1.EQ.(-1)) GO TO 14
   18 NLOAD=NLOAD+1
      IF (NLOAD.LE.LOADMX) GO TO 19
      PRINT 139
      STOP
   19 LDTYP(NLOAD)=ITMP1
      LDTAG(NLOAD)=ITMP2
      IF (ITMP4.EQ.0) ITMP4=ITMP3
      LDTAGF(NLOAD)=ITMP3
      LDTAGT(NLOAD)=ITMP4
      IF (ITMP4.GE.ITMP3) GO TO 20
      PRINT 140, NLOAD,ITMP3,ITMP4
      STOP
   20 ZLR(NLOAD)=TMP1
      ZLI(NLOAD)=TMP2
      ZLC(NLOAD)=TMP3
      GO TO 14
C
C     GROUND PARAMETERS UNDER THE ANTENNA
C
   21 IFLOW=4
      IF(ICASX.EQ.0)GO TO 10
      PRINT 303,AIN
      STOP
   10 IF (IGO.GT.2) IGO=2
      IF (ITMP1.NE.(-1)) GO TO 22
      KSYMP=1
      NRADL=0
      IPERF=0
      GO TO 14
   22 IPERF=ITMP1
```

```fortran
      NRADL=ITMP2
      KSYMP=2
      EPSR=TMP1
      SIG=TMP2
      IF (NRADL.EQ.0) GO TO 23
      IF(IPERF.NE.2)GO TO 314
      PRINT 390
      STOP
314   SCRWLT=TMP3
      SCRWRT=TMP4
      GO TO 14
23    EPSR2=TMP3
      SIG2=TMP4
      CLT=TMP5
      CHT=TMP6
      GO TO 14
C
C     EXCITATION PARAMETERS
C
24    IF (IFLOW.EQ.5) GO TO 25
      NSANT=0
      NVQD=0
      IPED=0
      IFLOW=5
      IF (IGO.GT.3) IGO=3
25    MASYM=ITMP4/10
      IF (ITMP1.GT.0.AND.ITMP1.NE.5) GO TO 27
      IXTYP=ITMP1
      NTSOL=0
      IF(IXTYP.EQ.0)GO TO 205
      NVQD=NVQD+1
      IF(NVQD.GT.NSMAX)GO TO 206
      IVQD(NVQD)=ISEGNO(ITMP2,ITMP3)
      VQD(NVQD)=CMPLX(TMP1,TMP2)
      IF(CABS(VQD(NVQD)).LT.1.E-20)VQD(NVQD)=(1.,0.)
      GO TO 207
205   NSANT=NSANT+1
      IF (NSANT.LE.NSMAX) GO TO 26
206   PRINT 141
      STOP
26    ISANT(NSANT)=ISEGNO(ITMP2,ITMP3)
      VSANT(NSANT)=CMPLX(TMP1,TMP2)
      IF (CABS(VSANT(NSANT)).LT.1.E-20) VSANT(NSANT)=(1.,0.)
207   IPED=ITMP4-MASYM*10
      ZPNORM=TMP3
      IF (IPED.EQ.1.AND.ZPNORM.GT.0) IPED=2
      GO TO 14
27    IF (IXTYP.EQ.0.OR.IXTYP.EQ.5) NTSOL=0
      IXTYP=ITMP1
      NTHI=ITMP2
      NPHI=ITMP3
      XPR1=TMP1
      XPR2=TMP2
      XPR3=TMP3
      XPR4=TMP4
      XPR5=TMP5
      XPR6=TMP6
      NSANT=0
      NVQD=0
      THETIS=XPR1
      PHISS=XPR2
      GO TO 14
C
C     NETWORK PARAMETERS
```

```
C                                                              MA 320
 28     IF (IFLOW.EQ.6) GO TO 29                               MA 321
        NONET=0                                                MA 322
        NTSOL=0                                                MA 323
        IFLOW=6                                                MA 324
        IF (IGO.GT.3) IGO=3                                    MA 325
        IF (ITMP2.EQ.(-1)) GO TO 14                            MA 326
 29     NONET=NONET+1                                          MA 327
        IF (NONET.LE.NETMX) GO TO 30                           MA 328
        PRINT 142                                              MA 329
        STOP                                                   MA 330
 30     NTYP(NONET)=2                                          MA 331
        IF (AIN.EQ.ATST(6)) NTYP(NONET)=1                      MA 332
        ISEG1(NONET)=ISEGNO(ITMP1,ITMP2)                       MA 333
        ISEG2(NONET)=ISEGNO(ITMP3,ITMP4)                       MA 334
        X11R(NONET)=TMP1                                       MA 335
        X11I(NONET)=TMP2                                       MA 336
        X12R(NONET)=TMP3                                       MA 337
        X12I(NONET)=TMP4                                       MA 338
        X22R(NONET)=TMP5                                       MA 339
        X22I(NONET)=TMP6                                       MA 340
        IF (NTYP(NONET).EQ.1.OR.TMP1.GT.0.) GO TO 14           MA 341
        NTYP(NONET)=3                                          MA 342
        X11R(NONET)=-TMP1                                      MA 343
        GO TO 14                                               MA 344
C                                                              MA 345
C       PRINT CONTROL FOR CURRENT                              MA 346
C                                                              MA 347
 31     IPTFLG=ITMP1                                           MA 348
        IPTAG=ITMP2                                            MA 349
        IPTAGF=ITMP3                                           MA 350
        IPTAGT=ITMP4                                           MA 351
        IF(ITMP3.EQ.0.AND.IPTFLG.NE.-1)IPTFLG=-2               MA 352
        IF (ITMP4.EQ.0) IPTAGT=IPTAGF                          MA 353
        GO TO 14                                               MA 354
C                                                              MA 355
C       PRINT CONTROL FOR CHARGE                               MA 356
C                                                              MA 357
 319    IPTFLQ=ITMP1                                           MA 358
        IPTAQ=ITMP2                                            MA 359
        IPTAQF=ITMP3                                           MA 360
        IPTAQT=ITMP4                                           MA 361
        IF(ITMP3.EQ.0.AND.IPTFLQ.NE.-1)IPTFLQ=-2               MA 362
        IF(ITMP4.EQ.0)IPTAQT=IPTAQF                            MA 363
        GO TO 14                                               MA 364
C                                                              MA 365
C       NEAR FIELD CALCULATION PARAMETERS                      MA 366
C                                                              MA 367
 208    NFEH=1                                                 MA 368
        GO TO 209                                              MA 369
 32     NFEH=0                                                 MA 370
 209    IF (.NOT.(IFLOW.EQ.8.AND.NFRQ.NE.1)) GO TO 33          MA 371
        PRINT 143                                              MA 372
 33     NEAR=ITMP1                                             MA 373
        NRX=ITMP2                                              MA 374
        NRY=ITMP3                                              MA 375
        NRZ=ITMP4                                              MA 376
        XNR=TMP1                                               MA 377
        YNR=TMP2                                               MA 378
        ZNR=TMP3                                               MA 379
        DXNR=TMP4                                              MA 380
        DYNR=TMP5                                              MA 381
        DZNR=TMP6                                              MA 382
        IFLOW=8                                                MA 383
```

```
      IF (NFRQ.NE.1) GO TO 14                                           MA 384
      GO TO (41,46,53,71,72), IGO                                       MA 385
C                                                                       MA 386
C     GROUND REPRESENTATION                                             MA 387
C                                                                       MA 388
34    EPSR2=TMP1                                                        MA 389
      SIG2=TMP2                                                         MA 390
      CLT=TMP3                                                          MA 391
      CHT=TMP4                                                          MA 392
      IFLOW=9                                                           MA 393
      GO TO 14                                                          MA 394
C                                                                       MA 395
C     STANDARD OBSERVATION ANGLE PARAMETERS                             MA 396
C                                                                       MA 397
36    IFAR=ITMP1                                                        MA 398
      NTH=ITMP2                                                         MA 399
      NPH=ITMP3                                                         MA 400
      IF (NTH.EQ.0) NTH=1                                               MA 401
      IF (NPH.EQ.0) NPH=1                                               MA 402
      IPD=ITMP4/10                                                      MA 403
      IAVP=ITMP4-IPD*10                                                 MA 404
      INOR=IPD/10                                                       MA 405
      IPD=IPD-INOR*10                                                   MA 406
      IAX=INOR/10                                                       MA 407
      INOR=INOR-IAX*10                                                  MA 408
      IF (IAX.NE.0) IAX=1                                               MA 409
      IF (IPD.NE.0) IPD=1                                               MA 410
      IF (NTH.LT.2.OR.NPH.LT.2) IAVP=0                                  MA 411
      IF (IFAR.EQ.1) IAVP=0                                             MA 412
      THETS=TMP1                                                        MA 413
      PHIS=TMP2                                                         MA 414
      DTH=TMP3                                                          MA 415
      DPH=TMP4                                                          MA 416
      RFLD=TMP5                                                         MA 417
      GNOR=TMP6                                                         MA 418
      IFLOW=10                                                          MA 419
      GO TO (41,46,53,71,78), IGO                                       MA 420
C                                                                       MA 421
C     WRITE NUMERICAL GREEN'S FUNCTION TAPE                             MA 422
C                                                                       MA 423
322   IFLOW=12                                                          MA 424
      IF(ICASX.EQ.0)GO TO 301                                           MA 425
      PRINT 302                                                         MA 426
      STOP                                                              MA 427
301   IRNGF=IRESRV/2                                                    MA 428
      GO TO (41,46,52,52,52),IGO                                        MA 429
C                                                                       MA 430
C     EXECUTE CARD  -  CALC. INCLUDING RADIATED FIELDS                  MA 431
C                                                                       MA 432
37    IF (IFLOW.EQ.10.AND.ITMP1.EQ.0) GO TO 14                          MA 433
      IF (NFRQ.EQ.1.AND.ITMP1.EQ.0.AND.IFLOW.GT.7) GO TO 14             MA 434
      IF (ITMP1.NE.0) GO TO 39                                          MA 435
      IF (IFLOW.GT.7) GO TO 38                                          MA 436
      IFLOW=7                                                           MA 437
      GO TO 40                                                          MA 438
38    IFLOW=11                                                          MA 439
      GO TO 40                                                          MA 440
39    IFAR=0                                                            MA 441
      RFLD=0                                                            MA 442
      IPD=0                                                             MA 443
      IAVP=0                                                            MA 444
      INOR=0                                                            MA 445
      IAX=0                                                             MA 446
      NTH=91                                                            MA 447
```

```
      NPH=1
      THETS=0.
      PHIS=0.
      DTH=1.0
      DPH=0.
      IF (ITMP1.EQ.2) PHIS=90.
      IF (ITMP1.NE.3) GO TO 40
      NPH=2
      DPH=90.
   40 GO TO (41,46,53,71,78), IGO
C
C     END OF THE MAIN INPUT SECTION
C
C     BEGINNING OF THE FREQUENCY DO LOOP
C
   41 MHZ=1
C     CORE ALLOCATION FOR PRIMARY INTERACTON MATRIX.  (A)
      IF(IMAT.EQ.0)CALL FBLOCK(NPEQ,NEQ,IRESRV,IRNGF,IPSYM)
   42 IF (MHZ.EQ.1) GO TO 44
      IF (IFRQ.EQ.1) GO TO 43
      FMHZ=FMHZ+DELFRQ
      GO TO 44
   43 FMHZ=FMHZ*DELFRQ
   44 FR=FMHZ/FMHZS
      WLAM=CVEL/FMHZ
      PRINT 145, FMHZ,WLAM
      PRINT 196,RKH
      IF(IEXK.EQ.1)PRINT 321
C     FREQUENCY SCALING OF GEOMETRIC PARAMETERS
      FMHZS=FMHZ
      IF(N.EQ.0)GO TO 306
      DO 45 I=1,N
      X(I)=X(I)*FR
      Y(I)=Y(I)*FR
      Z(I)=Z(I)*FR
      SI(I)=SI(I)*FR
   45 BI(I)=BI(I)*FR
  306 IF(M.EQ.0)GO TO 307
      FR2=FR*FR
      J=LD+1
      DO 245 I=1,M
      J=J-1
      X(J)=X(J)*FR
      Y(J)=Y(J)*FR
      Z(J)=Z(J)*FR
  245 BI(J)=BI(J)*FR2
  307 IGO=2
C     STRUCTURE SEGMENT LOADING
   46 PRINT 146
      IF(NLOAD.NE.0) CALL LOAD(LDTYP,LDTAG,LDTAGF,LDTAGT,ZLR,ZLI,ZLC)
      IF(NLOAD.EQ.0.AND.NLODF.EQ.0)PRINT 147
      IF(NLOAD.EQ.0.AND.NLODF.NE.0)PRINT 327
C     GROUND PARAMETER
      PRINT 148
      IF (KSYMP.EQ.1) GO TO 49
      FRATI=(1.,0.)
      IF (IPERF.EQ.1) GO TO 48
      IF(SIG.LT.0.)SIG=-SIG/(59.96*WLAM)
      EPSC=CMPLX(EPSR,-SIG*WLAM*59.96)
      ZRATI=1./CSQRT(EPSC)
      U=ZRATI
      U2=U*U
      IF (NRADL.EQ.0) GO TO 47
      SCRWL=SCRWLT/WLAM
```

```
      SCRWR=SCRWRT/WLAM
      T1=FJ*2367.067/FLOAT(NRADL)
      T2=SCRWR*FLOAT(NRADL)
      PRINT 170, NRADL,SCRWLT,SCRWRT
      PRINT 149
   47 IF(IPERF.EQ.2)GO TO 328
      PRINT 391
      GO TO 329
  328 IF(NXA(1).EQ.0)READ(21)AR1,AR2,AR3,EPSCF,DXA,DYA,XSA,YSA,NXA,NYA
      FRATI=(EPSC-1.)/(EPSC+1.)
      IF(CABS((EPSCF-EPSC)/EPSC).LT.1.E-3)GO TO 330
      PRINT 393,EPSCF,EPSC
      STOP
  330 PRINT 392
  329 PRINT 150, EPSR,SIG,EPSC
      GO TO 50
   48 PRINT 151
      GO TO 50
   49 PRINT 152
   50 CONTINUE
C * * * *
C     FILL AND FACTOR PRIMARY INTERACTION MATRIX
C
      CALL SECOND (TIM1)
      IF(ICASX.NE.0)GO TO 324
      CALL CMSET(NEQ,CM,RKH,IEXK)
      CALL SECOND (TIM2)
      TIM=TIM2-TIM1
      CALL FACTRS(NPEQ,NEQ,CM,IP,IX,11,12,13,14)
      GO TO 323
C
C     N.G.F. - FILL B, C, AND D AND FACTOR D-C(INV(A)B)
C
  324 CALL CMNGF(CM(IB11),CM(IC11),CM(ID11),NPBX,NEQ,NEQ2,RKH,IEXK)
      CALL SECOND (TIM2)
      TIM=TIM2-TIM1
      CALL FACGF(CM,CM(IB11),CM(IC11),CM(ID11),CM(IX11),IP,IX,NP,N1,MP,
     1M1,NEQ,NEQ2)
  323 CALL SECOND (TIM1)
      TIM2=TIM1-TIM2
      PRINT 153, TIM,TIM2
      IGO=3
      NTSOL=0
      IF(IFLOW.NE.12)GO TO 53
C     WRITE N.G.F. FILE
   52 CALL GFOUT
      GO TO 14
C
C     EXCITATION SET UP (RIGHT HAND SIDE, -E INC.)
C
   53 NTHIC=1
      NPHIC=1
      INC=1
      NPRINT=0
   54 IF (IXTYP.EQ.0.OR.IXTYP.EQ.5) GO TO 56
      IF (IPTFLG.LE.0.OR.IXTYP.EQ.4) PRINT 154
      TMP5=TA*XPR5
      TMP4=TA*XPR4
      IF (IXTYP.NE.4) GO TO 55
      TMP1=XPR1/WLAM
      TMP2=XPR2/WLAM
      TMP3=XPR3/WLAM
      TMP6=XPR6/(WLAM*WLAM)
      PRINT 156, XPR1,XPR2,XPR3,XPR4,XPR5,XPR6
```

```
      GO TO 56
55    TMP1=TA*XPR1
      TMP2=TA*XPR2
      TMP3=TA*XPR3
      TMP6=XPR6
      IF (IPTFLG.LE.0) PRINT 155, XPR1,XPR2,XPR3,HPOL(IXTYP),XPR6
56    CALL ETMNS (TMP1,TMP2,TMP3,TMP4,TMP5,TMP6,IXTYP,CUR)
C
C     MATRIX SOLVING  (NETWK CALLS SOLVES)
C
      IF (NONET.EQ.0.OR.INC.GT.1) GO TO 60
      PRINT 158
      ITMP3=0
      ITMP1=NTYP(1)
      DO 59 I=1,2
      IF (ITMP1.EQ.3) ITMP1=2
      IF (ITMP1.EQ.2) PRINT 159
      IF (ITMP1.EQ.1) PRINT 160
      DO 58 J=1,NONET
      ITMP2=NTYP(J)
      IF ((ITMP2/ITMP1).EQ.1) GO TO 57
      ITMP3=ITMP2
      GO TO 58
57    ITMP4=ISEG1(J)
      ITMP5=ISEG2(J)
      IF (ITMP2.GE.2.AND.X11I(J).LE.0.) X11I(J)=WLAM*SQRT((X(ITMP5)-
     1 X(ITMP4))**2+(Y(ITMP5)-Y(ITMP4))**2+(Z(ITMP5)-Z(ITMP4))**2)
      PRINT 157, ITAG(ITMP4),ITMP4,ITAG(ITMP5),ITMP5,X11R(J),X11I(J),
     1X12R(J),X12I(J),X22R(J),X22I(J),PNET(2*ITMP2-1),PNET(2*ITMP2)
58    CONTINUE
      IF (ITMP3.EQ.0) GO TO 60
      ITMP1=ITMP3
59    CONTINUE
60    CONTINUE
      IF (INC.GT.1.AND.IPTFLG.GT.0) NPRINT=1
      CALL NETWK(CM,CM(IB11),CM(IC11),CM(ID11),IP,CUR)
      NTSOL=1
      IF (IPED.EQ.0) GO TO 61
      ITMP1=MHZ+4*(MHZ-1)
      IF (ITMP1.GT.(NORMF-3)) GO TO 61
      FNORM(ITMP1)=REAL(ZPED)
      FNORM(ITMP1+1)=AIMAG(ZPED)
      FNORM(ITMP1+2)=CABS(ZPED)
      FNORM(ITMP1+3)=CANG(ZPED)
      IF (IPED.EQ.2) GO TO 61
      IF (FNORM(ITMP1+2).GT.ZPNORM) ZPNORM=FNORM(ITMP1+2)
61    CONTINUE
C
C     PRINTING STRUCTURE CURRENTS
C
      IF(N.EQ.0)GO TO 308
      IF (IPTFLG.EQ.(-1)) GO TO 63
      IF (IPTFLG.GT.0) GO TO 62
      PRINT 161
      PRINT 162
      GO TO 63
62    IF (IPTFLG.EQ.3.OR.INC.GT.1) GO TO 63
      PRINT 163, XPR3,HPOL(IXTYP),XPR6
63    PLOSS=0.
      ITMP1=0
      JUMP=IPTFLG+1
      DO 69 I=1,N
      CURI=CUR(I)*WLAM
      CMAG=CABS(CURI)
```

```
      PH=CANG(CURI)
      IF (NLOAD.EQ.0.AND.NLODF.EQ.0) GO TO 64
      IF (ABS(REAL(ZARRAY(I))).LT.1.E-20) GO TO 64
      PLOSS=PLOSS+.5*CMAG*CMAG*REAL(ZARRAY(I))*SI(I)
64    IF (JUMP) 68,69,65
65    IF (IPTAG.EQ.0) GO TO 66
      IF (ITAG(I).NE.IPTAG) GO TO 69
66    ITMP1=ITMP1+1
      IF (ITMP1.LT.IPTAGF.OR.ITMP1.GT.IPTAGT) GO TO 69
      IF (IPTFLG.EQ.0) GO TO 68
      IF (IPTFLG.LT.2.OR.INC.GT.NORMF) GO TO 67
      FNORM(INC)=CMAG
      ISAVE=I
67    IF (IPTFLG.NE.3) PRINT 164, XPR1,XPR2,CMAG,PH,I
      GO TO 69
68    PRINT 165, I,ITAG(I),X(I),Y(I),Z(I),SI(I),CURI,CMAG,PH
69    CONTINUE
      IF(IPTFLQ.EQ.(-1))GO TO 308
      PRINT 315
      ITMP1=0
      FR=1.E-6/FMHZ
      DO 316 I=1,N
      IF(IPTFLQ.EQ.(-2))GO TO 318
      IF(IPTAQ.EQ.0)GO TO 317
      IF(ITAG(I).NE.IPTAQ)GO TO 316
317   ITMP1=ITMP1+1
      IF(ITMP1.LT.IPTAQF.OR.ITMP1.GT.IPTAQT)GO TO 316
318   CURI=FR*CMPLX(-BII(I),BIR(I))
      CMAG=CABS(CURI)
      PH=CANG(CURI)
      PRINT 165,I,ITAG(I),X(I),Y(I),Z(I),SI(I),CURI,CMAG,PH
316   CONTINUE
308   IF(M.EQ.0)GO TO 310
      PRINT 197
      J=N-2
      ITMP1=LD+1
      DO 309 I=1,M
      J=J+3
      ITMP1=ITMP1-1
      EX=CUR(J)
      EY=CUR(J+1)
      EZ=CUR(J+2)
      ETH=EX*T1X(ITMP1)+EY*T1Y(ITMP1)+EZ*T1Z(ITMP1)
      EPH=EX*T2X(ITMP1)+EY*T2Y(ITMP1)+EZ*T2Z(ITMP1)
      ETHM=CABS(ETH)
      ETHA=CANG(ETH)
      EPHM=CABS(EPH)
      EPHA=CANG(EPH)
309   PRINT 198,I,X(ITMP1),Y(ITMP1),Z(ITMP1),ETHM,ETHA,EPHM,EPHA,EX,EY,
     1 EZ
310   IF (IXTYP.NE.0.AND.IXTYP.NE.5) GO TO 70
      TMP1=PIN-PNLS-PLOSS
      TMP2=100.*TMP1/PIN
      PRINT 166, PIN,TMP1,PLOSS,PNLS,TMP2
70    CONTINUE
      IGO=4
      IF(NCOUP.GT.0)CALL COUPLE(CUR,WLAM)
      IF (IFLOW.NE.7) GO TO 71
      IF (IXTYP.GT.0.AND.IXTYP.LT.4) GO TO 113
      IF (NFRQ.NE.1) GO TO 120
      PRINT 135
      GO TO 14
71    IGO=5
C
```

```
C     NEAR FIELD CALCULATION
C
72    IF (NEAR.EQ.(-1)) GO TO 78
      CALL NFPAT
      IF (MHZ.EQ.NFRQ) NEAR=-1
      IF (NFRQ.NE.1) GO TO 78
      PRINT 135
      GO TO 14
C
C     STANDARD FAR FIELD CALCULATION
C
78    IF(IFAR.EQ.-1)GO TO 113
      PINR=PIN
      PNLR=PNLS
      CALL RDPAT
113   IF (IXTYP.EQ.0.OR.IXTYP.GE.4) GO TO 119
      NTHIC=NTHIC+1
      INC=INC+1
      XPR1=XPR1+XPR4
      IF (NTHIC.LE.NTHI) GO TO 54
      NTHIC=1
      XPR1=THETIS
      XPR2=XPR2+XPR5
      NPHIC=NPHIC+1
      IF (NPHIC.LE.NPHI) GO TO 54
      NPHIC=1
      XPR2=PHISS
      IF (IPTFLG.LT.2) GO TO 119
C     NORMALIZED RECEIVING PATTERN PRINTED
      ITMP1=NTHI*NPHI
      IF (ITMP1.LE.NORMF) GO TO 114
      ITMP1=NORMF
      PRINT 181
114   TMP1=FNORM(1)
      DO 115 J=2,ITMP1
      IF (FNORM(J).GT.TMP1) TMP1=FNORM(J)
115   CONTINUE
      PRINT 182, TMP1,XPR3,HPOL(IXTYP),XPR6,ISAVE
      DO 118 J=1,NPHI
      ITMP2=NTHI*(J-1)
      DO 116 I=1,NTHI
      ITMP3=I+ITMP2
      IF (ITMP3.GT.ITMP1) GO TO 117
      TMP2=FNORM(ITMP3)/TMP1
      TMP3=DB20(TMP2)
      PRINT 183, XPR1,XPR2,TMP3,TMP2
      XPR1=XPR1+XPR4
116   CONTINUE
117   XPR1=THETIS
      XPR2=XPR2+XPR5
118   CONTINUE
      XPR2=PHISS
119   IF (MHZ.EQ.NFRQ) IFAR=-1
      IF (NFRQ.NE.1) GO TO 120
      PRINT 135
      GO TO 14
120   MHZ=MHZ+1
      IF (MHZ.LE.NFRQ) GO TO 42
      IF (IPED.EQ.0) GO TO 123
      IF(NVQD.LT.1)GO TO 199
      PRINT 184,IVQD(NVQD),ZPNORM
      GO TO 204
199   PRINT 184, ISANT(NSANT),ZPNORM
204   ITMP1=NFRQ
```

```
          IF (ITMP1.LE.(NORMF/4)) GO TO 121                            MA 768
          ITMP1=NORMF/4                                                MA 769
          PRINT 185                                                    MA 770
  121     IF (IFRQ.EQ.0) TMP1=FMHZ-(NFRQ-1)*DELFRQ                     MA 771
          IF (IFRQ.EQ.1) TMP1=FMHZ/(DELFRQ**(NFRQ-1))                  MA 772
          DO 122 I=1,ITMP1                                             MA 773
          ITMP2=I+4*(I-1)                                              MA 774
          TMP2=FNORM(ITMP2)/ZPNORM                                     MA 775
          TMP3=FNORM(ITMP2+1)/ZPNORM                                   MA 776
          TMP4=FNORM(ITMP2+2)/ZPNORM                                   MA 777
          TMP5=FNORM(ITMP2+3)                                          MA 778
          PRINT 186, TMP1,FNORM(ITMP2),FNORM(ITMP2+1),FNORM(ITMP2+2),  MA 779
         1FNORM(ITMP2+3),TMP2,TMP3,TMP4,TMP5                           MA 780
          IF (IFRQ.EQ.0) TMP1=TMP1+DELFRQ                              MA 781
          IF (IFRQ.EQ.1) TMP1=TMP1*DELFRQ                              MA 782
  122     CONTINUE                                                     MA 783
          PRINT 135                                                    MA 784
  123     CONTINUE                                                     MA 785
          NFRQ=1                                                       MA 786
          MHZ=1                                                        MA 787
          GO TO 14                                                     MA 788
  125     FORMAT (A2,13A6)                                             MA 789
  126     FORMAT  (1H1)                                                MA 790
  127     FORMAT (///,33X,36H************************************,//,36X,  MA 791
         1 31HNUMERICAL ELECTROMAGNETICS CODE,//,33X,                  MA 792
         2 36H************************************)                    MA 793
  128     FORMAT (////,37X,24H- - - - COMMENTS - - - -,//)             MA 794
  129     FORMAT (25X,13A6)                                            MA 795
  130     FORMAT (///,10X,34HINCORRECT LABEL FOR A COMMENT CARD)       MA 796
  135     FORMAT (/////)                                               MA 797
  136     FORMAT (A2,I3,3I5,6E10.3)                                    MA 798
  137     FORMAT (1X, 19H***** DATA CARD NO.,I3,3X,A2,1X,I3,3(1X,I5),  MA 799
         1 6(1X,E12.5))                                                MA 800
  138     FORMAT (///,10X,45HFAULTY DATA CARD LABEL AFTER GEOMETRY SECTION)  MA 801
  139     FORMAT (///,10X,48HNUMBER OF LOADING CARDS EXCEEDS STORAGE ALLOTTE MA 802
         1D)                                                           MA 803
  140     FORMAT (///,10X,31HDATA FAULT ON LOADING CARD NO.=,I5,5X,11HITAG S MA 804
         1TEP1=,I5,29H  IS GREATER THAN ITAG STEP2=,I5)                MA 805
  141     FORMAT (///,10X,51HNUMBER OF EXCITATION CARDS EXCEEDS STORAGE ALLO MA 806
         1TTED)                                                        MA 807
  142     FORMAT (///,10X,48HNUMBER OF NETWORK CARDS EXCEEDS STORAGE ALLOTTE MA 808
         1D)                                                           MA 809
  143     FORMAT(///,10X,79HWHEN MULTIPLE FREQUENCIES ARE REQUESTED, ONLY ON MA 810
         1E NEAR FIELD CARD CAN BE USED -,/,10X,22HLAST CARD READ IS USED)  MA 811
  145     FORMAT (////,33X,33H- - - - - - FREQUENCY - - - - - -,//,36X,10HFR MA 812
         1EQUENCY=,E11.4,4H MHZ,/,36X,11HWAVELENGTH=,E11.4,7H METERS)  MA 813
  146     FORMAT (///,30X,40H - - - STRUCTURE IMPEDANCE LOADING - - -)  MA 814
  147     FORMAT (/ ,35X,28HTHIS STRUCTURE IS NOT LOADED)              MA 815
  148     FORMAT (///,34X,31H- - - ANTENNA ENVIRONMENT - - -,/)        MA 816
  149     FORMAT (40X,21HMEDIUM UNDER SCREEN -)                        MA 817
  150     FORMAT (40X,27HRELATIVE DIELECTRIC CONST.=,F7.3,/,40X,13HCONDUCTIV MA 818
         1ITY=,E10.3,11H MHOS/METER,/,40X,28HCOMPLEX DIELECTRIC CONSTANT=,  MA 819
         12E12.5)                                                      MA 820
  151     FORMAT (  42X,14HPERFECT GROUND)                             MA 821
  152     FORMAT (  44X,10HFREE SPACE)                                 MA 822
  153     FORMAT (///,32X,25H- - - MATRIX TIMING - - -,//,24X,5HFILL=,F9.3,  MA 823
         115H SEC.,  FACTOR=,F9.3,5H SEC.)                             MA 824
  154     FORMAT (///,40X,22H- - - EXCITATION - - -)                   MA 825
  155     FORMAT (/,4X,10HPLANE WAVE,4X,6HTHETA=,F7.2,11H DEG,  PHI=,F7.2,  MA 826
         1 11H DEG,  ETA=,F7.2,13H DEG,  TYPE -,A6,15H=  AXIAL RATIO=,F6.3)  MA 827
  156     FORMAT (/,31X,17HPOSITION (METERS),14X,18HORIENTATION (DEG)=/,28X,  MA 828
         11HX,12X,1HY,12X,1HZ,10X,5HALPHA,5X,4HBETA,4X,13HDIPOLE MOMENT,//  MA 829
         2 ,4X,14HCURRENT SOURCE,1X,3(3X,F10.5),1X,2(3X,F7.2),4X,F8.3)  MA 830
  157     FORMAT (4X,4(I5,1X),6(3X,E11.4),3X,A6,A2)                    MA 831
```

```
  158 FORMAT (///,44X,24H- - - NETWORK DATA - - -)                      MA 832
  159 FORMAT (/,6X,18H- FROM -    - TO -,11X,17HTRANSMISSION LINE,15X,36 MA 833
     1H-  -  SHUNT ADMITTANCES (MHOS)  -  -,14X,4HLINE,/,6X,21HTAG  SEG. MA 834
     2   TAG  SEG.,6X,9HIMPEDANCE,6X,6HLENGTH,12X,11H- END ONE -,17X,11H MA 835
     3- END TWO -,12X,4HTYPE,/    ,6X,21HNO.   NO.   NO.   NO.,9X,4HOHMS MA 836
     4,8X,6HMETERS,9X, 4HREAL,10X,5HIMAG.,9X,4HREAL,10X,5HIMAG.)        MA 837
  160 FORMAT (/,6X,8H- FROM -,4X,6H- TO -,26X,45H-  -  ADMITTANCE MATRIX MA 838
     1 ELEMENTS (MHOS)  -  -,/    ,6X,21HTAG  SEG.   TAG  SEG.,13X,9H(ON MA 839
     2E,ONE),19X,    9H(ONE,TWO),19X,9H(TWO,TWO),/ ,6X,21HNO.   NO.   NO MA 840
     3.   NO.,8X,4HREAL,10X,5HIMAG.,9X,4HREAL,10X,5HIMAG.,9X,4HREAL,    MA 841
     4 10X,5HIMAG.)                                                     MA 842
  161 FORMAT (///,29X,33H- - - CURRENTS AND LOCATION - - -,//,33X,24HDIS MA 843
     1TANCES IN WAVELENGTHS)                                            MA 844
  162 FORMAT (  //,2X,4HSEG.,2X,3HTAG,4X,21HCOORD. OF SEG. CENTER,5X,   MA 845
     1 4HSEG.,12X,26H- - - CURRENT (AMPS) - - -,/,2X,3HNO.,3X,3HNO.,    MA 846
     2 5X,1HX,8X,1HY,8X,1HZ,6X,6HLENGTH,5X,4HREAL,8X,5HIMAG.,7X,4HMAG., MA 847
     3 8X,5HPHASE)                                                      MA 848
  163 FORMAT (///,33X,40H- - - RECEIVING PATTERN PARAMETERS - - -,/  ,43 MA 849
     1X,4HETA=,F7.2,8H DEGREES,/,43X,6HTYPE -,A6,/,43X,12HAXIAL RATIO=, MA 850
     2 F6.3,//   ,11X,5HTHETA,6X,3HPHI,10X,13H-  CURRENT  -,9X,3HSEG,/  MA 851
     3,11X,5H(DEG),5X,5H(DEG),7X,9HMAGNITUDE,4X,5HPHASE,6X,3HNO.,/)     MA 852
  164 FORMAT (10X,2(F7.2,3X),1X,E11.4,3X,F7.2,4X,I5)                    MA 853
  165 FORMAT (1X,2I5,3F9.4,F9.5,1X,3E12.4,F9.3)                         MA 854
  166 FORMAT (///,40X,24H- - - POWER BUDGET - - -,//     ,43X,15HINPUT PO MA 855
     1WER    =,E11.4,6H WATTS,/ ,43X,15HRADIATED POWER=,E11.4,6H WATTS,/ MA 856
     2 ,43X,15HSTRUCTURE LOSS=,E11.4,6H WATTS,/ ,43X,15HNETWORK LOSS  =, MA 857
     3  E11.4,6H WATTS,/,43X,15HEFFICIENCY    =, F7.2,8H PERCENT)       MA 858
  170 FORMAT (40X,25HRADIAL WIRE GROUND SCREEN,/,40X,   I5,6H WIRES,/,40 MA 859
     1X,12HWIRE LENGTH=,F8.2,7H METERS,/,40X,12HWIRE RADIUS=,E10.3,7H ME MA 860
     2TERS)                                                             MA 861
  181 FORMAT (///,4X,51HRECEIVING PATTERN STORAGE TOO SMALL,ARRAY TRUNCA MA 862
     1TED)                                                              MA 863
  182 FORMAT (///,32X,40H- - - NORMALIZED RECEIVING PATTERN - - -,/,41X, MA 864
     1 21HNORMALIZATION FACTOR=,E11.4,/,41X,4HETA=,F7.2,8H DEGREES,/,41X MA 865
     2,6HTYPE -,A6,/,41X,12HAXIAL RATIO=,F6.3,/,41X,12HSEGMENT NO.=,I5,/ MA 866
     3/,21X,5HTHETA,6X,3HPHI,9X,13H-  PATTERN  -,/,21X,5H(DEG),5X,5H(DEG MA 867
     4),8X,2HDB,8X,9HMAGNITUDE,/)                                       MA 868
  183 FORMAT (20X,2(F7.2,3X),1X,F7.2,4X,E11.4)                          MA 869
  184 FORMAT (///,36X,32H- - - INPUT IMPEDANCE DATA - - -,/    ,45X,18HSO MA 870
     1URCE SEGMENT NO.,I4,/  ,45X,21HNORMALIZATION FACTOR=,E12.5,//     MA 871
     2,7X,5HFREQ.,13X,34H-  -  UNNORMALIZED IMPEDANCE  -  -,21X,   32H- MA 872
     3 -  NORMALIZED IMPEDANCE  -  -,/     ,19X,10HRESISTANCE,4X,9HREACTA MA 873
     4NCE,6X,9HMAGNITUDE,4X,5HPHASE,7X,10HRESISTANCE,4X,9HREACTANCE,6X, MA 874
     5 9HMAGNITUDE,4X,5HPHASE,/     ,8X,3HMHZ,11X,4HOHMS,10X,4HOHMS,11X, MA 875
     6 4HOHMS,5X,7HDEGREES,47X,7HDEGREES,/)                             MA 876
  185 FORMAT (///,4X,62HSTORAGE FOR IMPEDANCE NORMALIZATION TOO SMALL, A MA 877
     1RRAY TRUNCATED)                                                   MA 878
  186 FORMAT (3X,F9.3,2X,2(2X,E12.5),3X,E12.5,2X,F7.2,2X,2(2X,E12.5),3X, MA 879
     1 E12.5,2X,F7.2)                                                   MA 880
  196 FORMAT(   ////,20X,55HAPPROXIMATE INTEGRATION EMPLOYED FOR SEGMENT MA 881
     1S MORE THAN,F8.3,18H WAVELENGTHS APART)                           MA 882
  197 FORMAT(   ////,41X,38H- - - - SURFACE PATCH CURRENTS - - - -,//,  MA 883
     1 50X,23HDISTANCE IN WAVELENGTHS,/,50X,21HCURRENT IN AMPS/METER,   MA 884
     1 //,28X,26H- - SURFACE COMPONENTS - -,19X,34H- - - RECTANGULAR COM MA 885
     1PONENTS - - -,/,6X,12HPATCH CENTER,6X,16HTANGENT VECTOR 1,3X,     MA 886
     116HTANGENT VECTOR 2,11X,1HX,19X,1HY,19X,1HZ,/,5X,1HX,6X,1HY,6X,   MA 887
     11HZ,5X,4HMAG.,7X,5HPHASE,3X,4HMAG.,7X,5HPHASE,3(4X,4HREAL,6X,     MA 888
     1 6HIMAG. ))                                                       MA 889
  198 FORMAT(1X,I4,/,1X,3F7.3,2(E11.4,F8.2),6E10.2)                     MA 890
  201 FORMAT(/,11H RUN TIME =,F10.3)                                    MA 891
  315 FORMAT(///,34X,28H- - - CHARGE DENSITIES - - -,//,36X,            MA 892
     1 24HDISTANCES IN WAVELENGTHS,///,2X,4HSEG.,2X,3HTAG,4X,           MA 893
     2 21HCOORD. OF SEG. CENTER,5X,4HSEG.,10X,                          MA 894
     3 31HCHARGE DENSITY (COULOMBS/METER),/,2X,3HNO.,3X,3HNO.,5X,1HX,8X, MA 895
```

```
     4 1HY,8X,1HZ,6X,6HLENGTH,5X,4HREAL,8X,5HIMAG.,7X,4HMAG.,8X,5HPHASE) MA 896
 321  FORMAT( /,20X,42HTHE EXTENDED THIN WIRE KERNEL WILL BE USED)        MA 897
 303  FORMAT(/,9H ERROR - ,A2,32H CARD IS NOT ALLOWED WITH N.G.F.)        MA 898
 327  FORMAT(/,35X,31H LOADING ONLY IN N.G.F. SECTION)                    MA 899
 302  FORMAT(48H ERROR - N.G.F. IN USE.  CANNOT WRITE NEW N.G.F.)         MA 900
 313  FORMAT(/,62H NUMBER OF SEGMENTS IN COUPLING CALCULATION (CP) EXCEE MA 901
     1DS LIMIT)                                                           MA 902
 390  FORMAT(78H RADIAL WIRE G. S. APPROXIMATION MAY NOT BE USED WITH SO MA 903
     1MMERFELD GROUND OPTION)                                             MA 904
 391  FORMAT(40X,52HFINITE GROUND.  REFLECTION COEFFICIENT APPROXIMATION MA 905
     1)                                                                   MA 906
 392  FORMAT(40X,35HFINITE GROUND.  SOMMERFELD SOLUTION)                  MA 907
 393  FORMAT(/,29H ERROR IN GROUND PARAMETERS -,/,41H COMPLEX DIELECTRIC MA 908
     1 CONSTANT FROM FILE IS,2E12.5,/,32X,9HREQUESTED,2E12.5)             MA 909
      END                                                                 MA 910-
```

ARC

PURPOSE

To fill COMMON/DATA/ with segment coordinates for a circular arc of segments.

METHOD

The formal parameters specify the number of segments, radius of the arc, starting angle, final angle and wire radius. Segment coordinates are computed for the arc in the x, z plane with a left hand rotation about the y axis.

SYMBOL DICTIONARY

ANG	= angle of point on the arc (radians, zero on x axis)
ANG1	= angle at first end
ANG2	= angle at second end
DANG	= angle covered by each segment
IST	= number of initial segment
ITG	= tag number assigned to each segment
NS	= number of segments
RAD	= wire radius
RADA	= arc radius
TA	= $\pi/180$
XS1	= x coordinate of first end of segment
XS2	= x coordinate of second end of segment
ZS1	= z coordinate of first end of segment
ZS2	= z coordinate of second end of segment

CONSTANTS

.01745329252	= $\pi/180$
360.00001	= test for angle greater than 360 degrees

```
      SUBROUTINE ARC (ITG,NS,RADA,ANG1,ANG2,RAD)
C
C     ARC GENERATES SEGMENT GEOMETRY DATA FOR AN ARC OF NS SEGMENTS
C
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      DIMENSION X2(1), Y2(1), Z2(1)
      EQUIVALENCE (X2,SI), (Y2,ALP), (Z2,BET)
      DATA TA/.01745329252/
      IST=N+1
      N=N+NS
      NP=N
      MP=M
      IPSYM=0
      IF (NS.LT.1) RETURN
      IF (ABS(ANG2-ANG1).LT.360.00001) GO TO 1
      PRINT 3
      STOP
1     ANG=ANG1*TA
      DANG=(ANG2-ANG1)*TA/NS
      XS1=RADA*COS(ANG)
      ZS1=RADA*SIN(ANG)
      DO 2 I=IST,N
      ANG=ANG+DANG
      XS2=RADA*COS(ANG)
      ZS2=RADA*SIN(ANG)
      X(I)=XS1
      Y(I)=0.
      Z(I)=ZS1
      X2(I)=XS2
      Y2(I)=0.
      Z2(I)=ZS2
      XS1=XS2
      ZS1=ZS2
      BI(I)=RAD
2     ITAG(I)=ITG
      RETURN
C
3     FORMAT (40H ERROR -- ARC ANGLE EXCEEDS 360. DEGREES)
      END
```

ATGN2

PURPOSE

To return zero when both arguments of a two-argument arctangent function are zero. (Most standard arctangent functions give an error return when both arguments are zero.)

METHOD

System function ATAN2 is used except when both arguments are zero, in which case the value zero is returned. The value returned is the angle (in radians) whose sine is X and cosine is Y.

SYMBOL DICTIONARY

X = first argument

Y = second argument

CODE LISTING

```
      FUNCTION ATGN2 (X,Y)                                           AT   1
C                                                                    AT   2
C     ATGN2 IS ARCTANGENT FUNCTION MODIFIED TO RETURN 0. WHEN X=Y=0. AT   3
C                                                                    AT   4
      IF (X) 3,1,3                                                   AT   5
    1 IF (Y) 3,2,3                                                   AT   6
    2 ATGN2=0.                                                       AT   7
      RETURN                                                         AT   8
    3 ATGN2=ATAN2(X,Y)                                               AT   9
      RETURN                                                         AT  10
      END                                                            AT  11-
```

BLCKOT

PURPOSE

To control the writing and reading of matrix blocks on files for the out-of-core matrix solution. The routine also checks for the end-of-file condition during reading.

METHOD

The routine uses a binary read and write with implied DO loops for reading and writing variable length strings into and out of various core locations. The end-of-file condition is checked by a call to function ENF. If an unexpected end of file is detected (governed by NEOF) the program stops.

CODING

BL9 - BL12 Write a record on file NUNIT.
BL14 - BL20 Read NBLKS records from NUNIT, and check for end of file.
BL21 - BL24 Code if end of file detected.

SYMBOL DICTIONARY

AR = matrix array
ENF = external function (checks end-of-file condition)
I = DO loop index
I1, I2 = implied DO loop limits, inclusive matrix locations written from or read into
J = implied DO index
NBLKS = number of records to be read
NEOF = EOF check flag, also used to trace the call to BLCKOT
NUNIT = file number

CONSTANT

777 = NEOF when EOF is expected by calling program

```
      SUBROUTINE BLCKOT (AR,NUNIT,IX1,IX2,NBLKS,NEOF)
C
C     BLCKOT CONTROLS THE READING AND WRITING OF MATRIX BLOCKS ON FILES
C     FOR THE OUT-OF-CORE MATRIX SOLUTION.
C
      LOGICAL ENF
      COMPLEX AR
      DIMENSION AR(1)
      I1=(IX1+1)/2
      I2=(IX2+1)/2
1     WRITE (NUNIT) (AR(J),J=I1,I2)
      RETURN
      ENTRY BLCKIN
      I1=(IX1+1)/2
      I2=(IX2+1)/2
      DO 2 I=1,NBLKS
      READ (NUNIT) (AR(J),J=I1,I2)
      IF (ENF(NUNIT)) GO TO 3
2     CONTINUE
      RETURN
3     PRINT 4, NUNIT,NBLKS,NEOF
      IF (NEOF.NE.777) STOP
      NEOF=0
      RETURN
C
4     FORMAT (13H  EOF ON UNIT,I3,9H  NBLKS= ,I3,8H  NEOF= ,I5)
      END
```

CABC

PURPOSE

To compute the coefficients in the current function on each segment, given the basis function amplitudes. Surface current components are also computed.

METHOD

The total current on segment i is

$$I_i(s) = A_i + B_i \sin [k(s - s_i)] + C_i \cos [k(s - s_i)] ,$$

where s is distance along the wire, and $s = s_i$ at the center of segment i. The coefficients A_i, B_i, and C_i are the sums of the corresponding coefficients in the portion of each basis function that extends onto segment i.

CODING

CB35 Call to TBF computes components of basis function I.

CB36 - CB43 The basis function components are multiplied by the basis function amplitude from array CURX and summed for each segment.

CB45 - CB63 For a current slope discontinuity source, the special basis function with discontinuous slope, from which the exciting electric field was computed, is recomputed and added to the current coefficients. The call to TBF, with the second argument zero and ICON1(I) temporarily zero, computes a basis function going to zero with non-zero derivative at end one of segment I.

CB64 - CB65 Total current at the center of each segment is computed and stored in place of the basis function amplitudes.

CB68 - CB79 The $\hat{t}_1$ and $\hat{t}_2$ components of surface current for each patch are expanded to x, y, and z components.

SYMBOL DICTIONARY

AR, AI = real and imaginary parts of the basis function amplitude

CCJ, CCX = $-j/60$

CS1, CS2 = $\hat{t}_1$ and $\hat{t}_2$ components of surface current on a patch

CURD = amplitude of the special basis function for a current slope discontinuity source

CURX = input array of basis function amplitudes that are replaced by values of current at segment centers

J = number of a segment onto which a basis function extends

JCO1, JCO2 = array locations of the $\hat{t}_1$ and $\hat{t}_2$ surface current components for a patch

JX = DO loop index; temporary storage of connection number

K = array location for patch geometry data

SH = (half segment length)/λ

TP = 2π

```
      SUBROUTINE CABC (CURX)
C
C     CABC COMPUTES COEFFICIENTS OF THE CONSTANT (A), SINE (B), AND
C     COSINE (C) TERMS IN THE CURRENT INTERPOLATION FUNCTIONS FOR THE
C     CURRENT VECTOR CUR.
C
      COMPLEX CUR,CURX,VQDS,CURD,CCJ,VSANT,VQD,CS1,CS2
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /CRNT/ AIR(300),AII(300),BIR(300),BII(300),CIR(300),CII(300
     1),CUR(900)
      COMMON /SEGJ/ AX(30),BX(30),CX(30),JCO(30),JSNO,ISCON(50),NSCON,IP
     1CON(10),NPCON
      COMMON /VSORC/ VQD(30),VSANT(30),VQDS(30),IVQD(30),ISANT(30),IQDS(
     130),NVQD,NSANT,NQDS
      COMMON /ANGL/ SALP(300)
      DIMENSION T1X(1), T1Y(1), T1Z(1), T2X(1), T2Y(1), T2Z(1)
      DIMENSION CURX(1), CCJX(2)
      EQUIVALENCE (T1X,SI), (T1Y,ALP), (T1Z,BET), (T2X,ICON1), (T2Y,ICON
     12), (T2Z,ITAG)
      EQUIVALENCE (CCJ,CCJX)
      DATA TP/6.283185308/,CCJX/0.,-0.01666666667/
      IF (N.EQ.0) GO TO 6
      DO 1 I=1,N
      AIR(I)=0.
      AII(I)=0.
      BIR(I)=0.
      BII(I)=0.
      CIR(I)=0.
1     CII(I)=0.
      DO 2 I=1,N
      AR=REAL(CURX(I))
      AI=AIMAG(CURX(I))
      CALL TBF (I,1)
      DO 2 JX=1,JSNO
      J=JCO(JX)
      AIR(J)=AIR(J)+AX(JX)*AR
      AII(J)=AII(J)+AX(JX)*AI
      BIR(J)=BIR(J)+BX(JX)*AR
      BII(J)=BII(J)+BX(JX)*AI
      CIR(J)=CIR(J)+CX(JX)*AR
2     CII(J)=CII(J)+CX(JX)*AI
      IF (NQDS.EQ.0) GO TO 4
      DO 3 IS=1,NQDS
      I=IQDS(IS)
      JX=ICON1(I)
      ICON1(I)=0
      CALL TBF (I,0)
      ICON1(I)=JX
      SH=SI(I)*.5
      CURD=CCJ*VQDS(IS)/((ALOG(2.*SH/BI(I))-1.)*(BX(JSNO)*COS(TP*SH)+CX(
     1JSNO)*SIN(TP*SH))*WLAM)
      AR=REAL(CURD)
      AI=AIMAG(CURD)
      DO 3 JX=1,JSNO
      J=JCO(JX)
      AIR(J)=AIR(J)+AX(JX)*AR
      AII(J)=AII(J)+AX(JX)*AI
      BIR(J)=BIR(J)+BX(JX)*AR
      BII(J)=BII(J)+BX(JX)*AI
      CIR(J)=CIR(J)+CX(JX)*AR
3     CII(J)=CII(J)+CX(JX)*AI
4     DO 5 I=1,N
```

```
    5 CURX(I)=CMPLX(AIR(I)+CIR(I),AII(I)+CII(I))                          CB  65
    6 IF (M.EQ.0) RETURN                                                  CB  66
C     CONVERT SURFACE CURRENTS FROM T1,T2 COMPONENTS TO X,Y,Z COMPONENTS  CB  67
      K=LD-M                                                              CB  68
      JCO1=N+2*M+1                                                        CB  69
      JCO2=JCO1+M                                                         CB  70
      DO 7 I=1,M                                                          CB  71
      K=K+1                                                               CB  72
      JCO1=JCO1-2                                                         CB  73
      JCO2=JCO2-3                                                         CB  74
      CS1=CURX(JCO1)                                                      CB  75
      CS2=CURX(JCO1+1)                                                    CB  76
      CURX(JCO2)=CS1*T1X(K)+CS2*T2X(K)                                    CB  77
      CURX(JCO2+1)=CS1*T1Y(K)+CS2*T2Y(K)                                  CB  78
    7 CURX(JCO2+2)=CS1*T1Z(K)+CS2*T2Z(K)                                  CB  79
      RETURN                                                              CB  80
      END                                                                 CB  81-
```

CANG

PURPOSE

To calculate the phase angle of a complex number in degrees.

METHOD

$z = x + jy$

$\phi = [\arctan(y/x)]\ 57.29577951$

SYMBOL DICTIONARY

AIMAG = external routine (imaginary part of complex number)

ATGN2 = external routine (arctan for all quadrants)

CANG = ϕ

REAL = external routine (real part of a complex number)

Z = input complex quantity

CONSTANT

57.29577951 conversion from radians to degrees

CODE LISTING

```
      FUNCTION CANG (Z)                                                 CA   1
C                                                                       CA   2
C     CANG RETURNS THE PHASE ANGLE OF A COMPLEX NUMBER IN DEGREES.      CA   3
C                                                                       CA   4
      COMPLEX Z                                                         CA   5
      CANG=ATGN2(AIMAG(Z),REAL(Z))*57.29577951                          CA   6
      RETURN                                                            CA   7
      END                                                               CA   8-
```

CMNGF

PURPOSE

To compute and store the matrices B, C and D for the NGF solution.

METHOD

The structure of matrices B, C and D is described in Section VI. The coding to fill these matrices is involved due to their complex structure, as shown in Figure 12 of Section VI. The complexity is increased by the need to divide the matrices into blocks of rows when they are stored on files (see Section VII).

Much of the coding in CMNGF has to do with connections between new and NGF segments and patches. When a new segment or patch connects to a NGF segment the basis function associated with the NGF segment is modified due to the new junction condition. The amplitude of the modified basis function is a new unknown associated with the B' and D' sections of the matrix. The modified basis function may extend onto other NGF segments that may or may not connect directly to new segments. Also, the basis function of the new segment extends onto the NGF segment to which it connects. Hence fields must be computed for the currents on some NGF segments as well as all new segments.

Comments in the code should be of some help in understanding the procedure. The notation D(WS) in the comments corresponds to D_{sw} in Figure 12. Some parts of the code are explained below.

CG61 - CG70 TRIO computes the components of all basis functions on segment J, where J is a new segment, and stores the coefficients in COMMON/SEGJ/. The array JCO contains the basis-function numbers which ordinarily are the matrix columns associated with the basis functions. If the basis function is for a new segment then JCO is set at CG66 to the column relative to the beginning of the matrix B. If the basis function is for a NGF segment modified by the connection, then JCO is set at CG68 to the column in B'_{ww} relative to the beginning of B. Thus the calls to CMWW and CMWS may store contributions in B'_{ww} and B'_{sw} as well as B_{ww} and B_{sw}.

CG90 - CG108 In this section the fields are evaluated for NGF segments that connect to new segments or patches. TRIO finds all basis functions that contribute to the current on the segment. For a component of a new basis function IR is set to the column in B_{ww} at CG95. For a component of a modified basis function IR is set to the column in B'_{ww}, relative to the start of B, at CG99. If the basis function component is for a NGF basis function that has not been modified the test at CG98 skips to the end of the loop. The arrays in COMMON/SEGJ/ are adjusted from CG101 to CG104 so that CMWW and CMWS will store the matrix element contributions in the correct locations.

CG109 - CG119 If a NGF segment connects to a new segment on one end and to a NGF patch on the opposite end the modified basis function extends onto the patch as a singular component of the patch current. The field due to this component on the patch is added to the matrix element of the modified basis function at CG119.

CG122-CG136 This is similar to CG90 to CG108, but evaluates fields of NGF segments that get contributions from modified basis functions, but do not connect directly to new segments. TBF is called, rather than TRIO to compute modified basis function J on all segments on which it exists. New segments and NGF segments for which contributions have already been evaluated are skipped at CG133 and CG134.

CG165 - CG263 Filling C and D is similar to that for B but fields must be evaluated for all NGF segments and patches as well as new segments and patches.

SYMBOL DICTIONARY

CB = array for matrix B

CC = array for matrix C

CD = array for matrix D

IEXKX = flag to select extended thin-wire kernel

MIEQ = number of patch equations in NGF

MEQ = total number of patch equations

NB = row dimension of CB. CB will contain only one block of B when ICASX = 3 or 4

NC = row dimension of CC (C transposed)

ND = row dimension of CD (D transposed)

NEQN = starting column of D_{ws}, relative to start of C

NEQP = starting column of zeros after D'_{ww}, relative to start of D

NEQS = starting column of D'_{ww}, relative to start of D

NEQSP = starting column of D'_{ww}, relative to start of C

RKHX = minimum range for using the lumped current approximation for the field of a segment

```
      SUBROUTINE CMNGF (CB,CC,CD,NB,NC,ND,RKHX,IEXKX)
C     CMNGF FILLS INTERACTION MATRICIES B, C, AND D FOR N.G.F. SOLUTION
      COMPLEX CB,CC,CD,ZARRAY,EXK,EYK,EZK,EXS,EYS,EZS,EXC,EYC,EZC
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /ZLOAD/ ZARRAY(300),NLOAD,NLODF
      COMMON /SEGJ/ AX(30),BX(30),CX(30),JCO(30),JSNO,ISCON(50),NSCON,IP
     1CON(10),NPCON
      COMMON /DATAJ/ S,B,XJ,YJ,ZJ,CABJ,SABJ,SALPJ,EXK,EYK,EZK,EXS,EYS,EZ
     1S,EXC,EYC,EZC,RKH,IEXK,IND1,IND2,IPGND
      COMMON /MATPAR/ ICASE,NBLOKS,NPBLK,NLAST,NBLSYM,NPSYM,NLSYM,IMAT,I
     1CASX,NBBX,NPBX,NLBX,NBBL,NPBL,NLBL
      DIMENSION CB(NB,1), CC(NC,1), CD(ND,1)
      RKH=RKHX
      IEXK=IEXKX
      M1EQ=2*M1
      M2EQ=M1EQ+1
      MEQ=2*M
      NEQP=ND-NPCON*2
      NEQS=NEQP-NSCON
      NEQSP=NEQS+NC
      NEQN=NC+N-N1
      ITX=1
      IF (NSCON.GT.0) ITX=2
      IF (ICASX.EQ.1) GO TO 1
      REWIND 12
      REWIND 14
      REWIND 15
      IF (ICASX.GT.2) GO TO 5
1     DO 4 J=1,ND
      DO 2 I=1,ND
2     CD(I,J)=(0.,0.)
      DO 3 I=1,NB
      CB(I,J)=(0.,0.)
3     CC(I,J)=(0.,0.)
4     CONTINUE
5     IST=N-N1+1
      IT=NPBX
      ISV=-NPBX
C     LOOP THRU 24 FILLS B.  FOR ICASX=1 OR 2 ALSO FILLS D(WW), D(WS)
      DO 24 IBLK=1,NBBX
      ISV=ISV+NPBX
      IF (IBLK.EQ.NBBX) IT=NLBX
      IF (ICASX.LT.3) GO TO 7
      DO 6 J=1,ND
      DO 6 I=1,IT
6     CB(I,J)=(0.,0.)
7     I1=ISV+1
      I2=ISV+IT
      IN2=I2
      IF (IN2.GT.N1) IN2=N1
      IM1=I1-N1
      IM2=I2-N1
      IF (IM1.LT.1) IM1=1
      IMX=1
      IF (I1.LE.N1) IMX=N1-I1+2
      IF (N2.GT.N) GO TO 12
C     FILL B(WW),B(WS).  FOR ICASX=1,2 FILL D(WW),D(WS)
      DO 11 J=N2,N
      CALL TRIO (J)
      DO 9 I=1,JSNO
      JSS=JCO(I)
      IF (JSS.LT.N2) GO TO 8
```

```
C     SET JCO WHEN SOURCE IS NEW BASIS FUNCTION ON NEW SEGMENT
      JCO(I)=JSS-N1
      GO TO 9
C     SOURCE IS PORTION OF MODIFIED BASIS FUNCTION ON NEW SEGMENT
8     JCO(I)=NEQS+ICONX(JSS)
9     CONTINUE
      IF (I1.LE.IN2) CALL CMWW (J,I1,IN2,CB,NB,CB,NB,0)
      IF (IM1.LE.IM2) CALL CMWS (J,IM1,IM2,CB(IMX,1),NB,CB,NB,0)
      IF (ICASX.GT.2) GO TO 11
      CALL CMWW (J,N2,N,CD,ND,CD,ND,1)
      IF (M2.LE.M) CALL CMWS (J,M2EQ,MEQ,CD(1,IST),ND,CD,ND,1)
C     LOADING IN D(WW)
      IF (NLOAD.EQ.0) GO TO 11
      IR=J-N1
      EXK=ZARRAY(J)
      DO 10 I=1,JSNO
      JSS=JCO(I)
10    CD(JSS,IR)=CD(JSS,IR)-(AX(I)+CX(I))*EXK
11    CONTINUE
12    IF (NSCON.EQ.0) GO TO 20
C     FILL B(WW)PRIME
      DO 19 I=1,NSCON
      J=ISCON(I)
C     SOURCES ARE NEW OR MODIFIED BASIS FUNCTIONS ON OLD SEGMENTS WHICH
C     CONNECT TO NEW SEGMENTS
      CALL TRIO (J)
      JSS=0
      DO 15 IX=1,JSNO
      IR=JCO(IX)
      IF (IR.LT.N2) GO TO 13
      IR=IR-N1
      GO TO 14
13    IR=ICONX(IR)
      IF (IR.EQ.0) GO TO 15
      IR=NEQS+IR
14    JSS=JSS+1
      JCO(JSS)=IR
      AX(JSS)=AX(IX)
      BX(JSS)=BX(IX)
      CX(JSS)=CX(IX)
15    CONTINUE
      JSNO=JSS
      IF (I1.LE.IN2) CALL CMWW (J,I1,IN2,CB,NB,CB,NB,0)
      IF (IM1.LE.IM2) CALL CMWS (J,IM1,IM2,CB(IMX,1),NB,CB,NB,0)
C     SOURCE IS SINGULAR COMPONENT OF PATCH CURRENT THAT IS PART OF
C     MODIFIED BASIS FUNCTION FOR OLD SEGMENT THAT CONNECTS TO A NEW
C     SEGMENT ON END OPPOSITE PATCH.
      IF (I1.LE.IN2) CALL CMSW (J,I,I1,IN2,CB,CB,0,NB,-1)
      IF (NLODF.EQ.0) GO TO 17
      JX=J-ISV
      IF (JX.LT.1.OR.JX.GT.IT) GO TO 17
      EXK=ZARRAY(J)
      DO 16 IX=1,JSNO
      JSS=JCO(IX)
16    CB(JX,JSS)=CB(JX,JSS)-(AX(IX)+CX(IX))*EXK
C     SOURCES ARE PORTIONS OF MODIFIED BASIS FUNCTION J ON OLD SEGMENTS
C     EXCLUDING OLD SEGMENTS THAT DIRECTLY CONNECT TO NEW SEGMENTS.
17    CALL TBF (J,1)
      JSX=JSNO
      JSNO=1
      IR=JCO(1)
      JCO(1)=NEQS+I
      DO 19 IX=1,JSX
      IF (IX.EQ.1) GO TO 18
```

```
      IR=JCO(IX)
      AX(1)=AX(IX)
      BX(1)=BX(IX)
      CX(1)=CX(IX)
   18 IF (IR.GT.N1) GO TO 19
      IF (ICONX(IR).NE.0) GO TO 19
      IF (I1.LE.IN2) CALL CMWW (IR,I1,IN2,CB,NB,CB,NB,0)
      IF (IM1.LE.IM2) CALL CMWS (IR,IM1,IM2,CB(IMX,1),NB,CB,NB,0)
C     LOADING FOR B(WW)PRIME
      IF (NLODF.EQ.0) GO TO 19
      JX=IR-ISV
      IF (JX.LT.1.OR.JX.GT.IT) GO TO 19
      EXK=ZARRAY(IR)
      JSS=JCO(1)
      CB(JX,JSS)=CB(JX,JSS)-(AX(1)+CX(1))*EXK
   19 CONTINUE
   20 IF (NPCON.EQ.0) GO TO 22
      JSS=NEQP
C     FILL B(SS)PRIME TO SET OLD PATCH BASIS FUNCTIONS TO ZERO FOR
C     PATCHES THAT CONNECT TO NEW SEGMENTS
      DO 21 I=1,NPCON
      IX=IPCON(I)*2+N1-ISV
      IR=IX-1
      JSS=JSS+1
      IF (IR.GT.0.AND.IR.LE.IT) CB(IR,JSS)=(1.,0.)
      JSS=JSS+1
      IF (IX.GT.0.AND.IX.LE.IT) CB(IX,JSS)=(1.,0.)
   21 CONTINUE
   22 IF (M2.GT.M) GO TO 23
C     FILL B(SW) AND B(SS)
      IF (I1.LE.IN2) CALL CMSW (M2,M,I1,IN2,CB(1,IST),CB,N1,NB,0)
      IF (IM1.LE.IM2) CALL CMSS (M2,M,IM1,IM2,CB(IMX,IST),NB,0)
   23 IF (ICASX.EQ.1) GO TO 24
      WRITE (14) ((CB(I,J),I=1,IT),J=1,ND)
   24 CONTINUE
C     FILLING B COMPLETE.  START ON C AND D
      IT=NPBL
      ISV=-NPBL
      DO 43 IBLK=1,NBBL
      ISV=ISV+NPBL
      ISVV=ISV+NC
      IF (IBLK.EQ.NBBL) IT=NLBL
      IF (ICASX.LT.3) GO TO 27
      DO 26 J=1,IT
      DO 25 I=1,NC
   25 CC(I,J)=(0.,0.)
      DO 26 I=1,ND
   26 CD(I,J)=(0.,0.)
   27 I1=ISVV+1
      I2=ISVV+IT
      IN1=I1-M1EQ
      IN2=I2-M1EQ
      IF (IN2.GT.N) IN2=N
      IM1=I1-N
      IM2=I2-N
      IF (IM1.LT.M2EQ) IM1=M2EQ
      IF (IM2.GT.MEQ) IM2=MEQ
      IMX=1
      IF (IN1.LE.IN2) IMX=NEQN-I1+2
      IF (ICASX.LT.3) GO TO 32
      IF (N2.GT.N) GO TO 32
C     SAME AS DO 24 LOOP TO FILL D(WW) FOR ICASX GREATER THAN 2
      DO 31 J=N2,N
      CALL TRIO (J)
```

```
      DO 29 I=1,JSNO
      JSS=JCO(I)
      IF (JSS.LT.N2) GO TO 28
      JCO(I)=JSS-N1
      GO TO 29
   28 JCO(I)=NEQS+ICONX(JSS)
   29 CONTINUE
      IF (IN1.LE.IN2) CALL CMWW (J,IN1,IN2,CD,ND,CD,ND,1)
      IF (IM1.LE.IM2) CALL CMWS (J,IM1,IM2,CD(1,IMX),ND,CD,ND,1)
      IF (NLOAD.EQ.0) GO TO 31
      IR=J-N1-ISV
      IF (IR.LT.1.OR.IR.GT.IT) GO TO 31
      EXK=ZARRAY(J)
      DO 30 I=1,JSNO
      JSS=JCO(I)
   30 CD(JSS,IR)=CD(JSS,IR)-(AX(I)+CX(I))*EXK
   31 CONTINUE
   32 IF (M2.GT.M) GO TO 33
C     FILL D(SW) AND D(SS)
      IF (IN1.LE.IN2) CALL CMSW (M2,M,IN1,IN2,CD(IST,1),CD,N1,ND,1)
      IF (IM1.LE.IM2) CALL CMSS (M2,M,IM1,IM2,CD(IST,IMX),ND,1)
   33 IF (N1.LT.1) GO TO 39
C     FILL C(WW),C(WS), D(WW)PRIME, AND D(WS)PRIME.
      DO 37 J=1,N1
      CALL TRIO (J)
      IF (NSCON.EQ.0) GO TO 36
      DO 35 IX=1,JSNO
      JSS=JCO(IX)
      IF (JSS.LT.N2) GO TO 34
      JCO(IX)=JSS+M1EQ
      GO TO 35
   34 IR=ICONX(JSS)
      IF (IR.NE.0) JCO(IX)=NEQSP+IR
   35 CONTINUE
   36 IF (IN1.LE.IN2) CALL CMWW (J,IN1,IN2,CC,NC,CD,ND,ITX)
      IF (IM1.LE.IM2) CALL CMWS (J,IM1,IM2,CC(1,IMX),NC,CD(1,IMX),ND,ITX
     1)
   37 CONTINUE
      IF (NSCON.EQ.0) GO TO 39
C     FILL C(WW)PRIME
      DO 38 IX=1,NSCON
      IR=ISCON(IX)
      JSS=NEQS+IX-ISV
      IF (JSS.GT.0.AND.JSS.LE.IT) CC(IR,JSS)=(1.,0.)
   38 CONTINUE
   39 IF (NPCON.EQ.0) GO TO 41
      JSS=NEQP-ISV
C     FILL C(SS)PRIME
      DO 40 I=1,NPCON
      IX=IPCON(I)*2+N1
      IR=IX-1
      JSS=JSS+1
      IF (JSS.GT.0.AND.JSS.LE.IT) CC(IR,JSS)=(1.,0.)
      JSS=JSS+1
      IF (JSS.GT.0.AND.JSS.LE.IT) CC(IX,JSS)=(1.,0.)
   40 CONTINUE
   41 IF (M1.LT.1) GO TO 42
C     FILL C(SW) AND C(SS)
      IF (IN1.LE.IN2) CALL CMSW (1,M1,IN1,IN2,CC(N2,1),CC,0,NC,1)
      IF (IM1.LE.IM2) CALL CMSS (1,M1,IM1,IM2,CC(N2,IMX),NC,1)
   42 CONTINUE
      IF (ICASX.EQ.1) GO TO 43
      WRITE (12) ((CD(J,I),J=1,ND),I=1,IT)
      WRITE (15) ((CC(J,I),J=1,NC),I=1,IT)
```

```
   43 CONTINUE
      IF(ICASX.EQ.1)RETURN
      REWIND 12
      REWIND 14
      REWIND 15
      RETURN
      END
```

PURPOSE

To control the filling of the interaction matrix.

METHOD

The linear equations resulting from the moment method solution of equations 13, 14 and the negative of equation 15 in Part I are written as

$$\sum_{j=1}^{N} a_j A_{ij} + \sum_{j=1}^{2M} b_j B_{ij} = E_i, \qquad i = 1, \ldots N$$

$$\sum_{j=1}^{N} c_j C_{kj} + \sum_{j=1}^{2M} d_j D_{kj} = H_k, \qquad k = 1, \ldots 2M$$

where N = number of segments
M = number of patches

$A_{ij} = \hat{s}_i \cdot (\bar{E}$ at $\bar{r}_i$ due to segment basis function j)

$B_{ij} = \hat{s}_i \cdot (\bar{E}$ at $\bar{r}_i$ due to current on patch $[(j + 1)/2]$ in direction $\hat{u}_j$)

$C_{kj} = -\hat{v}_k \cdot (\bar{H}$ at $\bar{p}_{[(k+1)/2]}$ due to segment basis function j) $\cdot\ S_{[(k+1)/2]}$

$D_{kj} = -\hat{v}_k \cdot (\bar{H}$ at $\bar{p}_{[(k+1)/2]}$ due to current on patch $[(j+1)/2]$ in direction $\hat{u}_j$) $S_{[(k+1)/2]} + \frac{1}{2}\sigma_{kj}$

$E_i = -\hat{s}_i \cdot$ (incident electric field at $\bar{r}_i$)

$H_k = \hat{v}_k \cdot$ (incident magnetic field at $\bar{p}_{[(k+1)/2]}$) $S_{[(k+1)/2]}$

$\bar{r}_i$ = position of the center of segment i

$\bar{p}_i$ = position of the center of patch i

$\hat{s}_i$ = unit vector in the direction of segment i

$$\hat{u}_i = \left\{\begin{array}{l}\hat{t}_1 \text{ if i is odd}\\ \\ \hat{t}_2 \text{ if i is even}\end{array}\right\} \text{for patch } [(i+1)/2]$$

$$\hat{v}_i = \left\{\begin{array}{l}\hat{t}_2 \text{ if i is odd}\\ \\ \hat{t}_1 \text{ if i is even}\end{array}\right\} \text{for patch } [(i+1)/2]$$

S_i = 1 if $\hat{t}_1 \times \hat{t}_2 = \hat{n}$ on patch i

-1 if $\hat{t}_1 \times \hat{t}_2 = -\hat{n}$ on patch i

σ_{kj} = -1 if k = j = odd

+1 if k = j = even

0 if k ≠ j

The basis function amplitudes a_j, b_j, c_j and d_j are determined later by solving the matrix equation of order N + 2M.

The matrix elements are computed by calling subroutines CMWW, CMSW, CMWS, and CMSS for the elements of A, B, C and D respectively. For A and C the components of all basis functions that extend across segment J are computed by calling TRIO at CM 52. CMWW and CMWS are then called to compute the components of A or C due to these basis function components on segment J.

If segment j, with length Δ_j, is loaded with impedance Z_j the elements of A are modified as $A_{jk} = A_{jk} - \frac{Z_j}{\Delta j}$ X (value of basis function k at the center of segment j) for k = the numbers of all basis functions that extend onto segment j. The summation over values of k (k = JSS) for loading on segment J occurs at CM 68.

The submatrices are stored in the array CM in transposed form. All references to rows and columns, here, apply to the nontransposed matrices. Thus "row" in this discussion refers to the second index of CM in the code.

For a structure without symmetry the submatrices are stored in the order

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix}$$

If the complete matrix is too large for the array CM then blocks of rows are filled and written on file 11. A block may then contain rows from A and B, rows from C and D or a combination. The row of CM at which C and D start is computed as IST.

For a structure having p symmetric sections the submatrices are stored in the form

$$\begin{bmatrix} A_1 & B_1 & A_2 & B_2 & & A_p & B_p \\ & & & & \cdots & & \\ C_1 & D_1 & C_2 & D_2 & & C_p & D_p \end{bmatrix}$$

where $\begin{bmatrix} A_i & B_i \\ C_i & D_i \end{bmatrix}$

represents A_i in the first row of submatrices in equation 108 of Part I. Each call to CMWW and CMWS may fill elements of A_i or C_i for any value of i. The column indices in array JCO are adjusted at CM 55 to allow for the columns occupied by the B_i and D_i matrices. B_i and D_i are filled for each value of i in the loop from CM 75 to CM 81. The Fourier transform of the submatrices, or the transform for planar symmetry (equation 116 of Part I) is computed from CM 85 to CM 100.

SYMBOL DICTIONARY

- CM = array for the matrix
- I1 = number of first equation in a block (patch equation +N for patches)
- I2 = number of the last equation in a block
- IEXKX = 1 to use extended thin wire kernel on wires, 0 otherwise
- IM1 = number of first patch equation in a block

IM2	=	number of last patch equation in a block
IN2	=	number of the last segment equation in a block
IOUT	=	number of real numbers in a block for output
IPR	=	row in CM (second index) for segment J
IST	=	row in CM of the first patch equation
ISV	=	I1 - 1
IT	=	number of rows in a block
IXBLK1	=	block number
JM1	=	number of first patch in a symmetric section
JM2	=	number of the last patch in a symmetric section
JST	=	column in CM of the first patch equation for a symmetric block
MP2	=	number of patch equations
NEQ	=	total number of equations
NOP	=	number of symmetric sections
NPEQ	=	number of equations in a symmetric section
NROW	=	row dimensions of the transposed CM array
RKHX	=	minimum interaction distance at which the infinitesimal dipole approximation is used for the field of a segment
ZAJ	=	Z_j/Δ_j

```
      SUBROUTINE CMSET (NROW,CM,RKHX,IEXKX)
C
C     CMSET SETS UP THE COMPLEX STRUCTURE MATRIX IN THE ARRAY CM
C
      COMPLEX CM,ZARRAY,ZAJ,ETK,ETS,ETC,EXK,EYK,EZK,EXS,EYS,EZS,EXC,EYC,
     1EZC,SSX,D,DETER
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /MATPAR/ ICASE,NBLOKS,NPBLK,NLAST,NBLSYM,NPSYM,NLSYM,IMAT,I
     1CASX,NBBX,NPBX,NLBX,NBBL,NPBL,NLBL
      COMMON /SMAT/ SSX(16,16)
      COMMON /SCRATM/ D(600)
      COMMON /ZLOAD/ ZARRAY(300),NLOAD,NLODF
      COMMON /SEGJ/ AX(30),BX(30),CX(30),JCO(30),JSNO,ISCON(50),NSCON,IP
     1CON(10),NPCON
      COMMON /DATAJ/ S,B,XJ,YJ,ZJ,CABJ,SABJ,SALPJ,EXK,EYK,EZK,EXS,EYS,EZ
     1S,EXC,EYC,EZC,RKH,IEXK,IND1,IND2,IPGND
      DIMENSION CM(NROW,1)
      MP2=2*MP
      NPEQ=NP+MP2
      NEQ=N+2*M
      NOP=NEQ/NPEQ
      IF (ICASE.GT.2) REWIND 11
      RKH=RKHX
      IEXK=IEXKX
      IOUT=2*NPBLK*NROW
      IT=NPBLK
C
C     CYCLE OVER MATRIX BLOCKS
C
      DO 13 IXBLK1=1,NBLOKS
      ISV=(IXBLK1-1)*NPBLK
      IF (IXBLK1.EQ.NBLOKS) IT=NLAST
      DO 1 I=1,NROW
      DO 1 J=1,IT
1     CM(I,J)=(0.,0.)
      I1=ISV+1
      I2=ISV+IT
      IN2=I2
      IF (IN2.GT.NP) IN2=NP
      IM1=I1-NP
      IM2=I2-NP
      IF (IM1.LT.1) IM1=1
      IST=1
      IF (I1.LE.NP) IST=NP-I1+2
      IF (N.EQ.0) GO TO 5
C
C     WIRE SOURCE LOOP
C
      DO 4 J=1,N
      CALL TRIO (J)
      DO 2 I=1,JSNO
      IJ=JCO(I)
2     JCO(I)=((IJ-1)/NP)*MP2+IJ
      IF (I1.LE.IN2) CALL CMWW (J,I1,IN2,CM,NROW,CM,NROW,1)
      IF (IM1.LE.IM2) CALL CMWS (J,IM1,IM2,CM(1,IST),NROW,CM,NROW,1)
      IF (NLOAD.EQ.0) GO TO 4
C
C     MATRIX ELEMENTS MODIFIED BY LOADING
C
      IF (J.GT.NP) GO TO 4
      IPR=J-ISV
      IF (IPR.LT.1.OR.IPR.GT.IT) GO TO 4
```

```
      ZAJ=ZARRAY(J)
      DO 3 I=1,JSNO
      JSS=JCO(I)
3     CM(JSS,IPR)=CM(JSS,IPR)-(AX(I)+CX(I))*ZAJ
4     CONTINUE
5     IF (M.EQ.0) GO TO 7
C     MATRIX ELEMENTS FOR PATCH CURRENT SOURCES
      JM1=1-MP
      JM2=0
      JST=1-MP2
      DO 6 I=1,NOP
      JM1=JM1+MP
      JM2=JM2+MP
      JST=JST+NPEQ
      IF (I1.LE.IN2) CALL CMSW (JM1,JM2,I1,IN2,CM(JST,1),CM,0,NROW,1)
      IF (IM1.LE.IM2) CALL CMSS (JM1,JM2,IM1,IM2,CM(JST,IST),NROW,1)
6     CONTINUE
7     IF (ICASE.EQ.1) GO TO 13
      IF (ICASE.EQ.3) GO TO 12
C     COMBINE ELEMENTS FOR SYMMETRY MODES
      DO 11 I=1,IT
      DO 11 J=1,NPEQ
      DO 8 K=1,NOP
      KA=J+(K-1)*NPEQ
8     D(K)=CM(KA,I)
      DETER=D(1)
      DO 9 KK=2,NOP
9     DETER=DETER+D(KK)
      CM(J,I)=DETER
      DO 11 K=2,NOP
      KA=J+(K-1)*NPEQ
      DETER=D(1)
      DO 10 KK=2,NOP
10    DETER=DETER+D(KK)*SSX(K,KK)
      CM(KA,I)=DETER
11    CONTINUE
      IF (ICASE.LT.3) GO TO 13
C     WRITE BLOCK FOR OUT-OF-CORE CASES.
12    CALL BLCKOT (CM,11,1,IOUT,1,31)
13    CONTINUE
      IF (ICASE.GT.2) REWIND 11
      RETURN
      END
```

CMSS

PURPOSE

To compute and store matrix elements representing the H field at patch centers due to the current on patches.

METHOD

CMSS computes the matrix elements D_{kj} defined in the description of subroutine CMSET. Subroutine HINTG is called to compute the magnetic field at the center of patch I due to current on patch J. H due to the current $\hat{t}_1$ on patch J is stored in EXK, EYK and EZK, while H due to current $\hat{t}_2$ is stored in EXS, EYS and EZS. The term 0.5 σ_{kj} in D_{kj} is added at CM 61 and CM 62 for odd and even equations. The matrix elements are stored in array CM from SS63 to SS78 in either normal or transposed order. Elements for both the even and odd equations are stored if both equations are within the block.

SYMBOL DICTIONARY

CM	= array for matrix storage
G11	= D_{kj} for k odd, j odd
G12	= D_{kj} for k odd, j even
G21	= D_{kj} for k even, j odd
G22	= D_{kj} for k even, j even
I1	= patch number for first equation
I2	= patch number for last equation
ICOMP	= equation number for the odd numbered equation for observation patch I
II1	= location of the odd numbered equation in CM
II2	= location of the even numbered equation in CM
IL	= array location for coordinates of patch I
IM1	= patch equation number for first equation in block
IM2	= patch equation number for last equation in block
ITRP	= 0 or 1 to select normal or transposed filling of CM
J1	= number of first source patch
J2	= number of last source patch

JJ1	= column in non-transposed matrix, of the first equation for patch J
JJ2	= column of second equation for patch J
JL	= array location for coordinates of patch J
NROW	= row dimension of CM
T1XI, T1YI, T1ZI T2XI, T2YI, T2ZI T1XJ, T1YJ, T1ZJ T2XJ, T2YJ, T2ZJ	= x, y and z components of $\hat{t}_1$ or $\hat{t}_2$ for patch I or J
XI, YI, ZI	= coordinates of center of patch I

```
      SUBROUTINE CMSS (J1,J2,IM1,IM2,CM,NROW,ITRP)
C     CMSS COMPUTES MATRIX ELEMENTS FOR SURFACE-SURFACE INTERACTIONS.
      COMPLEX G11,G12,G21,G22,CM,EXK,EYK,EZK,EXS,EYS,EZS,EXC,EYC,EZC
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /ANGL/ SALP(300)
      COMMON /DATAJ/ S,B,XJ,YJ,ZJ,CABJ,SABJ,SALPJ,EXK,EYK,EZK,EXS,EYS,EZ
     1S,EXC,EYC,EZC,RKH,IEXK,IND1,IND2,IPGND
      DIMENSION CM(NROW,1)
      DIMENSION T1X(1), T1Y(1), T1Z(1), T2X(1), T2Y(1), T2Z(1)
      EQUIVALENCE (T1X,SI), (T1Y,ALP), (T1Z,BET), (T2X,ICON1), (T2Y,ICON
     12), (T2Z,ITAG)
      EQUIVALENCE (T1XJ,CABJ), (T1YJ,SABJ), (T1ZJ,SALPJ), (T2XJ,B), (T2Y
     1J,IND1), (T2ZJ,IND2)
      LDP=LD+1
      I1=(IM1+1)/2
      I2=(IM2+1)/2
      ICOMP=I1*2-3
      II1=-1
      IF (ICOMP+2.LT.IM1) II1=-2
C     LOOP OVER OBSERVATION PATCHES
      DO 5 I=I1,I2
      IL=LDP-I
      ICOMP=ICOMP+2
      II1=II1+2
      II2=II1+1
      T1XI=T1X(IL)*SALP(IL)
      T1YI=T1Y(IL)*SALP(IL)
      T1ZI=T1Z(IL)*SALP(IL)
      T2XI=T2X(IL)*SALP(IL)
      T2YI=T2Y(IL)*SALP(IL)
      T2ZI=T2Z(IL)*SALP(IL)
      XI=X(IL)
      YI=Y(IL)
      ZI=Z(IL)
      JJ1=-1
C     LOOP OVER SOURCE PATCHES
      DO 5 J=J1,J2
      JL=LDP-J
      JJ1=JJ1+2
      JJ2=JJ1+1
      S=BI(JL)
      XJ=X(JL)
      YJ=Y(JL)
      ZJ=Z(JL)
      T1XJ=T1X(JL)
      T1YJ=T1Y(JL)
      T1ZJ=T1Z(JL)
      T2XJ=T2X(JL)
      T2YJ=T2Y(JL)
      T2ZJ=T2Z(JL)
      CALL HINTG (XI,YI,ZI)
      G11=-(T2XI*EXK+T2YI*EYK+T2ZI*EZK)
      G12=-(T2XI*EXS+T2YI*EYS+T2ZI*EZS)
      G21=-(T1XI*EXK+T1YI*EYK+T1ZI*EZK)
      G22=-(T1XI*EXS+T1YI*EYS+T1ZI*EZS)
      IF (I.NE.J) GO TO 1
      G11=G11-.5
      G22=G22+.5
    1 IF (ITRP.NE.0) GO TO 3
C     NORMAL FILL
      IF (ICOMP.LT.IM1) GO TO 2
      CM(II1,JJ1)=G11
```

```
      CM(II1,JJ2)=G12
    2 IF (ICOMP.GE.IM2) GO TO 5
      CM(II2,JJ1)=G21
      CM(II2,JJ2)=G22
      GO TO 5
C     TRANSPOSED FILL
    3 IF (ICOMP.LT.IM1) GO TO 4
      CM(JJ1,II1)=G11
      CM(JJ2,II1)=G12
    4 IF (ICOMP.GE.IM2) GO TO 5
      CM(JJ1,II2)=G21
      CM(JJ2,II2)=G22
    5 CONTINUE
      RETURN
      END
```

CMSW

PURPOSE

To compute and store matrix elements representing the electric field at segment centers due to the current on patches.

METHOD

SW30 - SW35	Coordinates of observation segment are stored.
SW36 - SW42	If either end of the observation segment connects to a surface IPCH is set to the number of the first of the four patches at the connection point.
SW48 - SW57	Coordinates of the source patch are stored in COMMON/DATAJ/.
SW61 - SW86	IF IPCH = J then patch J is the first patch at the point where segment I connects to the surface. Subroutine PCINT is called to integrate the current over the four patches at the connection point. The current on the patches includes the eight basis functions of the four patches and a portion of the basis function from the segment. Hence contributions to nine matrix elements are generated and stored in array EMEL. The field due to the segment basis function extending onto the patches is stored in array CW at SW76 or SW78. The fields due to the first patch basis function, EMEL(1) and EMEL(5), are then stored in array CM at SW80 and SW81 or at SW83 and SW84. ICGO is then incremented. For the next three times through the loop over J the call to PCINT is skipped at SW63 and the remaining values in EMEL are stored.
SW88 - SW96	If segment I and patch J are not connected, subroutine UNERE is called to compute the electric field due to the current on the patch with the current treated as Hertzian dipoles in the directions $\hat{t}_1$ and $\hat{t}_2$. The matrix elements are stored in CM.

SW102 - SW138 This is a special section of code to compute the electric field due to the component of a segment basis function that extends onto connected patches. It is used at line CG112 of subroutine CMNGF for the case where the connected segment and patches are in the NGF file and a new segment is connected to the outer end of the NGF segment modifying its basis function. Subroutine PCINT is called to evaluate the nine matrix elements. Only EMEL(9) is used since the patch basis functions have not been modified.

SYMBOL DICTIONARY

CABI	= x component of $\hat{i}$ in direction of segment I
CM	= array for E due to patch basis functions
CW	= array for E due to segment basis function extending onto surface at connection point
EMEL	= array of matrix elements from integrating over surface
FSIGN	= ± 1 depending on which end of segment connects to surface
I1	= number of first observation segment
I2	= number of last observation segment
ICGØ	= index for matrix elements at connection point
IL	= index for segment basis function in CW
IP	= 1 for direct field, 2 for image in ground
IPCH	= number of first patch connecting to a segment
ITRP	= 0 for normal matrix fill 1 for transposed fill -1 for special NGF case
J	= source patch
J1	= first source patch
J2	= last source patch
JL	= index for source patch in CM
JS	= index for patch coordinates
K	= index in CM or CW for observation segment
NCW	= index offset for CW

NEQS	= number of equations excluding NGF
NROW	= row dimensions of CM and CW
PI	= pi
PX	= $\sin k(s - s_0)$ } for s at the end of the segment
PY	= $\cos k(s - s_0)$ } connected to the surface
SABI	= y component of $\hat{i}$ in direction of segment I
SALPI	= z component of $\hat{i}$ in direction of segment I
XI, YI, ZI	= center of observation segment

CMSW

```
      SUBROUTINE CMSW (J1,J2,I1,I2,CM,CW,NCW,NROW,ITRP)
C     COMPUTES MATRIX ELEMENTS FOR E ALONG WIRES DUE TO PATCH CURRENT   SW   1
      COMPLEX CM,ZRATI,ZRATI2,T1,EXK,EYK,EZK,EXS,EYS,EZS,EXC,EYC,EZC,EME SW   2
     1L,CW,FRATI                                                        SW   3
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300 SW   4
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX( SW   5
     2300),WLAM,IPSYM                                                   SW   6
      COMMON /ANGL/ SALP(300)                                           SW   7
      COMMON /GND/ZRATI,ZRATI2,FRATI,CL,CH,SCRWL,SCRWR,NRADL,KSYMP,IFAR, SW   8
     1IPERF,T1,T2                                                       SW   9
      COMMON /DATAJ/ S,B,XJ,YJ,ZJ,CABJ,SABJ,SALPJ,EXK,EYK,EZK,EXS,EYS,EZ SW  10
     1S,EXC,EYC,EZC,RKH,IEXK,IND1,IND2,IPGND                            SW  11
      COMMON /SEGJ/ AX(30),BX(30),CX(30),JCO(30),JSNO,ISCON(50),NSCON,IP SW  12
     1CON(10),NPCON                                                     SW  13
      DIMENSION CAB(1), SAB(1), CM(NROW,1), CW(NROW,1)                  SW  14
      DIMENSION T1X(1), T1Y(1), T1Z(1), T2X(1), T2Y(1), T2Z(1), EMEL(9) SW  15
      EQUIVALENCE (T1X,SI), (T1Y,ALP), (T1Z,BET), (T2X,ICON1), (T2Y,ICON SW  16
     12), (T2Z,ITAG), (CAB,ALP), (SAB,BET)                              SW  17
      EQUIVALENCE (T1XJ,CABJ), (T1YJ,SABJ), (T1ZJ,SALPJ), (T2XJ,B), (T2Y SW  18
     1J,IND1), (T2ZJ,IND2)                                              SW  19
      DATA PI/3.141592654/                                              SW  20
      LDP=LD+1                                                          SW  21
      NEQS=N-N1+2*(M-M1)                                                SW  22
      IF (ITRP.LT.0) GO TO 13                                           SW  23
      K=0                                                               SW  24
      ICGO=1                                                            SW  25
C     OBSERVATION LOOP                                                  SW  26
      DO 12 I=I1,I2                                                     SW  27
      K=K+1                                                             SW  28
      XI=X(I)                                                           SW  29
      YI=Y(I)                                                           SW  30
      ZI=Z(I)                                                           SW  31
      CABI=CAB(I)                                                       SW  32
      SABI=SAB(I)                                                       SW  33
      SALPI=SALP(I)                                                     SW  34
      IPCH=0                                                            SW  35
      IF (ICON1(I).LT.10000) GO TO 1                                    SW  36
      IPCH=ICON1(I)-10000                                               SW  37
      FSIGN=-1.                                                         SW  38
1     IF (ICON2(I).LT.10000) GO TO 2                                    SW  39
      IPCH=ICON2(I)-10000                                               SW  40
      FSIGN=1.                                                          SW  41
2     JL=0                                                              SW  42
C     SOURCE LOOP                                                       SW  43
      DO 12 J=J1,J2                                                     SW  44
      JS=LDP-J                                                          SW  45
      JL=JL+2                                                           SW  46
      T1XJ=T1X(JS)                                                      SW  47
      T1YJ=T1Y(JS)                                                      SW  48
      T1ZJ=T1Z(JS)                                                      SW  49
      T2XJ=T2X(JS)                                                      SW  50
      T2YJ=T2Y(JS)                                                      SW  51
      T2ZJ=T2Z(JS)                                                      SW  52
      XJ=X(JS)                                                          SW  53
      YJ=Y(JS)                                                          SW  54
      ZJ=Z(JS)                                                          SW  55
      S=BI(JS)                                                          SW  56
C     GROUND LOOP                                                       SW  57
      DO 12 IP=1,KSYMP                                                  SW  58
      IPGND=IP                                                          SW  59
      IF (IPCH.NE.J.AND.ICGO.EQ.1) GO TO 9                              SW  60
      IF (IP.EQ.2) GO TO 9                                              SW  61
      IF (ICGO.GT.1) GO TO 6                                            SW  62
      CALL PCINT (XI,YI,ZI,CABI,SABI,SALPI,EMEL)                        SW  63
                                                                        SW  64
```

```
      PY=PI*SI(I)*FSIGN                                                 SW  65
      PX=SIN(PY)                                                        SW  66
      PY=COS(PY)                                                        SW  67
      EXC=EMEL(9)*FSIGN                                                 SW  68
      CALL TRIO (I)                                                     SW  69
      IF (I.GT.N1) GO TO 3                                              SW  70
      IL=NEQS+ICONX(I)                                                  SW  71
      GO TO 4                                                           SW  72
3     IL=I-NCW                                                          SW  73
      IF (I.LE.NP) IL=((IL-1)/NP)*2*MP+IL                               SW  74
4     IF (ITRP.NE.0) GO TO 5                                            SW  75
      CW(K,IL)=CW(K,IL)+EXC*(AX(JSNO)+BX(JSNO)*PX+CX(JSNO)*PY)          SW  76
      GO TO 6                                                           SW  77
5     CW(IL,K)=CW(IL,K)+EXC*(AX(JSNO)+BX(JSNO)*PX+CX(JSNO)*PY)          SW  78
6     IF (ITRP.NE.0) GO TO 7                                            SW  79
      CM(K,JL-1)=EMEL(ICGO)                                             SW  80
      CM(K,JL)=EMEL(ICGO+4)                                             SW  81
      GO TO 8                                                           SW  82
7     CM(JL-1,K)=EMEL(ICGO)                                             SW  83
      CM(JL,K)=EMEL(ICGO+4)                                             SW  84
8     ICGO=ICGO+1                                                       SW  85
      IF (ICGO.EQ.5) ICGO=1                                             SW  86
      GO TO 11                                                          SW  87
9     CALL UNERE (XI,YI,ZI)                                             SW  88
      IF (ITRP.NE.0) GO TO 10                                           SW  89
C     NORMAL FILL                                                       SW  90
      CM(K,JL-1)=CM(K,JL-1)+EXK*CABI+EYK*SABI+EZK*SALPI                 SW  91
      CM(K,JL)=CM(K,JL)+EXS*CABI+EYS*SABI+EZS*SALPI                     SW  92
      GO TO 11                                                          SW  93
C     TRANSPOSED FILL                                                   SW  94
10    CM(JL-1,K)=CM(JL-1,K)+EXK*CABI+EYK*SABI+EZK*SALPI                 SW  95
      CM(JL,K)=CM(JL,K)+EXS*CABI+EYS*SABI+EZS*SALPI                     SW  96
11    CONTINUE                                                          SW  97
12    CONTINUE                                                          SW  98
      RETURN                                                            SW  99
C     FOR OLD SEG. CONNECTING TO OLD PATCH ON ONE END AND NEW SEG. ON   SW 100
C     OTHER END INTEGRATE SINGULAR COMPONENT (9) OF SURFACE CURRENT ONLY SW 101
13    IF (J1.LT.I1.OR.J1.GT.I2) GO TO 16                                SW 102
      IPCH=ICON1(J1)                                                    SW 103
      IF (IPCH.LT.10000) GO TO 14                                       SW 104
      IPCH=IPCH-10000                                                   SW 105
      FSIGN=-1.                                                         SW 106
      GO TO 15                                                          SW 107
14    IPCH=ICON2(J1)                                                    SW 108
      IF (IPCH.LT.10000) GO TO 16                                       SW 109
      IPCH=IPCH-10000                                                   SW 110
      FSIGN=1.                                                          SW 111
15    IF (IPCH.GT.M1) GO TO 16                                          SW 112
      JS=LDP-IPCH                                                       SW 113
      IPGND=1                                                           SW 114
      T1XJ=T1X(JS)                                                      SW 115
      T1YJ=T1Y(JS)                                                      SW 116
      T1ZJ=T1Z(JS)                                                      SW 117
      T2XJ=T2X(JS)                                                      SW 118
      T2YJ=T2Y(JS)                                                      SW 119
      T2ZJ=T2Z(JS)                                                      SW 120
      XJ=X(JS)                                                          SW 121
      YJ=Y(JS)                                                          SW 122
      ZJ=Z(JS)                                                          SW 123
      S=BI(JS)                                                          SW 124
      XI=X(J1)                                                          SW 125
      YI=Y(J1)                                                          SW 126
      ZI=Z(J1)                                                          SW 127
      CABI=CAB(J1)                                                      SW 128
```

```
      SABI=SAB(J1)                                                     SW 129
      SALPI=SALP(J1)                                                   SW 130
      CALL PCINT (XI,YI,ZI,CABI,SABI,SALPI,EMEL)                       SW 131
      PY=PI*SI(J1)*FSIGN                                               SW 132
      PX=SIN(PY)                                                       SW 133
      PY=COS(PY)                                                       SW 134
      EXC=EMEL(9)*FSIGN                                                SW 135
      IL=JCO(JSNO)                                                     SW 136
      K=J1-I1+1                                                        SW 137
      CW(K,IL)=CW(K,IL)+EXC*(AX(JSNO)+BX(JSNO)*PX+CX(JSNO)*PY)         SW 138
   16 RETURN                                                           SW 139
      END                                                              SW 140-
```

CMWS

PURPOSE

To compute and store matrix elements representing the magnetic field at patch centers due to the current on wire segments.

METHOD

Matrix elements are computed for patch equations numbered I1 through I2 with the source segment J. For odd numbered equations the matrix element represents the first term on the right side of equation 14 of Part I. For even numbered equations it is the negative of the first term on the right side of equation 15. For equation I1 and for all odd numbered equations subroutine HSFLD is called to compute the H field at the center of the patch due to constant, sin $k(s - s_0)$ and cos $k(s - s_0)$ currents on segment J. The required component of the field, $-\hat{t}_2 \cdot \bar{H}$ or $-\hat{t}_1 \cdot \bar{H}$ for odd or even equations respectively, is computed from WS49 to WS51. Multiplication by SALP(JS) reverses the sign when $(\hat{t}_1, \hat{t}_2, \hat{n})$ has a left-hand orientation on a patch formed by reflection. The field component for each basis function component on segment J is computed and stored for WS56 through WS75. Storage of the matrix elements is similar to that in subroutine CMWW.

SYMBOL DICTIONARY

CM	= array for matrix elements
CW	= array for matrix elements (NGF only)
ETK, ETS, ETC	= $-\hat{t}_2 \cdot \bar{H}$ or $-\hat{t}_1 \cdot \bar{H}$ due to current of constant, sin $k(s - s_0)$, or cos $k(s - s_0)$ respectively
I	= equation number
I1	= number of first equation
I2	= number of second equation
IK	= 0 if I is even, 1 if I is odd
IPATCH	= patch number for equation I
IPR	= relative matrix location for equation I. Position in complete matrix depends on the address of CM in the call to CMWS

ITRP = 0 for non-transposed fill
1 for transposed fill
2 for transposed fill for NGF

J = source segment number

JS = location in COMMON/DATA/ of parameters for patch J

JX = matrix index for a particular basis function

LDP = LD + 1

NR = row dimension of CM

NW = row dimension of CW

TX, TY, TZ = x, y, and z components of $\hat{t}_1$ or $\hat{t}_2$

XI, YI, ZI = x, y and z coordinates of the center of the patch at which the field is computed

```
      SUBROUTINE CMWS (J,I1,I2,CM,NR,CW,NW,ITRP)
C
C     CMWS COMPUTES MATRIX ELEMENTS FOR WIRE-SURFACE INTERACTIONS
C
      COMPLEX CM,CW,ETK,ETS,ETC,EXK,EYK,EZK,EXS,EYS,EZS,EXC,EYC,EZC
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /ANGL/ SALP(300)
      COMMON /SEGJ/ AX(30),BX(30),CX(30),JCO(30),JSNO,ISCON(50),NSCON,IP
     1CON(10),NPCON
      COMMON /DATAJ/ S,B,XJ,YJ,ZJ,CABJ,SABJ,SALPJ,EXK,EYK,EZK,EXS,EYS,EZ
     1S,EXC,EYC,EZC,RKH,IEXK,IND1,IND2,IPGND
      DIMENSION CM(NR,1), CW(NW,1), CAB(1), SAB(1)
      DIMENSION T1X(1), T1Y(1), T1Z(1), T2X(1), T2Y(1), T2Z(1)
      EQUIVALENCE (CAB,ALP), (SAB,BET), (T1X,SI), (T1Y,ALP), (T1Z,BET)
      EQUIVALENCE (T2X,ICON1), (T2Y,ICON2), (T2Z,ITAG)
      LDP=LD+1
      S=SI(J)
      B=BI(J)
      XJ=X(J)
      YJ=Y(J)
      ZJ=Z(J)
      CABJ=CAB(J)
      SABJ=SAB(J)
      SALPJ=SALP(J)
C
C     OBSERVATION LOOP
C
      IPR=0
      DO 9 I=I1,I2
      IPR=IPR+1
      IPATCH=(I+1)/2
      IK=I-(I/2)*2
      IF (IK.EQ.0.AND.IPR.NE.1) GO TO 1
      JS=LDP-IPATCH
      XI=X(JS)
      YI=Y(JS)
      ZI=Z(JS)
      CALL HSFLD (XI,YI,ZI,0.)
      IF (IK.EQ.0) GO TO 1
      TX=T2X(JS)
      TY=T2Y(JS)
      TZ=T2Z(JS)
      GO TO 2
1     TX=T1X(JS)
      TY=T1Y(JS)
      TZ=T1Z(JS)
2     ETK=-(EXK*TX+EYK*TY+EZK*TZ)*SALP(JS)
      ETS=-(EXS*TX+EYS*TY+EZS*TZ)*SALP(JS)
      ETC=-(EXC*TX+EYC*TY+EZC*TZ)*SALP(JS)
C
C     FILL MATRIX ELEMENTS.  ELEMENT LOCATIONS DETERMINED BY CONNECTION
C     DATA.
C
      IF (ITRP.NE.0) GO TO 4
C     NORMAL FILL
      DO 3 IJ=1,JSNO
      JX=JCO(IJ)
3     CM(IPR,JX)=CM(IPR,JX)+ETK*AX(IJ)+ETS*BX(IJ)+ETC*CX(IJ)
      GO TO 9
4     IF (ITRP.EQ.2) GO TO 6
C     TRANSPOSED FILL
      DO 5 IJ=1,JSNO
```

```
      JX=JCO(IJ)
 5    CM(JX,IPR)=CM(JX,IPR)+ETK*AX(IJ)+ETS*BX(IJ)+ETC*CX(IJ)
      GO TO 9
C     TRANSPOSED FILL - C(WS) AND D(WS)PRIME (=CW)
 6    DO 8 IJ=1,JSNO
      JX=JCO(IJ)
      IF (JX.GT.NR) GO TO 7
      CM(JX,IPR)=CM(JX,IPR)+ETK*AX(IJ)+ETS*BX(IJ)+ETC*CX(IJ)
      GO TO 8
 7    JX=JX-NR
      CW(JX,IPR)=CW(JX,IPR)+ETK*AX(IJ)+ETS*BX(IJ)+ETC*CX(IJ)
 8    CONTINUE
 9    CONTINUE
      RETURN
      END
```

PURPOSE

To call subroutines to compute the electric field at segment centers due to current on other segments and to store matrix elements in array locations.

METHOD

WW17 - WW24	Parameters of source segment (J) are stored in COMMON/DATAJ/.
WW27 - WW43	First end of segment J is tested to determine whether the extended thin wire approximation can be used. It cannot be used at a junction of more than two wires (WW30), at a bend (WW37), at a change in radius (WW38), or at the base of a non-vertical segment connected to the ground (WW33).
WW44 - WW60	Second end of segment J is tested.
WW66	Loop over observation segments ranges from I1 to I2. The index IPR starts at 1 so the matrix element for I1 is stored in the first row or column of the array CM. The location in the complete matrix is determined by the address given for CM when CMWW is called.
WW76	EFLD computes the electric fields at (XI, YI, ZI) due to segment J and stores them in COMMON/DATAJ/.
WW77 - WW79	Electric field tangent to segment I is computed.
WW84 - WW103	Matrix elements are formed by combining the field components.
WW86 - WW88	Matrix elements are stored in non-transposed order.
WW92 - WW94	Matrix elements are stored in transposed order.
WW97 - WW104	When the source segment is from a NGF file the matrix elements will normally be stored in submatrix C of the NGF matrix structure. When the segment connects to a new segment, however, contributions to submatrix D result. The C and D contributions are stored in CM and CW, respectively, in transposed order.

SYMBOL DICTIONARY

AI	=	radius of observation segment
CABI	=	x component of unit vector in direction of segment
CM	=	array for matrix elements
CW	=	array for matrix elements (NGF only)
ETK, ETS, ETC	=	E field tangent to segment I due to current of constant, sin $k(s - s_0)$ and cos $k(s - s_0)$ distribution, respectively, on segment J.
I1	=	first observation segment
I2	=	final observation segment
IJ	=	0 for special treatment when I = J
IPR	=	relative matrix location for observation point
ITRP	=	0 for non-transposed fill 1 for transposed fill 2 for transposed fill for NGF
J	=	source segment number
JX	=	matrix index for a particular basis function
NR	=	row dimension of CM
NW	=	row dimension of CW
SABI	=	y component of unit vector in direction of segment
SALPI	=	z component of unit vector in direction of segment
XI, YI, ZI	=	coordinates of center of segment I.

CONSTANTS

0.999999	=	test for collinear segments

```
      SUBROUTINE CMWW (J,I1,I2,CM,NR,CW,NW,ITRP)
C
C     CMWW COMPUTES MATRIX ELEMENTS FOR WIRE-WIRE INTERACTIONS
C
      COMPLEX CM,CW,ETK,ETS,ETC,EXK,EYK,EZK,EXS,EYS,EZS,EXC,EYC,EZC
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /ANGL/ SALP(300)
      COMMON /SEGJ/ AX(30),BX(30),CX(30),JCO(30),JSNO,ISCON(50),NSCON,IP
     1CON(10),NPCON
      COMMON /DATAJ/ S,B,XJ,YJ,ZJ,CABJ,SABJ,SALPJ,EXK,EYK,EZK,EXS,EYS,EZ
     1S,EXC,EYC,EZC,RKH,IEXK,IND1,IND2,IPGND
      DIMENSION CM(NR,1), CW(NW,1), CAB(1), SAB(1)
      EQUIVALENCE (CAB,ALP), (SAB,BET)
C     SET SOURCE SEGMENT PARAMETERS
      S=SI(J)
      B=BI(J)
      XJ=X(J)
      YJ=Y(J)
      ZJ=Z(J)
      CABJ=CAB(J)
      SABJ=SAB(J)
      SALPJ=SALP(J)
      IF (IEXK.EQ.0) GO TO 16
C     DECIDE WETHER EXT. T.W. APPROX. CAN BE USED
      IPR=ICON1(J)
      IF (IPR) 1,6,2
    1 IPR=-IPR
      IF (-ICON1(IPR).NE.J) GO TO 7
      GO TO 4
    2 IF (IPR.NE.J) GO TO 3
      IF (CABJ*CABJ+SABJ*SABJ.GT.1.E-8) GO TO 7
      GO TO 5
    3 IF (ICON2(IPR).NE.J) GO TO 7
    4 XI=ABS(CABJ*CAB(IPR)+SABJ*SAB(IPR)+SALPJ*SALP(IPR))
      IF (XI.LT.0.999999) GO TO 7
      IF (ABS(BI(IPR)/B-1.).GT.1.E-6) GO TO 7
    5 IND1=0
      GO TO 8
    6 IND1=1
      GO TO 8
    7 IND1=2
    8 IPR=ICON2(J)
      IF (IPR) 9,14,10
    9 IPR=-IPR
      IF (-ICON2(IPR).NE.J) GO TO 15
      GO TO 12
   10 IF (IPR.NE.J) GO TO 11
      IF (CABJ*CABJ+SABJ*SABJ.GT.1.E-8) GO TO 15
      GO TO 13
   11 IF (ICON1(IPR).NE.J) GO TO 15
   12 XI=ABS(CABJ*CAB(IPR)+SABJ*SAB(IPR)+SALPJ*SALP(IPR))
      IF (XI.LT.0.999999) GO TO 15
      IF (ABS(BI(IPR)/B-1.).GT.1.E-6) GO TO 15
   13 IND2=0
      GO TO 16
   14 IND2=1
      GO TO 16
   15 IND2=2
   16 CONTINUE
C
C     OBSERVATION LOOP
C
```

```
      IPR=0
      DO 23 I=I1,I2
      IPR=IPR+1
      IJ=1-J
      XI=X(I)
      YI=Y(I)
      ZI=Z(I)
      AI=BI(I)
      CABI=CAB(I)
      SABI=SAB(I)
      SALPI=SALP(I)
      CALL EFLD (XI,YI,ZI,AI,IJ)
      ETK=EXK*CABI+EYK*SABI+EZK*SALPI
      ETS=EXS*CABI+EYS*SABI+EZS*SALPI
      ETC=EXC*CABI+EYC*SABI+EZC*SALPI
C
C     FILL MATRIX ELEMENTS.  ELEMENT LOCATIONS DETERMINED BY CONNECTION
C     DATA.
C
      IF (ITRP.NE.0) GO TO 18
C     NORMAL FILL
      DO 17 IJ=1,JSNO
      JX=JCO(IJ)
17    CM(IPR,JX)=CM(IPR,JX)+ETK*AX(IJ)+ETS*BX(IJ)+ETC*CX(IJ)
      GO TO 23
18    IF (ITRP.EQ.2) GO TO 20
C     TRANSPOSED FILL
      DO 19 IJ=1,JSNO
      JX=JCO(IJ)
19    CM(JX,IPR)=CM(JX,IPR)+ETK*AX(IJ)+ETS*BX(IJ)+ETC*CX(IJ)
      GO TO 23
C     TRANS. FILL FOR C(WW) - TEST FOR ELEMENTS FOR D(WW)PRIME.  (=CW)
20    DO 22 IJ=1,JSNO
      JX=JCO(IJ)
      IF (JX.GT.NR) GO TO 21
      CM(JX,IPR)=CM(JX,IPR)+ETK*AX(IJ)+ETS*BX(IJ)+ETC*CX(IJ)
      GO TO 22
21    JX=JX-NR
      CW(JX,IPR)=CW(JX,IPR)+ETK*AX(IJ)+ETS*BX(IJ)+ETC*CX(IJ)
22    CONTINUE
23    CONTINUE
      RETURN
      END
```

CONECT

PURPOSE

To locate segment ends that contact each other or contact the center of a surface patch.

METHOD

The ends of each segment are identified as end 1 and end 2, defined during geometry input. The connection data for segment I is stored in array variables ICON1 (I) for end 1 and ICON2 (I) for end 2.

Four conditions are possible at each segment end: (1) no connection (a free end), (2) connection to one or more other segments, (3) connection to a ground plane, or (4) connection to a surface modeled with patches. These conditions are indicated in the following way for end 1 of segment I:

(1) no connection . . . . ICON1 (I) = 0

(2) connection to segment J . . . . ICON1 (I) = ±J

(3) connection to a ground plane . . . . . . ICON1 (I) = I

(4) connection to patch K . . . . ICON1 (I) = 10000 + k

In case 2, if segment J has the same reference direction as segment I (end 2 of segment J connected to end 1 of segment I), the sign is positive. For opposed reference directions (end 1 to end 1) the sign is negative. If several segments connect to end 1 of segment I, then J is the number of the next connected segment in sequence.

If segment I connects to patch K, the segment end must coincide with the patch center. Patch K is then divided into four patches numbered K through K + 3 by a call to subroutine SUBPH.

The connection data is illustrated in the following listing for the six segments in the structure in figure 3.

ICON1 (I)	I	ICON2 (I)
10000 + K	1	2
1	2	3
4	3	0
0	4	-5
0	5	6
2	6	0

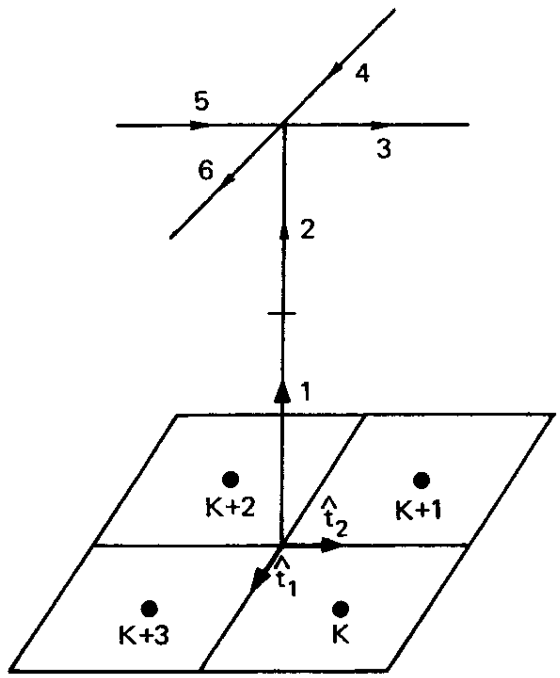


Figure 3. Structure for Illustrating Segment Connection Data.

Connections between patches are not checked, since, except where a wire connects to a surface, the current expansion function on a patch does not extend beyond that patch.

CODING

CN16 - CN27	Initialize and adjust symmetry conditions if necessary when ground is present.
CN40 - CN46	Check whether end 1 of segment I is below ground plane (error) or contacting ground plane. If the separation of the segment end and the ground is less than SMIN multiplied by the segment length, ICON1 is set to I and the z coordinate of the segment end is set to exactly zero.
CN49 - CN60	Check other segments from I + 1 through N and then 1 through I - 1, until a connected end is found. The separation of segment ends is determined by the sum of the separations in x, y, and z to save time.

CN95 - CN126	Search for segments connected to patches. Only new patches (not NGF) are checked. If a connection is found the patch is divided into four patches at its present location in the data arrays and patches following it are shifted up by three locations. This is done by calling SUBPH, an entry point of subroutine PATCH.
CN129 - CN162	Search for new segments connected to NGF patches. If a connection is found four patches, covering the area of the original patch, are added to the end of the data arrays by calling SUBPH. The original patch retains its location but the z coordinate at its center is changed to 10000.
CN182 - CN258	The loop through 44 locates segments connected to junctions.
CN183 - CN190	Parameters are initialized to find all segments connected to first end of segment J.
CN191 - CN215	Connected segments are located. If the number of any connected segment is less than J the loop is exited at CN200. Thus each junction is processed only once.
CN216 - CN230	The connected ends are set to the average of their previous values to ensure that they have identical values.
CN232 - CN244	If the junction includes new segments (NSFLG = 1) and IX is a NGF segment an equation number, NSCON, is assigned for the modified basis function of segment IX. The equation number is stored in array ICONX and the segment number is stored in ISCON.
CN245 - CN247	Segment numbers are printed for junctions of three or more segments.
CN248 - CN257	The loop is initialized for the second end of segment J and the steps from CN191 on are repeated.
CN262 - CN275	Equation numbers for modified basis functions are assigned for old segments that connect to new patches.

SYMBOL DICTIONARY

IGND	=	1 to adjust symmetry for ground and set ICON (I) = I; -1 to adjust symmetry only; 0 for no ground

CONECT

JMAX	=	maximum number of segments connected to a junction
NPMAX	=	maximum number of NGF patches connecting to new segments
NSFLG	=	1 if the junction includes any new segments when NGF is in use
NSMAX	=	maximum number of NGF segments connecting to new segments
SEP	=	approximate separation of segment ends
SLEN	=	maximum separation allowed for connection
SMIN	=	maximum separation as a fraction of segment length
XI1, YI1, ZI1	=	coordinates of end 1 of segment
XI2, YI2, ZI2	=	coordinates of end 2 of segment
XS, YS, ZS	=	coordinates of patch center

CONSTANT

1.E-3	=	maximum separation tolerance for connected segments as fraction of segment length.

```
      SUBROUTINE CONECT (IGND)
C
C     CONNECT SETS UP SEGMENT CONNECTION DATA IN ARRAYS ICON1 AND ICON2
C     BY SEARCHING FOR SEGMENT ENDS THAT ARE IN CONTACT.
C
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /SEGJ/ AX(30),BX(30),CX(30),JCO(30),JSNO,ISCON(50),NSCON,IP
     1CON(10),NPCON
      DIMENSION X2(1), Y2(1), Z2(1)
      EQUIVALENCE (X2,SI), (Y2,ALP), (Z2,BET)
      DATA JMAX/30/,SMIN/1.E-3/,NSMAX/50/,NPMAX/10/
      NSCON=0
      NPCON=0
      IF (IGND.EQ.0) GO TO 3
      PRINT 54
      IF (IGND.GT.0) PRINT 55
      IF (IPSYM.NE.2) GO TO 1
      NP=2*NP
      MP=2*MP
1     IF (IABS(IPSYM).LE.2) GO TO 2
      NP=N
      MP=M
2     IF (NP.GT.N) STOP
      IF (NP.EQ.N.AND.MP.EQ.M) IPSYM=0
3     IF (N.EQ.0) GO TO 26
      DO 15 I=1,N
      ICONX(I)=0
      XI1=X(I)
      YI1=Y(I)
      ZI1=Z(I)
      XI2=X2(I)
      YI2=Y2(I)
      ZI2=Z2(I)
      SLEN=SQRT((XI2-XI1)**2+(YI2-YI1)**2+(ZI2-ZI1)**2)*SMIN
C
C     DETERMINE CONNECTION DATA FOR END 1 OF SEGMENT.
C
      IF (IGND.LT.1) GO TO 5
      IF (ZI1.GT.-SLEN) GO TO 4
      PRINT 56, I
      STOP
4     IF (ZI1.GT.SLEN) GO TO 5
      ICON1(I)=I
      Z(I)=0.
      GO TO 9
5     IC=I
      DO 7 J=2,N
      IC=IC+1
      IF (IC.GT.N) IC=1
      SEP=ABS(XI1-X(IC))+ABS(YI1-Y(IC))+ABS(ZI1-Z(IC))
      IF (SEP.GT.SLEN) GO TO 6
      ICON1(I)=-IC
      GO TO 8
6     SEP=ABS(XI1-X2(IC))+ABS(YI1-Y2(IC))+ABS(ZI1-Z2(IC))
      IF (SEP.GT.SLEN) GO TO 7
      ICON1(I)=IC
      GO TO 8
7     CONTINUE
      IF (I.LT.N2.AND.ICON1(I).GT.10000) GO TO 8
      ICON1(I)=0
C
C     DETERMINE CONNECTION DATA FOR END 2 OF SEGMENT.
```

```
C
8     IF (IGND.LT.1) GO TO 12
9     IF (ZI2.GT.-SLEN) GO TO 10
      PRINT 56, I
      STOP
10    IF (ZI2.GT.SLEN) GO TO 12
      IF (ICON1(I).NE.I) GO TO 11
      PRINT 57, I
      STOP
11    ICON2(I)=I
      Z2(I)=0.
      GO TO 15
12    IC=I
      DO 14 J=2,N
      IC=IC+1
      IF (IC.GT.N) IC=1
      SEP=ABS(XI2-X(IC))+ABS(YI2-Y(IC))+ABS(ZI2-Z(IC))
      IF (SEP.GT.SLEN) GO TO 13
      ICON2(I)=IC
      GO TO 15
13    SEP=ABS(XI2-X2(IC))+ABS(YI2-Y2(IC))+ABS(ZI2-Z2(IC))
      IF (SEP.GT.SLEN) GO TO 14
      ICON2(I)=-IC
      GO TO 15
14    CONTINUE
      IF (I.LT.N2.AND.ICON2(I).GT.10000) GO TO 15
      ICON2(I)=0
15    CONTINUE
      IF (M.EQ.0) GO TO 26
C     FIND WIRE-SURFACE CONNECTIONS FOR NEW PATCHES
      IX=LD+1-M1
      I=M2
16    IF (I.GT.M) GO TO 20
      IX=IX-1
      XS=X(IX)
      YS=Y(IX)
      ZS=Z(IX)
      DO 18 ISEG=1,N
      XI1=X(ISEG)
      YI1=Y(ISEG)
      ZI1=Z(ISEG)
      XI2=X2(ISEG)
      YI2=Y2(ISEG)
      ZI2=Z2(ISEG)
      SLEN=(ABS(XI2-XI1)+ABS(YI2-YI1)+ABS(ZI2-ZI1))*SMIN
C     FOR FIRST END OF SEGMENT
      SEP=ABS(XI1-XS)+ABS(YI1-YS)+ABS(ZI1-ZS)
      IF (SEP.GT.SLEN) GO TO 17
C     CONNECTION - DIVIDE PATCH INTO 4 PATCHES AT PRESENT ARRAY LOC.
      ICON1(ISEG)=10000+I
      IC=0
      CALL SUBPH (I,IC,XI1,YI1,ZI1,XI2,YI2,ZI2,XA,YA,ZA,XS,YS,ZS)
      GO TO 19
17    SEP=ABS(XI2-XS)+ABS(YI2-YS)+ABS(ZI2-ZS)
      IF (SEP.GT.SLEN) GO TO 18
      ICON2(ISEG)=10000+I
      IC=0
      CALL SUBPH (I,IC,XI1,YI1,ZI1,XI2,YI2,ZI2,XA,YA,ZA,XS,YS,ZS)
      GO TO 19
18    CONTINUE
19    I=I+1
      GO TO 16
C     REPEAT SEARCH FOR NEW SEGMENTS CONNECTED TO NGF PATCHES.
20    IF (M1.EQ.0.OR.N2.GT.N) GO TO 26
```

```
      IX=LD+1
      I=1
   21 IF (I.GT.M1) GO TO 25
      IX=IX-1
      XS=X(IX)
      YS=Y(IX)
      ZS=Z(IX)
      DO 23 ISEG=N2,N
      XI1=X(ISEG)
      YI1=Y(ISEG)
      ZI1=Z(ISEG)
      XI2=X2(ISEG)
      YI2=Y2(ISEG)
      ZI2=Z2(ISEG)
      SLEN=(ABS(XI2-XI1)+ABS(YI2-YI1)+ABS(ZI2-ZI1))*SMIN
      SEP=ABS(XI1-XS)+ABS(YI1-YS)+ABS(ZI1-ZS)
      IF (SEP.GT.SLEN) GO TO 22
      ICON1(ISEG)=10001+M
      IC=1
      NPCON=NPCON+1
      IPCON(NPCON)=I
      CALL SUBPH (I,IC,XI1,YI1,ZI1,XI2,YI2,ZI2,XA,YA,ZA,XS,YS,ZS)
      GO TO 24
   22 SEP=ABS(XI2-XS)+ABS(YI2-YS)+ABS(ZI2-ZS)
      IF (SEP.GT.SLEN) GO TO 23
      ICON2(ISEG)=10001+M
      IC=1
      NPCON=NPCON+1
      IPCON(NPCON)=I
      CALL SUBPH (I,IC,XI1,YI1,ZI1,XI2,YI2,ZI2,XA,YA,ZA,XS,YS,ZS)
      GO TO 24
   23 CONTINUE
   24 I=I+1
      GO TO 21
   25 IF (NPCON.LE.NPMAX) GO TO 26
      PRINT 62, NPMAX
      STOP
   26 PRINT 58, N,NP,IPSYM
      IF (M.GT.0) PRINT 61, M,MP
      ISEG=(N+M)/(NP+MP)
      IF (ISEG.EQ.1) GO TO 30
      IF (IPSYM) 28,27,29
   27 STOP
   28 PRINT 59, ISEG
      GO TO 30
   29 IC=ISEG/2
      IF (ISEG.EQ.8) IC=3
      PRINT 60, IC
   30 IF (N.EQ.0) GO TO 48
      PRINT 50
      ISEG=0
C     ADJUST CONNECTED SEG. ENDS TO EXACTLY COINCIDE.  PRINT JUNCTIONS
C     OF 3 OR MORE SEG.  ALSO FIND OLD SEG. CONNECTING TO NEW SEG.
      DO 44 J=1,N
      IEND=-1
      JEND=-1
      IX=ICON1(J)
      IC=1
      JCO(1)=-J
      XA=X(J)
      YA=Y(J)
      ZA=Z(J)
   31 IF (IX.EQ.0) GO TO 43
      IF (IX.EQ.J) GO TO 43
```

```
      IF (IX.GT.10000) GO TO 43                                         CN 193
      NSFLG=0                                                           CN 194
   32 IF (IX) 33,49,34                                                  CN 195
   33 IX=-IX                                                            CN 196
      GO TO 35                                                          CN 197
   34 JEND=-JEND                                                        CN 198
   35 IF (IX.EQ.J) GO TO 37                                             CN 199
      IF (IX.LT.J) GO TO 43                                             CN 200
      IC=IC+1                                                           CN 201
      IF (IC.GT.JMAX) GO TO 49                                          CN 202
      JCO(IC)=IX*JEND                                                   CN 203
      IF (IX.GT.N1) NSFLG=1                                             CN 204
      IF (JEND.EQ.1) GO TO 36                                           CN 205
      XA=XA+X(IX)                                                       CN 206
      YA=YA+Y(IX)                                                       CN 207
      ZA=ZA+Z(IX)                                                       CN 208
      IX=ICON1(IX)                                                      CN 209
      GO TO 32                                                          CN 210
   36 XA=XA+X2(IX)                                                      CN 211
      YA=YA+Y2(IX)                                                      CN 212
      ZA=ZA+Z2(IX)                                                      CN 213
      IX=ICON2(IX)                                                      CN 214
      GO TO 32                                                          CN 215
   37 SEP=IC                                                            CN 216
      XA=XA/SEP                                                         CN 217
      YA=YA/SEP                                                         CN 218
      ZA=ZA/SEP                                                         CN 219
      DO 39 I=1,IC                                                      CN 220
      IX=JCO(I)                                                         CN 221
      IF (IX.GT.0) GO TO 38                                             CN 222
      IX=-IX                                                            CN 223
      X(IX)=XA                                                          CN 224
      Y(IX)=YA                                                          CN 225
      Z(IX)=ZA                                                          CN 226
      GO TO 39                                                          CN 227
   38 X2(IX)=XA                                                         CN 228
      Y2(IX)=YA                                                         CN 229
      Z2(IX)=ZA                                                         CN 230
   39 CONTINUE                                                          CN 231
      IF (N1.EQ.0) GO TO 42                                             CN 232
      IF (NSFLG.EQ.0) GO TO 42                                          CN 233
      DO 41 I=1,IC                                                      CN 234
      IX=IABS(JCO(I))                                                   CN 235
      IF (IX.GT.N1) GO TO 41                                            CN 236
      IF (ICONX(IX).NE.0) GO TO 41                                      CN 237
      NSCON=NSCON+1                                                     CN 238
      IF (NSCON.LE.NSMAX) GO TO 40                                      CN 239
      PRINT 62, NSMAX                                                   CN 240
      STOP                                                              CN 241
   40 ISCON(NSCON)=IX                                                   CN 242
      ICONX(IX)=NSCON                                                   CN 243
   41 CONTINUE                                                          CN 244
   42 IF (IC.LT.3) GO TO 43                                             CN 245
      ISEG=ISEG+1                                                       CN 246
      PRINT 51, ISEG,(JCO(I),I=1,IC)                                    CN 247
   43 IF (IEND.EQ.1) GO TO 44                                           CN 248
      IEND=1                                                            CN 249
      JEND=1                                                            CN 250
      IX=ICON2(J)                                                       CN 251
      IC=1                                                              CN 252
      JCO(1)=J                                                          CN 253
      XA=X2(J)                                                          CN 254
      YA=Y2(J)                                                          CN 255
      ZA=Z2(J)                                                          CN 256
```

```
      GO TO 31                                                          CN 257
   44 CONTINUE                                                          CN 258
      IF (ISEG.EQ.0) PRINT 52                                           CN 259
      IF (N1.EQ.0.OR.M1.EQ.M) GO TO 48                                  CN 260
C     FIND OLD SEGMENTS THAT CONNECT TO NEW PATCHES                     CN 261
      DO 47 J=1,N1                                                      CN 262
      IX=ICON1(J)                                                       CN 263
      IF (IX.LT.10000) GO TO 45                                         CN 264
      IX=IX-10000                                                       CN 265
      IF (IX.GT.M1) GO TO 46                                            CN 266
   45 IX=ICON2(J)                                                       CN 267
      IF (IX.LT.10000) GO TO 47                                         CN 268
      IX=IX-10000                                                       CN 269
      IF (IX.LT.M2) GO TO 47                                            CN 270
   46 IF (ICONX(J).NE.0) GO TO 47                                       CN 271
      NSCON=NSCON+1                                                     CN 272
      ISCON(NSCON)=J                                                    CN 273
      ICONX(J)=NSCON                                                    CN 274
   47 CONTINUE                                                          CN 275
   48 CONTINUE                                                          CN 276
      RETURN                                                            CN 277
   49 PRINT 53, IX                                                      CN 278
      STOP                                                              CN 279
C                                                                       CN 280
   50 FORMAT (//,9X,27H- MULTIPLE WIRE JUNCTIONS -,/,1X,8HJUNCTION,4X,36 CN 281
     1HSEGMENTS  (- FOR END 1, + FOR END 2))                            CN 282
   51 FORMAT (1X,I5,5X,20I5,/,(11X,20I5))                               CN 283
   52 FORMAT (2X,4HNONE)                                                CN 284
   53 FORMAT (47H CONNECT - SEGMENT CONNECTION ERROR FOR SEGMENT,I5)    CN 285
   54 FORMAT (/,3X,23HGROUND PLANE SPECIFIED.)                          CN 286
   55 FORMAT (/,3X,46HWHERE WIRE ENDS TOUCH GROUND, CURRENT WILL BE ,38H CN 287
     1INTERPOLATED TO IMAGE IN GROUND PLANE.,/)                         CN 288
   56 FORMAT (30H GEOMETRY DATA ERROR-- SEGMENT,I5,21H EXTENDS BELOW GRO CN 289
     1UND)                                                              CN 290
   57 FORMAT (29H GEOMETRY DATA ERROR--SEGMENT,I5,16H LIES IN GROUND ,6H CN 291
     1PLANE.)                                                           CN 292
   58 FORMAT (/,3X,20HTOTAL SEGMENTS USED=,I5,5X,12HNO. SEG. IN ,17HA SY CN 293
     1MMETRIC CELL=,I5,5X,14HSYMMETRY FLAG=,I3)                         CN 294
   59 FORMAT (14H STRUCTURE HAS,I4,25H FOLD ROTATIONAL SYMMETRY,/)      CN 295
   60 FORMAT (14H STRUCTURE HAS,I2,19H PLANES OF SYMMETRY,/)            CN 296
   61 FORMAT (3X,19HTOTAL PATCHES USED=,I5,6X,32HNO. PATCHES IN A SYMMET CN 297
     1RIC CELL=,I5)                                                     CN 298
   62 FORMAT (82H ERROR - NO. NEW SEGMENTS CONNECTED TO N.G.F. SEGMENTSO CN 299
     1R PATCHES EXCEEDS LIMIT OF,I5)                                    CN 300
      END                                                               CN 301-
```

COUPLE

PURPOSE

To compute the maximum coupling between pairs of segments.

METHOD

If a coupling calculation has been requested (CP card) subroutine COUPLE is called each time that the current is computed for a new excitation. The code from CP10 to CP12 checks that the excitation is a single applied-field voltage source on the segment specified in NCTAG and NCSEG. If the excitation is correct the input admittance and mutual admittances to all other segments specified in NCTAG and NCSEG are stored in Y11A and Y12A from CP13 to CP22.

When all segments have been excited (ICOUP = NCOUP) the second part of the code, from CP24 to CP58 is executed to evaluate the equations in Section V.6 of Part I.

SYMBOL DICTIONARY

C = L (see Part I, Section V.6)

CUR = array of values of current at the centers of segments

DBC = 10 $\log(G_{MAX})$

GMAX = G_{MAX}

ISG1 = segment number

ISG2 = segment number

J1 = index of Y_{12} in array Y12A

J2 = index of Y_{21} in array Y12A

K = segment number

RHO = ρ

WLAM = wavelength

Y11 = Y_{11}

Y12 = $(Y_{12} + Y_{21})/2$

Y22 = Y_{22}

YIN = Y_{IN}

YL = Y_L

ZIN = $1/Y_{IN}$

ZL = $1/Y_L$

```
      SUBROUTINE COUPLE (CUR,WLAM)
C
C     COUPLE COMPUTES THE MAXIMUM COUPLING BETWEEN PAIRS OF SEGMENTS.
C
      COMPLEX Y11A,Y12A,CUR,Y11,Y12,Y22,YL,YIN,ZL,ZIN,RHO,VQD,VSANT,VQDS
      COMMON /YPARM/ NCOUP,ICOUP,NCTAG(5),NCSEG(5),Y11A(5),Y12A(20)
      COMMON /VSORC/ VQD(30),VSANT(30),VQDS(30),IVQD(30),ISANT(30),IQDS(
     130),NVQD,NSANT,NQDS
      DIMENSION CUR(1)
      IF (NSANT.NE.1.OR.NVQD.NE.0) RETURN
      J=ISEGNO(NCTAG(ICOUP+1),NCSEG(ICOUP+1))
      IF (J.NE.ISANT(1)) RETURN
      ICOUP=ICOUP+1
      ZIN=VSANT(1)
      Y11A(ICOUP)=CUR(J)*WLAM/ZIN
      L1=(ICOUP-1)*(NCOUP-1)
      DO 1 I=1,NCOUP
      IF (I.EQ.ICOUP) GO TO 1
      K=ISEGNO(NCTAG(I),NCSEG(I))
      L1=L1+1
      Y12A(L1)=CUR(K)*WLAM/ZIN
1     CONTINUE
      IF (ICOUP.LT.NCOUP) RETURN
      PRINT 6
      NPM1=NCOUP-1
      DO 5 I=1,NPM1
      ITT1=NCTAG(I)
      ITS1=NCSEG(I)
      ISG1=ISEGNO(ITT1,ITS1)
      L1=I+1
      DO 5 J=L1,NCOUP
      ITT2=NCTAG(J)
      ITS2=NCSEG(J)
      ISG2=ISEGNO(ITT2,ITS2)
      J1=J+(I-1)*NPM1-1
      J2=I+(J-1)*NPM1
      Y11=Y11A(I)
      Y22=Y11A(J)
      Y12=.5*(Y12A(J1)+Y12A(J2))
      YIN=Y12*Y12
      DBC=CABS(YIN)
      C=DBC/(2.*REAL(Y11)*REAL(Y22)-REAL(YIN))
      IF (C.LT.0..OR.C.GT.1.) GO TO 4
      IF (C.LT..01) GO TO 2
      GMAX=(1.-SQRT(1.-C*C))/C
      GO TO 3
2     GMAX=.5*(C+.25*C*C*C)
3     RHO=GMAX*CONJG(YIN)/DBC
      YL=((1.-RHO)/(1.+RHO)+1.)*REAL(Y22)-Y22
      ZL=1./YL
      YIN=Y11-YIN/(Y22+YL)
      ZIN=1./YIN
      DBC=DB10(GMAX)
      PRINT 7, ITT1,ITS1,ISG1,ITT2,ITS2,ISG2,DBC,ZL,ZIN
      GO TO 5
4     PRINT 8, ITT1,ITS1,ISG1,ITT2,ITS2,ISG2,C
5     CONTINUE
      RETURN
C
6     FORMAT (///,36X,26H- - - ISOLATION DATA - - -,//,6X,24H- - COUPLIN
     1G BETWEEN - -,8X,7HMAXIMUM,15X,32H- - - FOR MAXIMUM COUPLING - - -
     2,/,12X,4HSEG.,14X,4HSEG.,3X,8HCOUPLING,4X,25HLOAD IMPEDANCE (2ND S
     3EG.),7X,15HINPUT IMPEDANCE,/,2X,8HTAG/SEG.,3X,3HNO.,4X,8HTAG/SEG.,
     43X,3HNO.,6X,4H(DB),8X,4HREAL,9X,5HIMAG.,9X,4HREAL,9X,5HIMAG.)
```

```
      7  FORMAT (2(1X,I4,1X,I4,1X,I5,2X),F9.3,2X,2(2X,E12.5,1X,E12.5))     CP  65
      8  FORMAT (2(1X,I4,1X,I4,1X,I5,2X)45H**ERROR** COUPLING IS NOT BETWEE CP  66
        1N 0 AND 1. (=,E12.5,1H))                                          CP  67
         END                                                               CP  68-
```

PURPOSE

To read structure input data and set segment and patch data.

METHOD

The main READ statement is at DA35. The READ statement at DA65 is for the continuation of wire data (GC card following GW), and the READ at DA133 is for the continuation of surface patch data (SC following SP or SM).

The first input parameter GM determines the function of the card as indicated in the following table:

GM	GØ TØ	FUNCTION
GA	8	define wire arc
GC	6	continuation of wire data
GE	29	end of geometry data
GF	27	read NGF file
GM	26	rotate or translate structure
GR	19	rotate about Z axis (symmetry)
GS	21	scale structure
GW	3	define straight wire
GX	18	reflect in coordinate planes (symmetry)
SC	10	continuation of patch data
SM	13	define multiple surface patches
SP	9	define surface patch

The functions of the other input parameters depend on the type of data card and can be determined from the data card descriptions in Part III of this manual.

Subroutines are called to perform many of the operations requested by the data cards. Coding in DATAGN performs other operations, prints information and checks for input errors. After a GE card is read subroutine CONECT is called at DA211 to find electrical connections of segments. Segment and patch data is printed from DA217 to DA256. Line DA241 tests for segments of zero length ($<10^{-20}$) or zero radius ($<10^{-101}$).

SYMBOL DICTIONARY

Variables have multiple uses which depend on the type of input card being processed.

```
      SUBROUTINE DATAGN
C
C     DATAGN IS THE MAIN ROUTINE FOR INPUT OF GEOMETRY DATA.
C
      INTEGER GM,ATST
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /ANGL/ SALP(300)
      DIMENSION X2(1), Y2(1), Z2(1), T1X(1), T1Y(1), T1Z(1), T2X(1), T2Y
     1(1), T2Z(1), ATST(12), IFX(2), IFY(2), IFZ(2), CAB(1), SAB(1), IPT
     2(4)
      EQUIVALENCE (T1X,SI), (T1Y,ALP), (T1Z,BET), (T2X,ICON1), (T2Y,ICON
     12), (T2Z,ITAG), (X2,SI), (Y2,ALP), (Z2,BET), (CAB,ALP), (SAB,BET)
      DATA ATST/2HGW,2HGX,2HGR,2HGS,2HGE,2HGM,2HSP,2HSM,2HGF,2HGA,2HSC,2
     1HGC/
      DATA IFX/1H ,1HX/,IFY/1H ,1HY/,IFZ/1H ,1HZ/
      DATA TA/0.01745329252/,TD/57.29577951/,IPT/1HP,1HR,1HT,1HQ/
      IPSYM=0
      NWIRE=0
      N=0
      NP=0
      M=0
      MP=0
      N1=0
      N2=1
      M1=0
      M2=1
      ISCT=0
      IPHD=0
C
C     READ GEOMETRY DATA CARD AND BRANCH TO SECTION FOR OPERATION
C     REQUESTED
C
1     READ (5,42) GM,ITG,NS,XW1,YW1,ZW1,XW2,YW2,ZW2,RAD
      IF (N+M.GT.LD) GO TO 37
      IF (GM.EQ.ATST(9)) GO TO 27
      IF (IPHD.EQ.1) GO TO 2
      PRINT 40
      PRINT 41
      IPHD=1
2     IF (GM.EQ.ATST(11)) GO TO 10
      ISCT=0
      IF (GM.EQ.ATST(1)) GO TO 3
      IF (GM.EQ.ATST(2)) GO TO 18
      IF (GM.EQ.ATST(3)) GO TO 19
      IF (GM.EQ.ATST(4)) GO TO 21
      IF (GM.EQ.ATST(7)) GO TO 9
      IF (GM.EQ.ATST(8)) GO TO 13
      IF (GM.EQ.ATST(5)) GO TO 29
      IF (GM.EQ.ATST(6)) GO TO 26
      IF (GM.EQ.ATST(10)) GO TO 8
      GO TO 36
C
C     GENERATE SEGMENT DATA FOR STRAIGHT WIRE.
C
3     NWIRE=NWIRE+1
      I1=N+1
      I2=N+NS
      PRINT 43, NWIRE,XW1,YW1,ZW1,XW2,YW2,ZW2,RAD,NS,I1,I2,ITG
      IF (RAD.EQ.0) GO TO 4
      XS1=1.
      YS1=1.
      GO TO 7
```

```
    4 READ (5,42) GM,IX,IY,XS1,YS1,ZS1
      IF (GM.EQ.ATST(12)) GO TO 6
    5 PRINT 48
      STOP
    6 PRINT 61, XS1,YS1,ZS1
      IF (YS1.EQ.0.OR.ZS1.EQ.0) GO TO 5
      RAD=YS1
      YS1=(ZS1/YS1)**(1./(NS-1.))
    7 CALL WIRE (XW1,YW1,ZW1,XW2,YW2,ZW2,RAD,XS1,YS1,NS,ITG)
      GO TO 1
C
C     GENERATE SEGMENT DATA FOR WIRE ARC
C
    8 NWIRE=NWIRE+1
      I1=N+1
      I2=N+NS
      PRINT 38, NWIRE,XW1,YW1,ZW1,XW2,NS,I1,I2,ITG
      CALL ARC (ITG,NS,XW1,YW1,ZW1,XW2)
      GO TO 1
C
C     GENERATE SINGLE NEW PATCH
C
    9 I1=M+1
      NS=NS+1
      IF (ITG.NE.0) GO TO 17
      PRINT 51, I1,IPT(NS),XW1,YW1,ZW1,XW2,YW2,ZW2
      IF (NS.EQ.2.OR.NS.EQ.4) ISCT=1
      IF (NS.GT.1) GO TO 14
      XW2=XW2*TA
      YW2=YW2*TA
      GO TO 16
   10 IF (ISCT.EQ.0) GO TO 17
      I1=M+1
      NS=NS+1
      IF (ITG.NE.0) GO TO 17
      IF (NS.NE.2.AND.NS.NE.4) GO TO 17
      XS1=X4
      YS1=Y4
      ZS1=Z4
      XS2=X3
      YS2=Y3
      ZS2=Z3
      X3=XW1
      Y3=YW1
      Z3=ZW1
      IF (NS.NE.4) GO TO 11
      X4=XW2
      Y4=YW2
      Z4=ZW2
   11 XW1=XS1
      YW1=YS1
      ZW1=ZS1
      XW2=XS2
      YW2=YS2
      ZW2=ZS2
      IF (NS.EQ.4) GO TO 12
      X4=XW1+X3-XW2
      Y4=YW1+Y3-YW2
      Z4=ZW1+Z3-ZW2
   12 PRINT 51, I1,IPT(NS),XW1,YW1,ZW1,XW2,YW2,ZW2
      PRINT 39, X3,Y3,Z3,X4,Y4,Z4
      GO TO 16
C
C     GENERATE MULTIPLE-PATCH SURFACE
```

```
C
   13 I1=M+1
      PRINT 59, I1,IPT(2),XW1,YW1,ZW1,XW2,YW2,ZW2,ITG,NS
      IF (ITG.LT.1.OR.NS.LT.1) GO TO 17
   14 READ (5,42) GM,IX,IY,X3,Y3,Z3,X4,Y4,Z4
      IF (NS.NE.2.AND.ITG.LT.1) GO TO 15
      X4=XW1+X3-XW2
      Y4=YW1+Y3-YW2
      Z4=ZW1+Z3-ZW2
   15 PRINT 39, X3,Y3,Z3,X4,Y4,Z4
      IF (GM.NE.ATST(11)) GO TO 17
   16 CALL PATCH (ITG,NS,XW1,YW1,ZW1,XW2,YW2,ZW2,X3,Y3,Z3,X4,Y4,Z4)
      GO TO 1
   17 PRINT 60
      STOP
C
C     REFLECT STRUCTURE ALONG X,Y, OR Z AXES OR ROTATE TO FORM CYLINDER.
C
   18 IY=NS/10
      IZ=NS-IY*10
      IX=IY/10
      IY=IY-IX*10
      IF (IX.NE.0) IX=1
      IF (IY.NE.0) IY=1
      IF (IZ.NE.0) IZ=1
      PRINT 44, IFX(IX+1),IFY(IY+1),IFZ(IZ+1),ITG
      GO TO 20
   19 PRINT 45, NS,ITG
      IX=-1
   20 CALL REFLC (IX,IY,IZ,ITG,NS)
      GO TO 1
C
C     SCALE STRUCTURE DIMENSIONS BY FACTOR XW1.
C
   21 IF (N.LT.N2) GO TO 23
      DO 22 I=N2,N
      X(I)=X(I)*XW1
      Y(I)=Y(I)*XW1
      Z(I)=Z(I)*XW1
      X2(I)=X2(I)*XW1
      Y2(I)=Y2(I)*XW1
      Z2(I)=Z2(I)*XW1
   22 BI(I)=BI(I)*XW1
   23 IF (M.LT.M2) GO TO 25
      YW1=XW1*XW1
      IX=LD+1-M
      IY=LD-M1
      DO 24 I=IX,IY
      X(I)=X(I)*XW1
      Y(I)=Y(I)*XW1
      Z(I)=Z(I)*XW1
   24 BI(I)=BI(I)*YW1
   25 PRINT 46, XW1
      GO TO 1
C
C     MOVE STRUCTURE OR REPRODUCE ORIGINAL STRUCTURE IN NEW POSITIONS.
C
   26 PRINT 47, ITG,NS,XW1,YW1,ZW1,XW2,YW2,ZW2,RAD
      XW1=XW1*TA
      YW1=YW1*TA
      ZW1=ZW1*TA
      CALL MOVE (XW1,YW1,ZW1,XW2,YW2,ZW2,INT(RAD+.5),NS,ITG)
      GO TO 1
C
```

```
C     READ NUMERICAL GREEN'S FUNCTION TAPE
C
   27 IF (N+M.EQ.0) GO TO 28
      PRINT 52
      STOP
   28 CALL GFIL (ITG)
      NPSAV=NP
      MPSAV=MP
      IPSAV=IPSYM
      GO TO 1
C
C     TERMINATE STRUCTURE GEOMETRY INPUT.
C
   29 IX=N1+M1
      IF (IX.EQ.0) GO TO 30
      NP=N
      MP=M
      IPSYM=0
   30 CALL CONECT (ITG)
      IF (IX.EQ.0) GO TO 31
      NP=NPSAV
      MP=MPSAV
      IPSYM=IPSAV
   31 IF (N+M.GT.LD) GO TO 37
      IF (N.EQ.0) GO TO 33
      PRINT 53
      PRINT 54
      DO 32 I=1,N
      XW1=X2(I)-X(I)
      YW1=Y2(I)-Y(I)
      ZW1=Z2(I)-Z(I)
      X(I)=(X(I)+X2(I))*.5
      Y(I)=(Y(I)+Y2(I))*.5
      Z(I)=(Z(I)+Z2(I))*.5
      XW2=XW1*XW1+YW1*YW1+ZW1*ZW1
      YW2=SQRT(XW2)
      YW2=(XW2/YW2+YW2)*.5
      SI(I)=YW2
      CAB(I)=XW1/YW2
      SAB(I)=YW1/YW2
      XW2=ZW1/YW2
      IF (XW2.GT.1.) XW2=1.
      IF (XW2.LT.-1.) XW2=-1.
      SALP(I)=XW2
      XW2=ASIN(XW2)*TD
      YW2=ATGN2(YW1,XW1)*TD
      PRINT 55, I,X(I),Y(I),Z(I),SI(I),XW2,YW2,BI(I),ICON1(I),I,ICON2(I)
     1,ITAG(I)
      IF (SI(I).GT.1.E-20.AND.BI(I).GT.1.E-101) GO TO 32
      PRINT 56
      STOP
   32 CONTINUE
   33 IF (M.EQ.0) GO TO 35
      PRINT 57
      J=LD+1
      DO 34 I=1,M
      J=J-1
      XW1=(T1Y(J)*T2Z(J)-T1Z(J)*T2Y(J))*SALP(J)
      YW1=(T1Z(J)*T2X(J)-T1X(J)*T2Z(J))*SALP(J)
      ZW1=(T1X(J)*T2Y(J)-T1Y(J)*T2X(J))*SALP(J)
      PRINT 58, I,X(J),Y(J),Z(J),XW1,YW1,ZW1,BI(J),T1X(J),T1Y(J),T1Z(J),
     1T2X(J),T2Y(J),T2Z(J)
   34 CONTINUE
   35 RETURN
```

```
   36 PRINT 48                                                          DA 257
      PRINT 49, GM,ITG,NS,XW1,YW1,ZW1,XW2,YW2,ZW2,RAD                   DA 258
      STOP                                                              DA 259
   37 PRINT 50                                                          DA 260
      STOP                                                              DA 261
C                                                                       DA 262
   38 FORMAT (1X,I5,2X,12HARC RADIUS =,F9.5,2X,4HFROM,F8.3,3H TO,F8.3,8H DA 263
     1 DEGREES,11X,F11.5,2X,I5,4X,I5,1X,I5,3X,I5)                       DA 264
   39 FORMAT (6X,3F11.5,1X,3F11.5)                                      DA 265
   40 FORMAT (////,33X,35H- - - STRUCTURE SPECIFICATION - - -,//,37X,28H DA 266
     1COORDINATES MUST BE INPUT IN,/,37X,29HMETERS OR BE SCALED TO METER DA 267
     2S,/,37X,31HBEFORE STRUCTURE INPUT IS ENDED,//)                    DA 268
   41 FORMAT (2X,4HWIRE,79X,6HNO. OF,4X,5HFIRST,2X,4HLAST,5X,3HTAG,/,2X, DA 269
     13HNO.,8X,2HX1,9X,2HY1,9X,2HZ1,10X,2HX2,9X,2HY2,9X,2HZ2,6X,6HRADIUS DA 270
     2,3X,4HSEG.,5X,4HSEG.,3X,4HSEG.,5X,3HNO.)                          DA 271
   42 FORMAT (A2,I3,I5,7F10.5)                                          DA 272
   43 FORMAT (1X,I5,3F11.5,1X,4F11.5,2X,I5,4X,I5,1X,I5,3X,I5)           DA 273
   44 FORMAT (6X,34HSTRUCTURE REFLECTED ALONG THE AXES,3(1X,A1),22H.  TA DA 274
     1GS INCREMENTED BY,I5)                                             DA 275
   45 FORMAT (6X,30HSTRUCTURE ROTATED ABOUT Z-AXIS,I3,30H TIMES.  LABLES DA 276
     1 INCREMENTED BY,I5)                                               DA 277
   46 FORMAT (6X,26HSTRUCTURE SCALED BY FACTOR,F10.5)                   DA 278
   47 FORMAT (6X,49HTHE STRUCTURE HAS BEEN MOVED, MOVE DATA CARD IS -/6X DA 279
     1,I3,I5,7F10.5)                                                    DA 280
   48 FORMAT (25H GEOMETRY DATA CARD ERROR)                             DA 281
   49 FORMAT (1X,A2,I3,I5,7F10.5)                                       DA 282
   50 FORMAT (69H NUMBER OF WIRE SEGMENTS AND SURFACE PATCHES EXCEEDS DI DA 283
     1MENSION LIMIT.)                                                   DA 284
   51 FORMAT (1X,I5,A1,F10.5,2F11.5,1X,3F11.5)                          DA 285
   52 FORMAT (44H ERROR - GF MUST BE FIRST GEOMETRY DATA CARD)          DA 286
   53 FORMAT (////33X,33H- - - SEGMENTATION DATA - - - -,//,40X,21HCOO DA 287
     1RDINATES IN METERS,//,25X,50HI+ AND I- INDICATE THE SEGMENTS BEFOR DA 288
     2E AND AFTER I,//)                                                 DA 289
   54 FORMAT (2X,4HSEG.,3X,26HCOORDINATES OF SEG. CENTER,5X,4HSEG.,5X,18 DA 290
     1HORIENTATION ANGLES,4X,4HWIRE,4X,15HCONNECTION DATA,3X,3HTAG,/,2X, DA 291
     23HNO.,7X,1HX,9X,1HY,9X,1HZ,7X,6HLENGTH,5X,5HALPHA,5X,4HBETA,6X,6HR DA 292
     3ADIUS,4X,2HI-,3X,1HI,4X,2HI+,4X,3HNO.)                            DA 293
   55 FORMAT (1X,I5,4F10.5,1X,3F10.5,1X,3I5,2X,I5)                      DA 294
   56 FORMAT (19H SEGMENT DATA ERROR)                                   DA 295
   57 FORMAT (////,44X,30H- - - SURFACE PATCH DATA - - -,//,49X,21HCOORD DA 296
     1INATES IN METERS,//,1X,5HPATCH,5X,22HCOORD. OF PATCH CENTER,7X,18H DA 297
     2UNIT NORMAL VECTOR,6X,5HPATCH,12X,34HCOMPONENTS OF UNIT TANGENT VE DA 298
     3CTORS,/,2X,3HNO.,6X,1HX,9X,1HY,9X,1HZ,9X,1HX,7X,1HY,7X,1HZ,7X,4HAR DA 299
     4EA,7X,2HX1,6X,2HY1,6X,2HZ1,7X,2HX2,6X,2HY2,6X,2HZ2)               DA 300
   58 FORMAT (1X,I4,3F10.5,1X,3F8.4,F10.5,1X,3F8.4,1X,3F8.4)            DA 301
   59 FORMAT (1X,I5,A1,F10.5,2F11.5,1X,3F11.5,5X,9HSURFACE -,I4,3H BY,I3 DA 302
     1,8H PATCHES)                                                      DA 303
   60 FORMAT (17H PATCH DATA ERROR)                                     DA 304
   61 FORMAT (9X,43HABOVE WIRE IS TAPERED.  SEG. LENGTH RATIO =,F9.5,/,3 DA 305
     13X,11HRADIUS FROM,F9.5,3H TO,F9.5)                                DA 306
      END                                                               DA 307-
```

DB10

PURPOSE

To convert an input magnitude quantity (field) or magnitude squared quantity (power) into decibels.

METHOD

For a squared quantity, the decibel conversion is

$$Q_{db} = 10 \log_{10} Q^2 \qquad (Q^2 \text{ input}) ,$$

and for an unsquared quantity,

$$Q_{db} = 20 \log_{10} Q .$$

DB10 is used for the squared quantity while the entry DB20 is used for the quantity which is not squared.

SYMBOL DICTIONARY

ALOG10 = external routine (log to the base 10)

DB10 = Q_{db}

F = scaling term

X = input quantity

CONSTANT

-999.99 = returned for an input less than 10^{-20}

CODE LISTING

```
        FUNCTION DB10 (X)                                                       DB   1
C                                                                               DB   2
C       FUNCTION DB-- RETURNS DB FOR MAGNITUDE (FIELD) OR MAG**2 (POWER) I     DB   3
C                                                                               DB   4
        F=10.                                                                   DB   5
        GO TO 1                                                                 DB   6
        ENTRY DB20                                                              DB   7
        F=20.                                                                   DB   8
1       IF (X.LT.1.E-20) GO TO 2                                                DB   9
        DB10=F*ALOG10(X)                                                        DB  10
        RETURN                                                                  DB  11
2       DB10=-999.99                                                            DB  12
        RETURN                                                                  DB  13
        END                                                                     DB  14-
```

PURPOSE

To compute the near electric field due to constant, sine, and cosine current distributions on a segment in free space or over ground.

METHOD

The electric field is computed at the point XI, YI, ZI due to the segment defined by parameters in COMMON/DATAJ/. Either the thin wire or extended thin wire formulas may be used. When a ground is present, the code is executed twice in a loop. In the second pass, the field of the image of the segment is computed, multiplied by the reflection coefficients, and added to the direct field. The reflection coefficients for the reflected ray from the center of the source segment are used for the entire segment.

The field is evaluated in a cylindrical coordinate system with the source segment at the origin, along the z axis. The ρ coordinate of the field evaluation point is computed for the surface of the observation segment as

$$\rho' = (\rho^2 + a^2)^{1/2},$$

where ρ is the distance from the axis of the source segment to (XI, YI, ZI) and a is the radius of the observation segment. The field is computed in ρ and z components as

$$\bar{E} = E_\rho(\bar{\rho}/\rho') + E_z\hat{z} \ .$$

Use of ρ' avoids a singularity when (XI, YI, ZI) is the center of the source segment. In the addition of field components, $\bar{\rho}/\rho'$ is used rather than $\hat{\rho}$, since E_ρ is the field in the direction $\hat{\rho}'$ to one side of the observation segment.

When the Sommerfeld/Norton option is used for an antenna over ground the electric field at $\bar{r}$ due to the current on a segment is evaluated in three terms as

$$\bar{E}(\bar{r}) = \bar{E}_D(\bar{r}) + \frac{k_1^2 - k_2^2}{k_1^2 + k_2^2}\,\bar{E}_I(\bar{r}) + \bar{E}_S(\bar{r})$$

$\bar{E}_D$ is the direct field of the segment in the absence of ground, and $\bar{E}_I$ is the field of the image of the segment reflected in a perfectly conducting ground. These field comonents are evaluated in EFLD between EF19 and EF150. The factor $(k_1^2 - k_2^2)/(k_1^2 + k_2^2)$ is contained in the variable FRATI.

The field $\bar{E}_S$, due to the Sommerfeld integrals is evaluated from EF155 to EF227. If the separation of the observation point and the center of the source segment is less than one wavelength, subroutine ROM2 is called at EF191 to integrate over the segment. DMIN is set to the magnitude of the first two terms in $\bar{E}$ divided by 100 as a lower limit on the denominator of the relative error test in the numerical integration. This relaxes the relative accuracy requirement when $\bar{E}_S$ is small compared to the first two terms.

If the separation of the source segment and observation point is greater than a wavelength, SFLDS is called at EF197 to evaluate $\bar{E}_S$ by the Norton approximation.

To compute $\bar{E}_S$ with the thin wire approximation applied in a manner consistent with that for $\bar{E}_I$, the field is evaluated at a point displaced normal to the image of the source segment and normal to the separation $\bar{R}$. If the direction of the image of the source segment is $\hat{j}$ the displacement is $\bar{D}$ where

$$\bar{D} = \pm a\hat{d} \text{ for } \hat{z} \cdot \hat{d} \gtrless 0$$

$$\hat{d} = (\hat{j} \times \bar{R})/|\hat{j} \times \bar{R}|$$

$$a = \text{radius of observation segment}$$

This displaced observation point (XO, YO, ZO) is computed from EF166 to EF181. Some of the complexity is needed to make the result independent of orientation of segments relative to the coordinate axes.

To adjust the ρ component of field for the factor $|\bar{\rho}/\rho'|$ the field $\bar{E}'$ is computed as

$$\bar{E}' = F\bar{E} + (1 - F)(\bar{E} \cdot \hat{j})\hat{j}$$

where $F = [\rho^2/(\rho^2 + a^2)]^{1/2}$

$$\rho^2 = |\bar{R}|^2 - (\bar{R} \cdot \hat{j})^2$$

This is done from EF204 to EF218 but is skipped if F (DMIN) is greater than 0.95.

CODING

EF23	Loop over direct and image fields.
EF29 - EF31	Components of $\bar{\rho}$.
EF33 - EF40	Components of $\bar{\rho}/\rho'$ computed.
EF46 - EF62	Electric field of the segment computed by infinitesimal dipole approximation.
EF68	Field computed by thin wire approximation.
EF70	Field computed by extended thin wire approximation.
EF72 - EF80	Field converted to x, y, and z components.
EF89 - EF111	Reflection coefficients computed.
EF112 - EF129	Image fields modified by reflection coefficients.
EF130 - EF138	Reflected fields added to direct fields.

SYMBOL DICTIONARY

AI	= radius of segment on which field is evaluated
CTH	= cos θ; θ = angle from axis of infinitesimal dipole or angle between the reflecting ray and vertical
EGND	= components of $\bar{E}_S$ (see EQUIVALENCE statement)
EPX, EPY	= x and y components of $(\bar{E} \cdot \hat{p})\hat{p}$ (see PX)
ETA	= $\eta = (\mu_0/\epsilon_0)^{1/2}$
IJ	= IJX = flag to indicate field evaluation point is on the source segment (IJ = 0)
PI	= π

PX, PY = x and y components of unit vector normal to the plane of incidence of the reflected wave ($\hat{\rho}$)

R = distance from field evaluation point to the center of the source segment

REFPS = reflection coefficient for a horizontally polarized field

REFS = reflection coefficient for a vertically polarized field

RFL = +1 for direct field, -1 for reflected field

RH = ρ'

RHOSPC = distance from coordinate origin to the point where the ray from the source to (XI, YI, ZI) reflects from the ground

RHOX, RHOY, RHOZ = x, y, and z components of $\bar{\rho}$ or $\bar{\rho}/\rho'$ or $\hat{j} \times \bar{R}$

RMAG = $2\pi R$ or R or dipole moment for sin ks current

SALPR = z component of unit vector in the direction of the source segment or its image

SHAF = half of segment length

TERC, TERS, TERK = ρ component of field due to cos ks, sin ks, and constant currents, respectively

TEZC, TEZS, TEZK = z component of field due to cos ks, sin ks, and constant current, respectively

TP = 2π

TXC, TYC, TZC, TXS, TYS, TZS, TXK, TYK, TZK = x, y, and z components of field due to cos ks, sin ks, and constant current

XI, YI, ZI = x, y, z coordinates of field evaluation point

XIJ, YIJ, ZIJ = components of distance from source to observation point

XO, YO, ZO = coordinates of field evaluation point for E_S

XSPEC, YSPEC = x, y coordinates of ground plane reflection point

XYMAG = horizontal distance from center of source segment to observation point

ZP = projection of the vector from the source segment (XI, YI, ZI) onto the axis of the source segment

ZRATX = temporary storage for ZRATI

ZRSIN = $(1 - Z_R^2 \sin^2 \theta)^{1/2}$ for ground

ZSCRN = quantity used in computing reflection coefficient for radial wire ground screen

CONSTANT

3.141592654 = π

376.73 = $\eta = \sqrt{\mu_0/\varepsilon_0}$

6.283185308 = 2π

```
      SUBROUTINE EFLD (XI,YI,ZI,AI,IJ)
C
C     COMPUTE NEAR E FIELDS OF A SEGMENT WITH SINE, COSINE, AND
C     CONSTANT CURRENTS.  GROUND EFFECT INCLUDED.
C
      COMPLEX TXK,TYK,TZK,TXS,TYS,TZS,TXC,TYC,TZC,EXK,EYK,EZK,EXS,EYS,EZ
     1S,EXC,EYC,EZC,EPX,EPY,ZRATI,REFS,REFPS,ZRSIN,ZRATX,T1,ZSCRN,ZRATI2
     2,TEZS,TERS,TEZC,TERC,TEZK,TERK,EGND,FRATI
      COMMON /DATAJ/ S,B,XJ,YJ,ZJ,CABJ,SABJ,SALPJ,EXK,EYK,EZK,EXS,EYS,EZ
     1S,EXC,EYC,EZC,RKH,IEXK,IND1,IND2,IPGND
      COMMON /GND/ ZRATI,ZRATI2,FRATI,CL,CH,SCRWL,SCRWR,NRADL,KSYMP,IFAR
     1,IPERF,T1,T2
      COMMON /INCOM/ XO,YO,ZO,SN,XSN,YSN,ISNOR
      DIMENSION EGND(9)
      EQUIVALENCE (EGND(1),TXK), (EGND(2),TYK), (EGND(3),TZK), (EGND(4),
     1TXS), (EGND(5),TYS), (EGND(6),TZS), (EGND(7),TXC), (EGND(8),TYC),
     2(EGND(9),TZC)
      DATA ETA/376.73/,PI/3.141592654/,TP/6.283185308/
      XIJ=XI-XJ
      YIJ=YI-YJ
      IJX=IJ
      RFL=-1.
      DO 12 IP=1,KSYMP
      IF (IP.EQ.2) IJX=1
      RFL=-RFL
      SALPR=SALPJ*RFL
      ZIJ=ZI-RFL*ZJ
      ZP=XIJ*CABJ+YIJ*SABJ+ZIJ*SALPR
      RHOX=XIJ-CABJ*ZP
      RHOY=YIJ-SABJ*ZP
      RHOZ=ZIJ-SALPR*ZP
      RH=SQRT(RHOX*RHOX+RHOY*RHOY+RHOZ*RHOZ+AI*AI)
      IF (RH.GT.1.E-10) GO TO 1
      RHOX=0.
      RHOY=0.
      RHOZ=0.
      GO TO 2
1     RHOX=RHOX/RH
      RHOY=RHOY/RH
      RHOZ=RHOZ/RH
2     R=SQRT(ZP*ZP+RH*RH)
      IF (R.LT.RKH) GO TO 3
C
C     LUMPED CURRENT ELEMENT APPROX. FOR LARGE SEPARATIONS
C
      RMAG=TP*R
      CTH=ZP/R
      PX=RH/R
      TXK=CMPLX(COS(RMAG),-SIN(RMAG))
      PY=TP*R*R
      TYK=ETA*CTH*TXK*CMPLX(1.,-1./RMAG)/PY
      TZK=ETA*PX*TXK*CMPLX(1.,RMAG-1./RMAG)/(2.*PY)
      TEZK=TYK*CTH-TZK*PX
      TERK=TYK*PX+TZK*CTH
      RMAG=SIN(PI*S)/PI
      TEZC=TEZK*RMAG
      TERC=TERK*RMAG
      TEZK=TEZK*S
      TERK=TERK*S
      TXS=(0.,0.)
      TYS=(0.,0.)
      TZS=(0.,0.)
      GO TO 6
3     IF (IEXK.EQ.1) GO TO 4
```

```
C
C     EKSC FOR THIN WIRE APPROX. OR EKSCX FOR EXTENDED T.W. APPROX.
C
      CALL EKSC (S,ZP,RH,TP,IJX,TEZS,TERS,TEZC,TERC,TEZK,TERK)
      GO TO 5
4     CALL EKSCX (B,S,ZP,RH,TP,IJX,IND1,IND2,TEZS,TERS,TEZC,TERC,TEZK,TE
     1RK)
5     TXS=TEZS*CABJ+TERS*RHOX
      TYS=TEZS*SABJ+TERS*RHOY
      TZS=TEZS*SALPR+TERS*RHOZ
6     TXK=TEZK*CABJ+TERK*RHOX
      TYK=TEZK*SABJ+TERK*RHOY
      TZK=TEZK*SALPR+TERK*RHOZ
      TXC=TEZC*CABJ+TERC*RHOX
      TYC=TEZC*SABJ+TERC*RHOY
      TZC=TEZC*SALPR+TERC*RHOZ
      IF (IP.NE.2) GO TO 11
      IF (IPERF.GT.0) GO TO 10
      ZRATX=ZRATI
      RMAG=R
      XYMAG=SQRT(XIJ*XIJ+YIJ*YIJ)
C
C     SET PARAMETERS FOR RADIAL WIRE GROUND SCREEN.
C
      IF (NRADL.EQ.0) GO TO 7
      XSPEC=(XI*ZJ+ZI*XJ)/(ZI+ZJ)
      YSPEC=(YI*ZJ+ZI*YJ)/(ZI+ZJ)
      RHOSPC=SQRT(XSPEC*XSPEC+YSPEC*YSPEC+T2*T2)
      IF (RHOSPC.GT.SCRWL) GO TO 7
      ZSCRN=T1*RHOSPC*ALOG(RHOSPC/T2)
      ZRATX=(ZSCRN*ZRATI)/(ETA*ZRATI+ZSCRN)
7     IF (XYMAG.GT.1.E-6) GO TO 8
C
C     CALCULATION OF REFLECTION COEFFICIENTS WHEN GROUND IS SPECIFIED.
C
      PX=0.
      PY=0.
      CTH=1.
      ZRSIN=(1.,0.)
      GO TO 9
8     PX=-YIJ/XYMAG
      PY=XIJ/XYMAG
      CTH=ZIJ/RMAG
      ZRSIN=CSQRT(1.-ZRATX*ZRATX*(1.-CTH*CTH))
9     REFS=(CTH-ZRATX*ZRSIN)/(CTH+ZRATX*ZRSIN)
      REFPS=-(ZRATX*CTH-ZRSIN)/(ZRATX*CTH+ZRSIN)
      REFPS=REFPS-REFS
      EPY=PX*TXK+PY*TYK
      EPX=PX*EPY
      EPY=PY*EPY
      TXK=REFS*TXK+REFPS*EPX
      TYK=REFS*TYK+REFPS*EPY
      TZK=REFS*TZK
      EPY=PX*TXS+PY*TYS
      EPX=PX*EPY
      EPY=PY*EPY
      TXS=REFS*TXS+REFPS*EPX
      TYS=REFS*TYS+REFPS*EPY
      TZS=REFS*TZS
      EPY=PX*TXC+PY*TYC
      EPX=PX*EPY
      EPY=PY*EPY
      TXC=REFS*TXC+REFPS*EPX
      TYC=REFS*TYC+REFPS*EPY
```

```
      TZC=REFS*TZC
10    EXK=EXK-TXK*FRATI
      EYK=EYK-TYK*FRATI
      EZK=EZK-TZK*FRATI
      EXS=EXS-TXS*FRATI
      EYS=EYS-TYS*FRATI
      EZS=EZS-TZS*FRATI
      EXC=EXC-TXC*FRATI
      EYC=EYC-TYC*FRATI
      EZC=EZC-TZC*FRATI
      GO TO 12
11    EXK=TXK
      EYK=TYK
      EZK=TZK
      EXS=TXS
      EYS=TYS
      EZS=TZS
      EXC=TXC
      EYC=TYC
      EZC=TZC
12    CONTINUE
      IF (IPERF.EQ.2) GO TO 13
      RETURN
C
C     FIELD DUE TO GROUND USING SOMMERFELD/NORTON
C
13    SN=SQRT(CABJ*CABJ+SABJ*SABJ)
      IF (SN.LT.1.E-5) GO TO 14
      XSN=CABJ/SN
      YSN=SABJ/SN
      GO TO 15
14    SN=0.
      XSN=1.
      YSN=0.
C
C     DISPLACE OBSERVATION POINT FOR THIN WIRE APPROXIMATION
C
15    ZIJ=ZI+ZJ
      SALPR=-SALPJ
      RHOX=SABJ*ZIJ-SALPR*YIJ
      RHOY=SALPR*XIJ-CABJ*ZIJ
      RHOZ=CABJ*YIJ-SABJ*XIJ
      RH=RHOX*RHOX+RHOY*RHOY+RHOZ*RHOZ
      IF (RH.GT.1.E-10) GO TO 16
      XO=XI-AI*YSN
      YO=YI+AI*XSN
      ZO=ZI
      GO TO 17
16    RH=AI/SQRT(RH)
      IF (RHOZ.LT.0.) RH=-RH
      XO=XI+RH*RHOX
      YO=YI+RH*RHOY
      ZO=ZI+RH*RHOZ
17    R=XIJ*XIJ+YIJ*YIJ+ZIJ*ZIJ
      IF (R.GT..95) GO TO 18
C
C     FIELD FROM INTERPOLATION IS INTEGRATED OVER SEGMENT
C
      ISNOR=1
      DMIN=EXK*CONJG(EXK)+EYK*CONJG(EYK)+EZK*CONJG(EZK)
      DMIN=.01*SQRT(DMIN)
      SHAF=.5*S
      CALL ROM2 (-SHAF,SHAF,EGND,DMIN)
      GO TO 19
```

```
C
C     NORTON FIELD EQUATIONS AND LUMPED CURRENT ELEMENT APPROXIMATION
C
   18 ISNOR=2
      CALL SFLDS (0.,EGND)
      GO TO 22
   19 ZP=XIJ*CABJ+YIJ*SABJ+ZIJ*SALPR
      RH=R-ZP*ZP
      IF (RH.GT.1.E-10) GO TO 20
      DMIN=0.
      GO TO 21
   20 DMIN=SQRT(RH/(RH+AI*AI))
   21 IF (DMIN.GT..95) GO TO 22
      PX=1.-DMIN
      TERK=(TXK*CABJ+TYK*SABJ+TZK*SALPR)*PX
      TXK=DMIN*TXK+TERK*CABJ
      TYK=DMIN*TYK+TERK*SABJ
      TZK=DMIN*TZK+TERK*SALPR
      TERS=(TXS*CABJ+TYS*SABJ+TZS*SALPR)*PX
      TXS=DMIN*TXS+TERS*CABJ
      TYS=DMIN*TYS+TERS*SABJ
      TZS=DMIN*TZS+TERS*SALPR
      TERC=(TXC*CABJ+TYC*SABJ+TZC*SALPR)*PX
      TXC=DMIN*TXC+TERC*CABJ
      TYC=DMIN*TYC+TERC*SABJ
      TZC=DMIN*TZC+TERC*SALPR
   22 EXK=EXK+TXK
      EYK=EYK+TYK
      EZK=EZK+TZK
      EXS=EXS+TXS
      EYS=EYS+TYS
      EZS=EZS+TZS
      EXC=EXC+TXC
      EYC=EYC+TYC
      EZC=EZC+TZC
      RETURN
      END
```

EKSC

PURPOSE

To compute the electric field due to current filaments with sin kz, cos kz and constant distributions.

METHOD

Equations 71 through 74 in Part I are used. The current filament is located at the origin of a cylindrical coordinate system, oriented along the z axis, and extending from $-\Delta/2$ to $\Delta/2$. The field is computed in ρ and z components.

SYMBOL DICTIONARY

CINT = $\int_{-\Delta/2}^{\Delta/2} \cos(kr)/r \, dz$

CON = CONX = $j\eta/(8\pi^2)$, $\eta = \sqrt{\mu_0/\varepsilon_0}$

CS = $\cos(k\Delta/2)$

ERS, EZS, ERC, EZC, ERK, EZK = ρ and z components of field due to sin kz, cos kz, and constant (S, C, K, respectively) current distributions extending from $z = -\Delta/2$ to $z = \Delta/2$

GP1, GP2 = $-(1 + jkr)\, G_0/r^2$ for $z = -\Delta/2$ and $\Delta/2$, respectively, where $G_0 = \exp(-jkr)/r$

GZ1, GZ2 = G_0 for $z = -\Delta/2$ and $\Delta/2$, respectively

GZP1, GZP2 = $\partial G_0/\partial z$ at EK21, EK22 and $\partial G_0/\partial \rho$ at EK28, EK29 for $z = -\Delta/2$ and $\Delta/2$, respectively

IJ = IJX = 0 to indicate that the field point is on the source segment

RH = ρ coordinate of field point

RHK = $k\rho$ ($k = 2\pi/\lambda$, $\lambda = 1$)

RKB2 = $(k\rho)^2$

S = Δ

SH = $\Delta/2$

SHK = $k\Delta/2$

$$SINT = \int_{-\Delta/2}^{\Delta/2} \sin(kr)/r \, dz$$

SS = $\sin(k\Delta/2)$

XK = $k = 2\pi/\lambda$, where $\lambda = 1$

Z = z coordinate of field point

Z1 = $-\Delta/2 - z$

Z2 = $\Delta/2 - z$

ZPK = kz

CONSTANT

$4.771341189 = \eta/(8\pi^2)$

CODE LISTING

```
      SUBROUTINE EKSC (S,Z,RH,XK,IJ,EZS,ERS,EZC,ERC,EZK,ERK)             EK   1
C     COMPUTE E FIELD OF SINE, COSINE, AND CONSTANT CURRENT FILAMENTS BY EK   2
C     THIN WIRE APPROXIMATION.                                           EK   3
      COMPLEX CON,GZ1,GZ2,GP1,GP2,GZP1,GZP2,EZS,ERS,EZC,ERC,EZK,ERK      EK   4
      COMMON /TMI/ ZPK,RKB2,IJX                                          EK   5
      DIMENSION CONX(2)                                                  EK   6
      EQUIVALENCE (CONX,CON)                                             EK   7
      DATA CONX/0.,4.771341189/                                          EK   8
      IJX=IJ                                                             EK   9
      ZPK=XK*Z                                                           EK  10
      RHK=XK*RH                                                          EK  11
      RKB2=RHK*RHK                                                       EK  12
      SH=.5*S                                                            EK  13
      SHK=XK*SH                                                          EK  14
      SS=SIN(SHK)                                                        EK  15
      CS=COS(SHK)                                                        EK  16
      Z2=SH-Z                                                            EK  17
      Z1=-(SH+Z)                                                         EK  18
      CALL GX (Z1,RH,XK,GZ1,GP1)                                         EK  19
      CALL GX (Z2,RH,XK,GZ2,GP2)                                         EK  20
      GZP1=GP1*Z1                                                        EK  21
      GZP2=GP2*Z2                                                        EK  22
      EZS=CON*((GZ2-GZ1)*CS*XK-(GZP2+GZP1)*SS)                           EK  23
      EZC=-CON*((GZ2+GZ1)*SS*XK+(GZP2-GZP1)*CS)                          EK  24
      ERK=CON*(GP2-GP1)*RH                                               EK  25
      CALL INTX (-SHK,SHK,RHK,IJ,CINT,SINT)                              EK  26
      EZK=-CON*(GZP2-GZP1+XK*XK*CMPLX(CINT,-SINT))                       EK  27
      GZP1=GZP1*Z1                                                       EK  28
      GZP2=GZP2*Z2                                                       EK  29
      IF (RH.LT.1.E-10) GO TO 1                                          EK  30
      ERS=-CON*((GZP2+GZP1+GZ2+GZ1)*SS-(Z2*GZ2-Z1*GZ1)*CS*XK)/RH         EK  31
      ERC=-CON*((GZP2-GZP1+GZ2-GZ1)*CS+(Z2*GZ2+Z1*GZ1)*SS*XK)/RH         EK  32
      RETURN                                                             EK  33
    1 ERS=(0.,0.)                                                        EK  34
      ERC=(0.,0.)                                                        EK  35
      RETURN                                                             EK  36
      END                                                                EK  37-
```

EKSCX

PURPOSE

To compute the electric field due to current distributions of sin kz, cos kz, and constant on the surface of a cylinder by the extended thin wire approximation.

METHOD

Equations 84 through 87 in Part I are used. The current tube is centered on the origin of a cylindrical coordinate system, oriented along the z axis and extending from $-\Delta/2$ to $\Delta/2$. The field is computed in ρ and z components.

If INX1 = 2, the field contributions from end 1 of the segment ($z = -\Delta/2$) are evaluated by the thin wire approximation for a current filament on the cylinder axis. INX2 has the same meaning for end 2 of the segment ($z = \Delta/2$). The thin-wire approximation is used at an end when there is a bend or change in radius from that end to the next segment.

When the ρ coordinate of the field point (RHX) is less than the radius of the current tube (BX), then RHX and BX are interchanged and a flag, IRA, is set to 1 to cause alternate forms for G_1 and its derivatives to be used in routine GXX.

SYMBOL DICTIONARY

A2 = B^2

B = radius of the current tube

BK = kB, where $k = 2\pi/\lambda$, $\lambda = 1$

BK2 = $(BK)^2/4$

BX = radius of the current tube

CINT = $\int_{-\Delta/2}^{\Delta/2} \cos(kr)/r \, dz$

CON = CONX = $j\eta/(8\pi^2)$, where $\eta = \sqrt{\mu_0/\epsilon_0}$

CS = $\cos(k\Delta/2)$

ERS, EZS, ERC, EZC, ERK, EZK = ρ and z components of field due to sin kz, cos kz, and constant (S, C, K, respectively) current distributions extending from $z = -\Delta/2$ to $z = \Delta/2$.

GR1, GR2	= G_2 for $z = -\Delta/2$ and $\Delta/2$, respectively
GRK1, GRK2	= $\partial G_1/\partial\rho$
GRP1, GRP2	= $\partial G_2/\partial z'$
GZ1, GZ2	= G_1
GZP1, GZP2	= $\partial G_1/\partial z'$
GZZ1, GZZ2	= $\partial G_0/\partial z'$
IJ	= IJX = 0 to indicate that the field point is on the source segment
INX1, INX2	= 2 to use the thin wire form at end 1 or end 2, respectively
IRA	= 1 to indicate RHX < BX
RH	= ρ coordinate of the field point or wire radius
RHK	= k(RH)
RHX	= ρ coordinate of the field point
RKB2	= $(RHK)^2$
S	= Δ
SH	= $\Delta/2$
SHK	= $k\Delta/2$
SINT	= $\int_{-\Delta/2}^{\Delta/2} \sin(kr)/r \, dz$
SS	= $\sin(k\Delta/2)$
XK	= $k = 2\pi/\lambda$, $\lambda = 1$
Z	= z coordinate of field point
Z1	= $-\Delta/2 - z$
Z2	= $\Delta/2 - z$
ZPK	= kz

CONSTANT

4.77134118 = $\eta/(8\pi^2)$

EKSCX

```
      SUBROUTINE EKSCX (BX,S,Z,RHX,XK,IJ,INX1,INX2,EZS,ERS,EZC,ERC,EZK,E EX   1
     1RK)                                                               EX   2
C     COMPUTE E FIELD OF SINE, COSINE, AND CONSTANT CURRENT FILAMENTS BY EX   3
C     EXTENDED THIN WIRE APPROXIMATION.                                  EX   4
      COMPLEX CON,GZ1,GZ2,GZP1,GZP2,GR1,GR2,GRP1,GRP2,EZS,EZC,ERS,ERC,GR EX   5
     1K1,GRK2,EZK,ERK,GZZ1,GZZ2                                         EX   6
      COMMON /TMI/ ZPK,RKB2,IJX                                         EX   7
      DIMENSION CONX(2)                                                 EX   8
      EQUIVALENCE (CONX,CON)                                            EX   9
      DATA CONX/0.,4.771341189/                                         EX  10
      IF (RHX.LT.BX) GO TO 1                                            EX  11
      RH=RHX                                                            EX  12
      B=BX                                                              EX  13
      IRA=0                                                             EX  14
      GO TO 2                                                           EX  15
1     RH=BX                                                             EX  16
      B=RHX                                                             EX  17
      IRA=1                                                             EX  18
2     SH=.5*S                                                           EX  19
      IJX=IJ                                                            EX  20
      ZPK=XK*Z                                                          EX  21
      RHK=XK*RH                                                         EX  22
      RKB2=RHK*RHK                                                      EX  23
      SHK=XK*SH                                                         EX  24
      SS=SIN(SHK)                                                       EX  25
      CS=COS(SHK)                                                       EX  26
      Z2=SH-Z                                                           EX  27
      Z1=-(SH+Z)                                                        EX  28
      A2=B*B                                                            EX  29
      IF (INX1.EQ.2) GO TO 3                                            EX  30
      CALL GXX (Z1,RH,B,A2,XK,IRA,GZ1,GZP1,GR1,GRP1,GRK1,GZZ1)          EX  31
      GO TO 4                                                           EX  32
3     CALL GX (Z1,RHX,XK,GZ1,GRK1)                                      EX  33
      GZP1=GRK1*Z1                                                      EX  34
      GR1=GZ1/RHX                                                       EX  35
      GRP1=GZP1/RHX                                                     EX  36
      GRK1=GRK1*RHX                                                     EX  37
      GZZ1=(0.,0.)                                                      EX  38
4     IF (INX2.EQ.2) GO TO 5                                            EX  39
      CALL GXX (Z2,RH,B,A2,XK,IRA,GZ2,GZP2,GR2,GRP2,GRK2,GZZ2)          EX  40
      GO TO 6                                                           EX  41
5     CALL GX (Z2,RHX,XK,GZ2,GRK2)                                      EX  42
      GZP2=GRK2*Z2                                                      EX  43
      GR2=GZ2/RHX                                                       EX  44
      GRP2=GZP2/RHX                                                     EX  45
      GRK2=GRK2*RHX                                                     EX  46
      GZZ2=(0.,0.)                                                      EX  47
6     EZS=CON*((GZ2-GZ1)*CS*XK-(GZP2+GZP1)*SS)                          EX  48
      EZC=-CON*((GZ2+GZ1)*SS*XK+(GZP2-GZP1)*CS)                         EX  49
      ERS=-CON*((Z2*GRP2+Z1*GRP1+GR2+GR1)*SS-(Z2*GR2-Z1*GR1)*CS*XK)     EX  50
      ERC=-CON*((Z2*GRP2-Z1*GRP1+GR2-GR1)*CS+(Z2*GR2+Z1*GR1)*SS*XK)     EX  51
      ERK=CON*(GRK2-GRK1)                                               EX  52
      CALL INTX (-SHK,SHK,RHK,IJ,CINT,SINT)                             EX  53
      BK=B*XK                                                           EX  54
      BK2=BK*BK*.25                                                     EX  55
      EZK=-CON*(GZP2-GZP1+XK*XK*(1.-BK2)*CMPLX(CINT,-SINT)-BK2*(GZZ2-GZZ EX  56
     11))                                                               EX  57
      RETURN                                                            EX  58
      END                                                               EX  59-
```

ENF

PURPOSE

To check for an end of file.

METHOD

ENF uses the standard Fortran end-of-file test and returns the logical values .TRUE. or .FALSE. This separate function is used for convenience in adapting the code to particular computers, since the Fortran end-of-file test statements often differ between computers. The form of ENF here is for CDC computers.

SYMBOL DICTIONARY

ENF = logical value: .TRUE. if end of file was encountered; .FALSE. otherwise

NUNIT = logical unit number

CODE LISTING

```
      LOGICAL FUNCTION ENF(NUNIT)
      IF (EOF,NUNIT) 1,2
1     ENF=.TRUE.
      RETURN
2     ENF=.FALSE.
      RETURN
      END
```

ETMNS

PURPOSE

To fill the array representing the right-hand side of the matrix equation with the negative of the electric field tangent to the segments and with the tangential magnetic field on the surfaces.

METHOD

The array E represents the right-hand side of the matrix equation. For the i^{th} segment, the right-hand side is the negative of the applied electric field component tangent to the segment, and is stored in location i in array E. For the i^{th} surface patch, there are two rows in the matrix equation (from the two components of the vector equations) with locations N + 2i - 1 and N + 2i, where N is the total number of wire segments. The contents of E for these locations are

$$E\ (N + 2i - 1) = -\hat{t}_1 \cdot (\hat{n} \times \overline{H}_i) = \pm t_2 \cdot \overline{H}_i$$

$$E\ (N + 2i) = \hat{t}_2 \cdot (\hat{n} \times \overline{H}_i) = \pm t_1 \cdot \overline{H}_i$$

where $\overline{H}_i$ is the magnetic field applied to patch i. The forms on the right are used in the code with the plus sign applying when $(\hat{t}_1, \hat{t}_2, \hat{n})$ forms a right-hand system and the minus sign when left-hand. To avoid the need to check $(\hat{t}_1, \hat{t}_2, \hat{n})$, the sign is stored in array SALP where, for patch i, SALP (LD + 1 - i) = ±1 according to $(\hat{t}_1, \hat{t}_2, \hat{n})$, with LD the length of the arrays in COMMON/DATA/. If the structure has symmetry, the entries in E are reordered by subroutine SOLVES.

The parameter IPR selects the type of excitation; the meanings of other parameters depend on the option selected by IPR and are explained below. The excitations associated with IPR values are:

IPR = 0 applied field voltage source
1 incident plane wave, linear polarization
2 incident plane wave, right-hand elliptic polarization
3 incident plane wave, left-hand elliptic polarization
4 infinitesimal current element source
5 current slope discontinuity voltage source

CODING

ET29 - ET34 Applied field voltage source (IPR = 0).

ET36 - ET38 QDSRC is called for each current slope discontinuity voltage source (IPR = 5).

ET44 - ET160 Incident plane wave. The direction of propagation and polarization of the wave are illustrated in figure 4 in which $\hat{p}$ is the unit vector normal to $\hat{k}$ in the plane defined by $\hat{k}$ and $\hat{z}$. The plane wave as a function of position $\bar{r}$ is

$$\bar{E}^I(\bar{r}) = \bar{E}_0 \exp(-j\bar{k}\cdot\bar{r})$$

$$\bar{H}^I(\bar{r}) = \frac{1}{\eta}\hat{k} \times \bar{E}_0 \exp(-j\bar{k}\cdot\bar{r})$$

where

$\bar{k} = (2\pi/\lambda)\,\hat{k}$

$\hat{k}$ = unit vector in direction of propagation

$\bar{E}_0 = \hat{E}_1$ for linear polarization

$= (\hat{E}_1 - jA\hat{E}_2)$ for right-hand elliptical polarization

$= (\hat{E}_1 + jA\hat{E}_2)$ for left-hand elliptical polarization

A = ellipse axes ratio

$\hat{E}_2 = \hat{k} \times \hat{E}_1$

ET44 - ET58 P1 = θ

P2 = ϕ

P3 = ξ

PX, PY, PZ = x, y, z components of $\hat{E}_1$

WX, WY, WZ = $\hat{k}$

QX, QY, QZ = $\hat{E}_2 = \hat{k} \times \hat{E}_1$

ET61 - ET68 Ground reflection coefficients computed:

RRH = reflection coefficient for E normal to the plane of incidence

RRV = reflection coefficient for E in the plane of incidence

ET70 - ET108 Linearly polarized wave (IPR = 1).

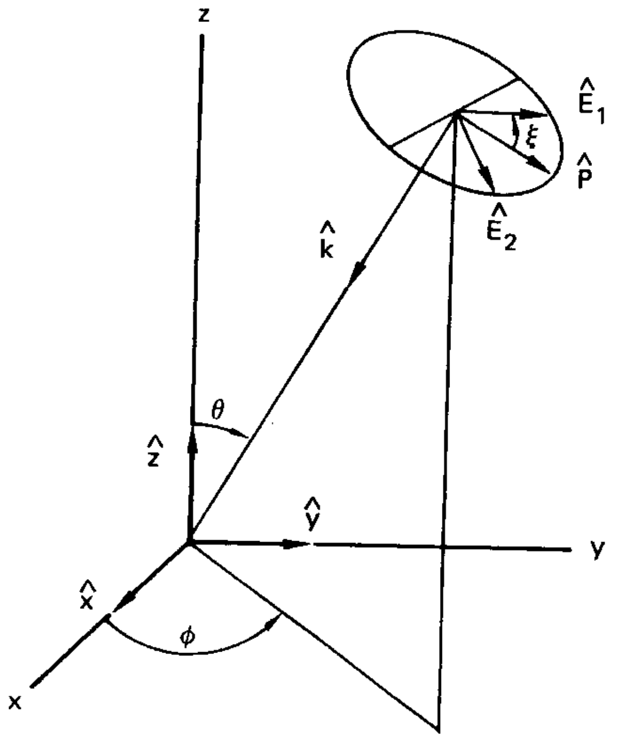


Figure 4. Coordinate Parameters for the Incident Plane Wave.

ET71 - ET73 Direct illumination of segments by E field. ARG = $-\bar{k} \cdot \bar{r}_i$, where $\bar{r}_i$ = center point of segment I. E(I) = $-(\hat{E}_1 \cdot \hat{i})$ $\exp(-j\bar{k} \cdot \bar{r}_i)$, where $\hat{i}$ = unit vector in the direction of segment I.

ET75 - ET82 Illumination of segments by the ground reflected field. CX, CY, CZ = reflected E field

ET84 - ET93 Direct H field illumination of patches.

ET95 - ET108 Illumination of patches by the ground reflected field. CX, CY, CZ = reflected H field

ET113 - ET159 Elliptically polarized wave (IPR = 2 or 3). P6 = ellipse axes ratio = A.

ET116 - ET121 Direct E field illumination of segments. CX, CY, CZ = $\hat{E}_1 \pm jA\hat{E}_2$ (+ for left-hand polarization, - for right-hand)

ET123 - ET130 Illumination of segments by the ground reflected E field.

ET132 - ET144 Illumination of patches by the direct H field. CX, CY, CZ = $\hat{k} \times \bar{E}_0$

ET146 - ET159 Illumination of patches by ground reflected H field.

ET164 - ET225 Infinitesimal current element source (IPR = 4). A current element of moment $I_0 \ell$ at the origin of a spherical coordinate system, as shown in figure 5, produces field components

$$\bar{E}_R(\bar{R}) = I_0 \ell \frac{\eta}{2\pi} \exp(-jkR) \left(1 - \frac{j}{kR}\right) \frac{1}{R^2} \cos\theta \, \hat{R}$$

$$\bar{E}_\theta(\bar{R}) = I_0 \ell \frac{\eta}{4\pi} \exp(-jkR) \left[\frac{jk}{R} + \left(1 - \frac{j}{kR}\right) \frac{1}{R^2}\right] \sin\theta \, \hat{\theta}$$

$$H_\phi = \frac{I_0 \ell}{4\pi} \exp(-jkR) \left(\frac{1}{R^2} + \frac{jk}{R}\right) \sin\theta$$

If the location and orientation of segment i and the current element with respect to the x, y, z coordinate system are

$\bar{r}_i$ = location of segment i

$\hat{i}$ = orientation of segment i

$\bar{D}$ = location of current element

$\hat{d}$ = orientation of current element

then

$\bar{R} = \bar{r}_i - \bar{D}$

$\hat{R} = \bar{R}/|\bar{R}|$

$\cos\theta = \hat{R} \cdot \hat{d}$

$\sin\theta = [1 - \cos^2\theta]^{1/2}$

The orientation of the current element is defined by its angle of elevation above the x-y plane, a, and the angle from the x axis to its projection on the x-y plane, b. Thus, $\hat{d} = \cos a \cos b \, \hat{x} + \cos a \sin b \, \hat{y} + \sin a \, \hat{z}$. The $\hat{R}$ and $\hat{\theta}$ field components are converted to $\hat{\rho}$ and $\hat{d}$ components E_ρ and E_d, where

$$E_d = E_R \cos\theta - E_\theta \sin\theta$$

$$E_\rho = E_R \sin\theta + E_\theta \cos\theta$$

and the excitation computed as

$$E(I) = -\hat{i} \cdot (E_d \hat{d} + E_\rho \hat{\rho}) .$$

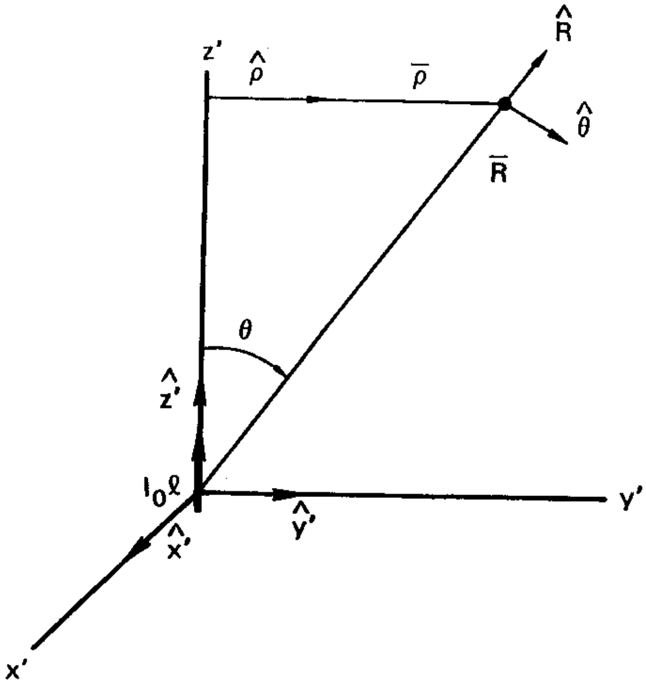


Figure 5. Coordinate Parameters for Current Element.

ET164 - ET225 P1, P2, P3 = x, y, z coordinates of current element ($\bar{D}$)

P4 = a

P5 = b

P6 = $I_0\ell/\lambda^2$

ET164 - ET169 WX, WY, WZ = x, y, and z components of $\hat{d}$

DS = $(\eta/2\pi)\ I_0\ell/\lambda^2$

DSH = $(1/4\pi)\ I_0\ell/\lambda^2$

ET173 Start of loop over all segments and patches.

ET176 - ET179 For patches,

IS = location of patch data in geometry arrays

I1, I2 = locations to be filled in E

ET180 - ET182 PX, PY, PZ = $\bar{R}/\lambda$

ET183 - ET193 R = $|\bar{R}/\lambda|$

PX, PY, PZ = $\hat{R}$

CTH = $\cos\theta$

STH = $\sin\theta$

QX, QY, QZ = $\hat{R} - (\hat{d}\cdot\hat{R})\hat{d}$

ET196 - ET204 QX, QY, QZ = $\hat{\rho}$

T1 = exp(-jk R)

ET206 - ET215 E field on segments

T2 = $(1 - j/kR)\lambda^2/R^2$

ER = E_R

ET = E_θ

ERH = E_ρ

EZH = E_z

CX, CY, CZ = x, y, z components of total E field

ET216 - ET224 H field on patches

PX, PY, PZ = $\hat{d} \times \hat{\rho} = \hat{\phi}$

T2 = $\pm H_\phi$

CX, CY, CZ = $\pm \bar{H}^I$

CONSTANTS

1.E-30 = tolerance in test for zero

2.654420938E-3 = $1/\eta = \sqrt{\varepsilon_0/\mu_0}$

59.958 = $\eta/2\pi$

6.283185308 = 2π

```
      SUBROUTINE ETMNS (P1,P2,P3,P4,P5,P6,IPR,E)
C
C     ETMNS FILLS THE ARRAY E WITH THE NEGATIVE OF THE ELECTRIC FIELD
C     INCIDENT ON THE STRUCTURE.  E IS THE RIGHT HAND SIDE OF THE MATRIX
C     EQUATION.
C
      COMPLEX E,CX,CY,CZ,VSANT,TX1,TX2,ER,ET,EZH,ERH,VQD,VQDS,ZRATI,ZRAT
     1I2,RRV,RRH,T1,TT1,TT2,FRATI
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /ANGL/ SALP(300)
      COMMON /VSORC/ VQD(30),VSANT(30),VQDS(30),IVQD(30),ISANT(30),IQDS(
     130),NVQD,NSANT,NQDS
      COMMON /GND/ZRATI,ZRATI2,FRATI,CL,CH,SCRWL,SCRWR,NRADL,KSYMP,IFAR,
     1IPERF,T1,T2
      DIMENSION CAB(1), SAB(1), E(600)
      DIMENSION T1X(1), T1Y(1), T1Z(1), T2X(1), T2Y(1), T2Z(1)
      EQUIVALENCE (CAB,ALP), (SAB,BET)
      EQUIVALENCE (T1X,SI), (T1Y,ALP), (T1Z,BET), (T2X,ICON1), (T2Y,ICON
     12), (T2Z,ITAG)
      DATA TP/6.283185308/,RETA/2.654420938E-3/
      NEQ=N+2*M
      NQDS=0
      IF (IPR.GT.0.AND.IPR.NE.5) GO TO 5
C
C     APPLIED FIELD OF VOLTAGE SOURCES FOR TRANSMITTING CASE
C
      DO 1 I=1,NEQ
1     E(I)=(0.,0.)
      IF (NSANT.EQ.0) GO TO 3
      DO 2 I=1,NSANT
      IS=ISANT(I)
2     E(IS)=-VSANT(I)/(SI(IS)*WLAM)
3     IF (NVQD.EQ.0) RETURN
      DO 4 I=1,NVQD
      IS=IVQD(I)
4     CALL QDSRC (IS,VQD(I),E)
      RETURN
5     IF (IPR.GT.3) GO TO 19
C
C     INCIDENT PLANE WAVE, LINEARLY POLARIZED.
C
      CTH=COS(P1)
      STH=SIN(P1)
      CPH=COS(P2)
      SPH=SIN(P2)
      CET=COS(P3)
      SET=SIN(P3)
      PX=CTH*CPH*CET-SPH*SET
      PY=CTH*SPH*CET+CPH*SET
      PZ=-STH*CET
      WX=-STH*CPH
      WY=-STH*SPH
      WZ=-CTH
      QX=WY*PZ-WZ*PY
      QY=WZ*PX-WX*PZ
      QZ=WX*PY-WY*PX
      IF (KSYMP.EQ.1) GO TO 7
      IF (IPERF.EQ.1) GO TO 6
      RRV=CSQRT(1.-ZRATI*ZRATI*STH*STH)
      RRH=ZRATI*CTH
      RRH=(RRH-RRV)/(RRH+RRV)
      RRV=ZRATI*RRV
```

```
      RRV=-(CTH-RRV)/(CTH+RRV)
      GO TO 7
    6 RRV=-(1.,0.)
      RRH=-(1.,0.)
    7 IF (IPR.GT.1) GO TO 13
      IF (N.EQ.0) GO TO 10
      DO 8 I=1,N
      ARG=-TP*(WX*X(I)+WY*Y(I)+WZ*Z(I))
    8 E(I)=-(PX*CAB(I)+PY*SAB(I)+PZ*SALP(I))*CMPLX(COS(ARG),SIN(ARG))
      IF (KSYMP.EQ.1) GO TO 10
      TT1=(PY*CPH-PX*SPH)*(RRH-RRV)
      CX=RRV*PX-TT1*SPH
      CY=RRV*PY+TT1*CPH
      CZ=-RRV*PZ
      DO 9 I=1,N
      ARG=-TP*(WX*X(I)+WY*Y(I)-WZ*Z(I))
    9 E(I)=E(I)-(CX*CAB(I)+CY*SAB(I)+CZ*SALP(I))*CMPLX(COS(ARG),SIN(ARG
     1)
   10 IF (M.EQ.0) RETURN
      I=LD+1
      I1=N-1
      DO 11 IS=1,M
      I=I-1
      I1=I1+2
      I2=I1+1
      ARG=-TP*(WX*X(I)+WY*Y(I)+WZ*Z(I))
      TT1=CMPLX(COS(ARG),SIN(ARG))*SALP(I)*RETA
      E(I2)=(QX*T1X(I)+QY*T1Y(I)+QZ*T1Z(I))*TT1
   11 E(I1)=(QX*T2X(I)+QY*T2Y(I)+QZ*T2Z(I))*TT1
      IF (KSYMP.EQ.1) RETURN
      TT1=(QY*CPH-QX*SPH)*(RRV-RRH)
      CX=-(RRH*QX-TT1*SPH)
      CY=-(RRH*QY+TT1*CPH)
      CZ=RRH*QZ
      I=LD+1
      I1=N-1
      DO 12 IS=1,M
      I=I-1
      I1=I1+2
      I2=I1+1
      ARG=-TP*(WX*X(I)+WY*Y(I)-WZ*Z(I))
      TT1=CMPLX(COS(ARG),SIN(ARG))*SALP(I)*RETA
      E(I2)=E(I2)+(CX*T1X(I)+CY*T1Y(I)+CZ*T1Z(I))*TT1
   12 E(I1)=E(I1)+(CX*T2X(I)+CY*T2Y(I)+CZ*T2Z(I))*TT1
      RETURN
C
C     INCIDENT PLANE WAVE, ELLIPTIC POLARIZATION.
C
   13 TT1=-(0.,1.)*P6
      IF (IPR.EQ.3) TT1=-TT1
      IF (N.EQ.0) GO TO 16
      CX=PX+TT1*QX
      CY=PY+TT1*QY
      CZ=PZ+TT1*QZ
      DO 14 I=1,N
      ARG=-TP*(WX*X(I)+WY*Y(I)+WZ*Z(I))
   14 E(I)=-(CX*CAB(I)+CY*SAB(I)+CZ*SALP(I))*CMPLX(COS(ARG),SIN(ARG))
      IF (KSYMP.EQ.1) GO TO 16
      TT2=(CY*CPH-CX*SPH)*(RRH-RRV)
      CX=RRV*CX-TT2*SPH
      CY=RRV*CY+TT2*CPH
      CZ=-RRV*CZ
      DO 15 I=1,N
      ARG=-TP*(WX*X(I)+WY*Y(I)-WZ*Z(I))
```

```
   15 E(I)=E(I)-(CX*CAB(I)+CY*SAB(I)+CZ*SALP(I))*CMPLX(COS(ARG),SIN(ARG)
     1)
   16 IF (M.EQ.0) RETURN
      CX=QX-TT1*PX
      CY=QY-TT1*PY
      CZ=QZ-TT1*PZ
      I=LD+1
      I1=N-1
      DO 17 IS=1,M
      I=I-1
      I1=I1+2
      I2=I1+1
      ARG=-TP*(WX*X(I)+WY*Y(I)+WZ*Z(I))
      TT2=CMPLX(COS(ARG),SIN(ARG))*SALP(I)*RETA
      E(I2)=(CX*T1X(I)+CY*T1Y(I)+CZ*T1Z(I))*TT2
   17 E(I1)=(CX*T2X(I)+CY*T2Y(I)+CZ*T2Z(I))*TT2
      IF (KSYMP.EQ.1) RETURN
      TT1=(CY*CPH-CX*SPH)*(RRV-RRH)
      CX=-(RRH*CX-TT1*SPH)
      CY=-(RRH*CY+TT1*CPH)
      CZ=RRH*CZ
      I=LD+1
      I1=N-1
      DO 18 IS=1,M
      I=I-1
      I1=I1+2
      I2=I1+1
      ARG=-TP*(WX*X(I)+WY*Y(I)-WZ*Z(I))
      TT1=CMPLX(COS(ARG),SIN(ARG))*SALP(I)*RETA
      E(I2)=E(I2)+(CX*T1X(I)+CY*T1Y(I)+CZ*T1Z(I))*TT1
   18 E(I1)=E(I1)+(CX*T2X(I)+CY*T2Y(I)+CZ*T2Z(I))*TT1
      RETURN
C
C     INCIDENT FIELD OF AN ELEMENTARY CURRENT SOURCE.
C
   19 WZ=COS(P4)
      WX=WZ*COS(P5)
      WY=WZ*SIN(P5)
      WZ=SIN(P4)
      DS=P6*59.958
      DSH=P6/(2.*TP)
      NPM=N+M
      IS=LD+1
      I1=N-1
      DO 24 I=1,NPM
      II=I
      IF (I.LE.N) GO TO 20
      IS=IS-1
      II=IS
      I1=I1+2
      I2=I1+1
   20 PX=X(II)-P1
      PY=Y(II)-P2
      PZ=Z(II)-P3
      RS=PX*PX+PY*PY+PZ*PZ
      IF (RS.LT.1.E-30) GO TO 24
      R=SQRT(RS)
      PX=PX/R
      PY=PY/R
      PZ=PZ/R
      CTH=PX*WX+PY*WY+PZ*WZ
      STH=SQRT(1.-CTH*CTH)
      QX=PX-WX*CTH
      QY=PY-WY*CTH
```

```
      QZ=PZ-WZ*CTH                                            ET 193
      ARG=SQRT(QX*QX+QY*QY+QZ*QZ)                             ET 194
      IF (ARG.LT.1.E-30) GO TO 21                             ET 195
      QX=QX/ARG                                               ET 196
      QY=QY/ARG                                               ET 197
      QZ=QZ/ARG                                               ET 198
      GO TO 22                                                ET 199
21    QX=1.                                                   ET 200
      QY=0.                                                   ET 201
      QZ=0.                                                   ET 202
22    ARG=-TP*R                                               ET 203
      TT1=CMPLX(COS(ARG),SIN(ARG))                            ET 204
      IF (I.GT.N) GO TO 23                                    ET 205
      TT2=CMPLX(1.,-1./(R*TP))/RS                             ET 206
      ER=DS*TT1*TT2*CTH                                       ET 207
      ET=.5*DS*TT1*((0.,1.)*TP/R+TT2)*STH                     ET 208
      EZH=ER*CTH-ET*STH                                       ET 209
      ERH=ER*STH+ET*CTH                                       ET 210
      CX=EZH*WX+ERH*QX                                        ET 211
      CY=EZH*WY+ERH*QY                                        ET 212
      CZ=EZH*WZ+ERH*QZ                                        ET 213
      E(I)=-(CX*CAB(I)+CY*SAB(I)+CZ*SALP(I))                  ET 214
      GO TO 24                                                ET 215
23    PX=WY*QZ-WZ*QY                                          ET 216
      PY=WZ*QX-WX*QZ                                          ET 217
      PZ=WX*QY-WY*QX                                          ET 218
      TT2=DSH*TT1*CMPLX(1./R,TP)/R*STH*SALP(II)               ET 219
      CX=TT2*PX                                               ET 220
      CY=TT2*PY                                               ET 221
      CZ=TT2*PZ                                               ET 222
      E(I2)=CX*T1X(II)+CY*T1Y(II)+CZ*T1Z(II)                  ET 223
      E(I1)=CX*T2X(II)+CY*T2Y(II)+CZ*T2Z(II)                  ET 224
24    CONTINUE                                                ET 225
      RETURN                                                  ET 226
      END                                                     ET 227-
```

FACGF

PURPOSE

To perform the steps in the NGF solution that do not depend on the excitation vector.

METHOD

The NGF solution procedure is discussed in Section VI. The steps performed in FACGF are to evaluate $A^{-1}B$ and $D - CA^{-1}B$. The matrix $D - CA^{-1}B$ is then factored into triangular matrices L and U. The procedure is complicated by the possible need to use file storage for the matrices. The comments in the code and the tables for ICASX = 2, 3 and 4 in Section VII offer a fairly complete description of the procedure.

SYMBOL DICTIONARY

A	= array for matrix A (L U factors) or block of A if file storage is used
B	= array for B or block of B
BX	= array for B when $A^{-1}B$ is being computed with ICASX = 2. The array B starts at the beginning of CM in this case. BX leaves room for A_F at the beginning of CM
C	= array for C or block of C (matrix transposed)
D	= array for D or block of D (matrix transposed)
IBFL	= file on which B is stored
ICASS	= saved value of ICASE
IP	= pivot index array
IX	= data on row interchanges in LFACTR
M1	= number of patches in the NGF
MP	= number of patches in a symmetric section in the NGF
N1	= number of segments in the NGF
N1C	= number of columns in C (same as order of A)
N1CP	= N1C + 1
N2C	= order of matrix D
NBLSYS	= saved value of NBLSYM
NIC	= index increment
NLSYS	= saved value of NLSYM

= number of segments in a symmetric section in the NGF

SYS = saved value of NPSYM

JM = summation variable for matrix products

```
      SUBROUTINE FACGF (A,B,C,D,BX,IP,IX,NP,N1,MP,M1,N1C,N2C)           FG   1
C     FACGF COMPUTES AND FACTORS D-C(INV(A)B).                          FG   2
      COMPLEX A,B,C,D,BX,SUM                                            FG   3
      COMMON /MATPAR/ ICASE,NBLOKS,NPBLK,NLAST,NBLSYM,NPSYM,NLSYM,IMAT,I FG   4
     1CASX,NBBX,NPBX,NLBX,NBBL,NPBL,NLBL                                FG   5
      DIMENSION A(1), B(N1C,1), C(N1C,1), D(N2C,1), BX(N1C,1), IP(1), IX FG   6
     1(1)                                                               FG   7
      IF (N2C.EQ.0) RETURN                                              FG   8
      IBFL=14                                                           FG   9
      IF (ICASX.LT.3) GO TO 1                                           FG  10
C     CONVERT B FROM BLOCKS OF ROWS ON T14 TO BLOCKS OF COL. ON T16     FG  11
      CALL REBLK (B,C,N1C,NPBX,N2C)                                     FG  12
      IBFL=16                                                           FG  13
1     NPB=NPBL                                                          FG  14
      IF (ICASX.EQ.2) REWIND 14                                         FG  15
C     COMPUTE INV(A)B AND WRITE ON TAPE14                               FG  16
      DO 2 IB=1,NBBL                                                    FG  17
      IF (IB.EQ.NBBL) NPB=NLBL                                          FG  18
      IF (ICASX.GT.1) READ (IBFL) ((BX(I,J),I=1,N1C),J=1,NPB)           FG  19
      CALL SOLVES (A,IP,BX,N1C,NPB,NP,N1,MP,M1,13,13)                   FG  20
      IF (ICASX.EQ.2) REWIND 14                                         FG  21
      IF (ICASX.GT.1) WRITE (14) ((BX(I,J),I=1,N1C),J=1,NPB)            FG  22
2     CONTINUE                                                          FG  23
      IF (ICASX.EQ.1) GO TO 3                                           FG  24
      REWIND 11                                                         FG  25
      REWIND 12                                                         FG  26
      REWIND 15                                                         FG  27
      REWIND IBFL                                                       FG  28
3     NPC=NPBL                                                          FG  29
C     COMPUTE D-C(INV(A)B) AND WRITE ON TAPE11                          FG  30
      DO 8 IC=1,NBBL                                                    FG  31
      IF (IC.EQ.NBBL) NPC=NLBL                                          FG  32
      IF (ICASX.EQ.1) GO TO 4                                           FG  33
      READ (15) ((C(I,J),I=1,N1C),J=1,NPC)                              FG  34
      READ (12) ((D(I,J),I=1,N2C),J=1,NPC)                              FG  35
      REWIND 14                                                         FG  36
4     NPB=NPBL                                                          FG  37
      NIC=0                                                             FG  38
      DO 7 IB=1,NBBL                                                    FG  39
      IF (IB.EQ.NBBL) NPB=NLBL                                          FG  40
      IF (ICASX.GT.1) READ (14) ((B(I,J),I=1,N1C),J=1,NPB)              FG  41
      DO 6 I=1,NPB                                                      FG  42
      II=I+NIC                                                          FG  43
      DO 6 J=1,NPC                                                      FG  44
      SUM=(0.,0.)                                                       FG  45
      DO 5 K=1,N1C                                                      FG  46
5     SUM=SUM+B(K,I)*C(K,J)                                             FG  47
6     D(II,J)=D(II,J)-SUM                                               FG  48
7     NIC=NIC+NPBL                                                      FG  49
      IF (ICASX.GT.1) WRITE (11) ((D(I,J),I=1,N2C),J=1,NPBL)            FG  50
8     CONTINUE                                                          FG  51
      IF (ICASX.EQ.1) GO TO 9                                           FG  52
      REWIND 11                                                         FG  53
      REWIND 12                                                         FG  54
      REWIND 14                                                         FG  55
      REWIND 15                                                         FG  56
9     N1CP=N1C+1                                                        FG  57
C     FACTOR D-C(INV(A)B)                                               FG  58
      IF (ICASX.GT.1) GO TO 10                                          FG  59
      CALL FACTR (N2C,D,IP(N1CP),N2C)                                   FG  60
      GO TO 13                                                          FG  61
10    IF (ICASX.EQ.4) GO TO 12                                          FG  62
      NPB=NPBL                                                          FG  63
      IC=0                                                              FG  64
```

```
      DO 11 IB=1,NBBL
      IF (IB.EQ.NBBL) NPB=NLBL
      II=IC+1
      IC=IC+N2C*NPB
   11 READ (11) (B(I,1),I=II,IC)
      REWIND 11
      CALL FACTR (N2C,B,IP(N1CP),N2C)
      NIC=N2C*N2C
      WRITE (11) (B(I,1),I=1,NIC)
      REWIND 11
      GO TO 13
   12 NBLSYS=NBLSYM
      NPSYS=NPSYM
      NLSYS=NLSYM
      ICASS=ICASE
      NBLSYM=NBBL
      NPSYM=NPBL
      NLSYM=NLBL
      ICASE=3
      CALL FACIO (B,N2C,1,IX(N1CP),11,12,16,11)
      CALL LUNSCR (B,N2C,1,IP(N1CP),IX(N1CP),12,11,16)
      NBLSYM=NBLSYS
      NPSYM=NPSYS
      NLSYM=NLSYS
      ICASE=ICASS
   13 RETURN
      END
```

FACIO

PURPOSE

To read and write matrix blocks needed for the LU decomposition.

METHOD

Sequential access is used on all files. The matrix is initially stored on file IU1 in blocks of columns of the transposed matrix. The block size is such that two blocks will fit into the array A for the Gauss elimination process. If the matrix were divided into four blocks, the order for reading the blocks into core would be

Blocks	
1, 2	1 and 2 will be completely factored
1, 3	3 and 4 partially factored
1, 4	
2, 3	factorization of 3 completed
2, 4	4 partially factored
3, 4	factorization complete

IU1 is the initial input file. Partially factored blocks are read from file IFILE3 and written to IFILE4 where IFILE3 = IU3 and IFILE4 = IU4 when IXBLK1 is odd, and IFILE3 = IU4 and IFILE4 = IU3 when IXBLK1 is even. Completed blocks are written to file IU2. Although the last block may be shorter than other blocks the same number of words is read or written. The excess words are ignored in subroutine LFACTR.

Subroutine LFACTR is called to perform the Gauss elimination. For a symmetric structure the loop from F018 to F043 factors each submatrix.

SYMBOL DICTIONARY

A	= array for matrix storage
I1	= location in A of beginning of block 1
I2	= location in A of end of block 1
I3	= location in A of beginning of block 2
I4	= location in A of end of block 2
IFILE3	= input file
IFILE4	= output file
IP	= array for pivot element indices

IT	= number of words in a matrix block
IU1, IU2, IU3, IU4	= file numbers
IXBLK1	= number of first block stored in A
IXBLK2	= number of second block stored in A
KA	= first location in IP for submatrix KK
NBM	= number of blocks minus one
NOP	= number of submatrices for symmetry
NROW	= number of rows in a block
T1, T2, TIME	= variables to sum total time spent in LFACTR

```
      SUBROUTINE FACIO (A,NROW,NOP,IP,IU1,IU2,IU3,IU4)
C
C     FACIO CONTROLS I/O FOR OUT-OF-CORE FACTORIZATION
C
      COMPLEX A
      COMMON /MATPAR/ ICASE,NBLOKS,NPBLK,NLAST,NBLSYM,NPSYM,NLSYM,IMAT,I
     1CASX,NBBX,NPBX,NLBX,NBBL,NPBL,NLBL
      DIMENSION A(NROW,1), IP(NROW)
      IT=2*NPSYM*NROW
      NBM=NBLSYM-1
      I1=1
      I2=IT
      I3=I2+1
      I4=2*IT
      TIME=0.
      REWIND IU1
      REWIND IU2
      DO 3 KK=1,NOP
      KA=(KK-1)*NROW+1
      IFILE3=IU1
      IFILE4=IU3
      DO 2 IXBLK1=1,NBM
      REWIND IU3
      REWIND IU4
      CALL BLCKIN (A,IFILE3,I1,I2,1,17)
      IXBP=IXBLK1+1
      DO 1 IXBLK2=IXBP,NBLSYM
      CALL BLCKIN (A,IFILE3,I3,I4,1,18)
      CALL SECOND (T1)
      CALL LFACTR (A,NROW,IXBLK1,IXBLK2,IP(KA))
      CALL SECOND (T2)
      TIME=TIME+T2-T1
      IF (IXBLK2.EQ.IXBP) CALL BLCKOT (A,IU2,I1,I2,1,19)
      IF (IXBLK1.EQ.NBM.AND.IXBLK2.EQ.NBLSYM) IFILE4=IU2
      CALL BLCKOT (A,IFILE4,I3,I4,1,20)
    1 CONTINUE
      IFILE3=IU3
      IFILE4=IU4
      IF ((IXBLK1/2)*2.NE.IXBLK1) GO TO 2
      IFILE3=IU4
      IFILE4=IU3
    2 CONTINUE
    3 CONTINUE
      REWIND IU1
      REWIND IU2
      REWIND IU3
      REWIND IU4
      PRINT 4, TIME
      RETURN
C
    4 FORMAT (35H CP TIME TAKEN FOR FACTORIZATION = ,E12.5)
      END
```

FACTR

PURPOSE

To factor a complex matrix into a lower triangular and an upper triangular matrix using the Gauss-Doolittle technique. The matrix in this case is a transposed matrix. The factored matrix is used by subroutine SOLVE to determine the solution of the matrix equation Ax = B.

METHOD

The algorithm used in this routine is presented by A. Ralston (ref. 1). The decomposition of the matrix A is such that A = LU, where L is a lower triangular matrix with 1's down the diagonal, and U is an upper triangular matrix. The L and U matrices overwrite the matrix A. The computations to obtain L and U are done using one complex scratch vector (D) and one integer vector (IP) that keep track of row interchanges when elements are positioned for size. If positioning for size is not taken into account, the general procedure is

$$a_{11} = u_{11}$$

$$a_{i1} = \ell_{i1}u_{11} \qquad i = 2, \ldots, n$$

which gives the first column of the L and U matrices. Then

$$a_{12} = u_{12}$$

$$a_{22} = \ell_{21}u_{12} + u_{22}$$

$$a_{i2} = \ell_{i1}u_{12} + \ell_{i2}u_{22} \qquad i = 3, \ldots, n$$

gives the second column. The computations for the successive columns continue in this way. The general equations for the r^{th} column are

$$a_{1r} = u_{1r}$$

$$a_{2r} = \ell_{21}u_{1r} + u_{2r}$$

$$\vdots$$

$$a_{rr} = \ell_{r1}u_{1r} + \ell_{r2}u_{2r} + \ldots + \ell_{r,r-1}u_{r-1,r} + u_{rr}$$

$$a_{ir} = \ell_{i1}u_{1r} + \ldots + \ell_{ir}u_{rr}, \qquad i = r + 1, \ldots, n$$

There are only two differences in the coding used in FACTR and the coding suggested by Ralston. The first is that double precision variables are not used for the accumulation of sums, since for the size and conditioning of the matrices anticipated in core, the computer word length is sufficient to insure accuracy. The second difference is that the row and column indices of the A matrix in the routine have been interchanged to handle the transposed matrix.

CODING

The coding is divided into five steps which correspond to the steps given by Ralston.

FA14	Loop over columns (rows with the interchanged indices used in the routine).
FA18 - FA20	Fill D vector with column (row) of A.
FA24 - FA35	Solution for u_{ir} $(i = 1,\ldots,r)$ in the above equations taking into account positioning.
FA40 - FA54	Selecting largest value for positioning.
FA58 - FA62	Solution for ℓ_{ir} $(i = r + 1,\ldots,n)$ in the above equations.
FA64 - FA66	Printing of small pivot elements.

SYMBOL DICTIONARY

A	= input transposed matrix overwritten with calculated L^T and U^T matrices
CONJG	= external routine (conjugate of a complex number)
D	= scratch vector
DMAX	= maximum value in D
ELMAG	= intermediate variable
I	= DO loop index
IFLG	= small pivot flag
IP	= integer vector storing positioning information
J	= DO loop index
JP1	= J + 1
K	= DO loop index
N	= order of matrix being factored
NDIM	= dimensions of the array where the matrix is stored. NDIM $\geq$ N
PJ	= intermediate variable
PR	= intermediate variable

R	=	DO loop index
REAL	=	external routine (real part of complex number)
RM1	=	R - 1
RP1	=	R + 1

```
      SUBROUTINE FACTR (N,A,IP,NDIM)
C
C     SUBROUTINE TO FACTOR A MATRIX INTO A UNIT LOWER TRIANGULAR MATRIX
C     AND AN UPPER TRIANGULAR MATRIX USING THE GAUSS-DOOLITTLE ALGORITHM
C     PRESENTED ON PAGES 411-416 OF A. RALSTON--A FIRST COURSE IN
C     NUMERICAL ANALYSIS.  COMMENTS BELOW REFER TO COMMENTS IN RALSTONS
C     TEXT.    (MATRIX TRANSPOSED.
C
      COMPLEX A,D,ARJ
      DIMENSION A(NDIM,NDIM), IP(NDIM)
      COMMON /SCRATM/ D(600)
      INTEGER R,RM1,RP1,PJ,PR
      IFLG=0
      DO 9 R=1,N
C
C     STEP 1
C
      DO 1 K=1,N
      D(K)=A(R,K)
1     CONTINUE
C
C     STEPS 2 AND 3
C
      RM1=R-1
      IF (RM1.LT.1) GO TO 4
      DO 3 J=1,RM1
      PJ=IP(J)
      ARJ=D(PJ)
      A(R,J)=ARJ
      D(PJ)=D(J)
      JP1=J+1
      DO 2 I=JP1,N
      D(I)=D(I)-A(J,I)*ARJ
2     CONTINUE
3     CONTINUE
4     CONTINUE
C
C     STEP 4
C
      DMAX=REAL(D(R)*CONJG(D(R)))
      IP(R)=R
      RP1=R+1
      IF (RP1.GT.N) GO TO 6
      DO 5 I=RP1,N
      ELMAG=REAL(D(I)*CONJG(D(I)))
      IF (ELMAG.LT.DMAX) GO TO 5
      DMAX=ELMAG
      IP(R)=I
5     CONTINUE
6     CONTINUE
      IF (DMAX.LT.1.E-10) IFLG=1
      PR=IP(R)
      A(R,R)=D(PR)
      D(PR)=D(R)
C
C     STEP 5
C
      IF (RP1.GT.N) GO TO 8
      ARJ=1./A(R,R)
      DO 7 I=RP1,N
      A(R,I)=D(I)*ARJ
7     CONTINUE
8     CONTINUE
      IF (IFLG.EQ.0) GO TO 9
```

```
      PRINT 10, R,DMAX
      IFLG=0
    9 CONTINUE
      RETURN
C
   10 FORMAT (1H ,6HPIVOT(,I3,2H)=,E16.8)
      END
```

FACTRS

PURPOSE

To call the appropriate subroutines for the LU decomposition of a matrix.

METHOD

The operation of FACTRS depends on the mode of storage of the matrix as determined by the value of ICASE (see COMMON/MATPAR/ in Section III). For ICASE = 1 subroutine FACTR is called at FS16 to factor the matrix. For ICASE = 2 FACTR is called for each of the NOP submatrices. If ICASE = 3 FACIO and LUNSCR are called at FS23 and FS24. FACIO reads the matrix from file IU1 and writes the result on file IU2. LUNSCR leaves the final result on file IU3.

For ICASE = 4 (symmetry, submatrices fit in core) or ICASE = 5 (symmetry, submatrices do not fit in core) the matrix elements on file IU1 are written in a new order on file IU2 from FS29 to FS46. The sequence of data on file IU1 is

column 1 of submatrix 1
column 1 of submatrix 2
⋮
column 1 of submatrix NOP
column 2 of submatrix 1
⋮
column 2 of submatrix NOP
column 3 of submatrix 1
⋮
column NPBLK of submatrix NOP

The matrices are written onto file IU2 in the sequence

column 1 of submatrix 1
column 2 of submatrix 1
⋮

column NPBLK of submatrix 1

column 1 of submatrix 2

⋮

column NPBLK of submatrix NOP

For ICASE = 4 each submatrix is then read into memory at FS58 and decomposed into LU factors by calling FACTR at FS60. The factored matrices are written to file IU3 at FS61.

For ICASE = 5 the matrices are transferred from file IU2 to IU1 at FS76 to FS77. Subroutine FACIO is then called to factor all of the NOP submatrices. The result is left on file IU2. LUNSCR reorders the rows of each matrix and leaves the result on IU3.

SYMBOL DICTIONARY

A	= array for matrix storage
I2	= number of words in a block
ICOLS	= number of columns in a block
IP	= array for pivot element indices
IR1, IR2, IRR1, IRR2	= row indices for reordering columns
IU1, IU2, IU3, IU4	= file numbers
IX	= array of pivot element data
KA	= starting location of a submatrix in the array
NOP	= number of symmetric sections
NP	= number of equations for each symmetric section (order of submatrix)
NROW	= total number of equations (NP x NOP)

```
      SUBROUTINE FACTRS (NP,NROW,A,IP,IX,IU1,IU2,IU3,IU4)
C
C     FACTRS, FOR SYMMETRIC STRUCTURE, TRANSFORMS SUBMATRICIES TO FORM
C     MATRICIES OF THE SYMMETRIC MODES AND CALLS ROUTINE TO FACTOR
C     MATRICIES.  IF NO SYMMETRY, THE ROUTINE IS CALLED TO FACTOR THE
C     COMPLETE MATRIX.
C
      COMPLEX A
      COMMON /MATPAR/ ICASE,NBLOKS,NPBLK,NLAST,NBLSYM,NPSYM,NLSYM,IMAT,I
     1CASX,NBBX,NPBX,NLBX,NBBL,NPBL,NLBL
      DIMENSION A(1), IP(NROW), IX(NROW)
      NOP=NROW/NP
      IF (ICASE.GT.2) GO TO 2
      DO 1 KK=1,NOP
      KA=(KK-1)*NP+1
    1 CALL FACTR (NP,A(KA),IP(KA),NROW)
      RETURN
    2 IF (ICASE.GT.3) GO TO 3
C
C     FACTOR SUBMATRICIES, OR FACTOR COMPLETE MATRIX IF NO SYMMETRY
C     EXISTS.
C
      CALL FACIO (A,NROW,NOP,IX,IU1,IU2,IU3,IU4)
      CALL LUNSCR (A,NROW,NOP,IP,IX,IU2,IU3,IU4)
      RETURN
C
C     REWRITE THE MATRICES BY COLUMNS ON TAPE 13
C
    3 I2=2*NPBLK*NROW
      REWIND IU2
      DO 5 K=1,NOP
      REWIND IU1
      ICOLS=NPBLK
      IR2=K*NP
      IR1=IR2-NP+1
      DO 5 L=1,NBLOKS
      IF (NBLOKS.EQ.1.AND.K.GT.1) GO TO 4
      CALL BLCKIN (A,IU1,1,I2,1,602)
      IF (L.EQ.NBLOKS) ICOLS=NLAST
    4 IRR1=IR1
      IRR2=IR2
      DO 5 ICOLDX=1,ICOLS
      WRITE (IU2) (A(I),I=IRR1,IRR2)
      IRR1=IRR1+NROW
      IRR2=IRR2+NROW
    5 CONTINUE
      REWIND IU1
      REWIND IU2
      IF (ICASE.EQ.5) GO TO 8
      REWIND IU3
      IRR1=NP*NP
      DO 7 KK=1,NOP
      IR1=1-NP
      IR2=0
      DO 6 I=1,NP
      IR1=IR1+NP
      IR2=IR2+NP
    6 READ (IU2) (A(J),J=IR1,IR2)
      KA=(KK-1)*NP+1
      CALL FACTR (NP,A,IP(KA),NP)
      WRITE (IU3) (A(I),I=1,IRR1)
    7 CONTINUE
      REWIND IU2
      REWIND IU3
```

```
      RETURN
    8 I2=2*NPSYM*NP
      DO 10 KK=1,NOP
      J2=NPSYM
      DO 10 L=1,NBLSYM
      IF (L.EQ.NBLSYM) J2=NLSYM
      IR1=1-NP
      IR2=0
      DO 9 J=1,J2
      IR1=IR1+NP
      IR2=IR2+NP
    9 READ (IU2) (A(I),I=IR1,IR2)
   10 CALL BLCKOT (A,IU1,1,I2,1,193)
      REWIND IU1
      CALL FACIO (A,NP,NOP,IX,IU1,IU2,IU3,IU4)
      CALL LUNSCR (A,NP,NOP,IP,IX,IU2,IU3,IU4)
      RETURN
      END
```

FBAR

PURPOSE

To compute the Sommerfeld attenuation function for Norton's asymptotic field approximations.

METHOD

The value returned for FBAR is

$$F(P) = 1 - j\,\sqrt{\pi P}\,\exp(-P)\,[1 - \mathrm{erf}(j\sqrt{P})]$$

where erf(z) is the error function. If $|j\sqrt{P}| \leq 3$ the value of $\mathrm{erf}(j\sqrt{P})$ is computed from the series

$$\mathrm{erf}(z) = \frac{2}{\sqrt{\pi}} \sum_{n=0}^{\infty} \frac{(-1)^n z^{2n+1}}{n!\,(2n+1)}$$

For $|j\sqrt{P}| > 3$, F(P) is evaluated from the first six terms of the asymptotic expansion

$$\sqrt{\pi}\, z \exp(z^2)\,(1 - \mathrm{erf}(z)) \approx 1 + \sum_{M=1}^{\infty} (-1)^M \frac{1 \cdot 3 \ldots (2M-1)}{(2z^2)^M}$$

for $z \to \infty$, $|\arg(z)| < \frac{3\pi}{4}$

SYMBOL DICTIONARY

ACCS	=	relative convergence test value
FJ	=	$j = \sqrt{-1}$
MINUS	=	1 if Re(z) < 0
P	=	P
POW	=	$(-1)^n\, z^{2n+1}/n!$
SMS	=	magnitude squared of series
SP	=	$\sqrt{\pi}$
SUM	=	series value

TERM = term in the series
TMS = $|TERM|^2$
TOSP = $2/\sqrt{\pi}$
Z = $j\sqrt{P}$
ZS = z^2

```
      COMPLEX FUNCTION FBAR(P)
C
C     FBAR IS SOMMERFELD ATTENUATION FUNCTION FOR NUMERICAL DISTANCE P
C
      COMPLEX Z,ZS,SUM,POW,TERM,P,FJ
      DIMENSION FJX(2)
      EQUIVALENCE (FJ,FJX)
      DATA TOSP/1.128379167/,ACCS/1.E-12/,SP/1.772453851/,FJX/0.,1./
      Z=FJ*CSQRT(P)
      IF (CABS(Z).GT.3.) GO TO 3
C
C     SERIES EXPANSION
C
      ZS=Z*Z
      SUM=Z
      POW=Z
      DO 1 I=1,100
      POW=-POW*ZS/FLOAT(I)
      TERM=POW/(2.*I+1.)
      SUM=SUM+TERM
      TMS=REAL(TERM*CONJG(TERM))
      SMS=REAL(SUM*CONJG(SUM))
      IF (TMS/SMS.LT.ACCS) GO TO 2
    1 CONTINUE
    2 FBAR=1.-(1.-SUM*TOSP)*Z*CEXP(ZS)*SP
      RETURN
C
C     ASYMPTOTIC EXPANSION
C
    3 IF (REAL(Z).GE.0.) GO TO 4
      MINUS=1
      Z=-Z
      GO TO 5
    4 MINUS=0
    5 ZS=.5/(Z*Z)
      SUM=(0.,0.)
      TERM=(1.,0.)
      DO 6 I=1,6
      TERM=-TERM*(2.*I-1.)*ZS
    6 SUM=SUM+TERM
      IF (MINUS.EQ.1) SUM=SUM-2.*SP*Z*CEXP(Z*Z)
      FBAR=-SUM
      RETURN
      END
```

FBLOCK

PURPOSE

To set parameters for storage of the interaction matrix.

METHOD

FBLOCK sets values of the parameters ICASE through NLSYM in COMMON/MATPAR/. The input parameters NROW and NCOL are the number of rows and columns in the non-transposed matrix. IMAX is the number of matrix elements that can be stored in the array in COMMON/CMB/. If a NGF file will be written (WG card) then IRNGF complex locations are reserved for future use. If a NGF file has not been requested then IRNGF is zero.

If (NROW)(NCOL) $\leq$ IMAX - IRNGF the complete matrix can be stored in COMMON/CMB/. ICASE is then 1 for no symmetry or 2 for symmetry. If the structure has symmetry and one submatrix fits in core but not the complete matrix,

$$(NROW)(NCOL) > IMAX - IRNGF$$
$$NROW^2 \leq IMAX - IRNGF,$$

then ICASE is 4.

If the matrix cannot fit in core for the LU decomposition then it is divided into blocks of rows (columns of the transposed matrix) for transfer between core and file storage. The blocks are made as large as possible so that one block fits into IMAX - IRNGF locations and two blocks fit into IMAX locations. Since two blocks are needed in core only during the Gauss elimination process this makes at least IRNGF locations available during the NGF solution.

CODING

FB10 - FB17	ICASE = 1 or 2
FB20 - FB32	ICASE = 3
FB34 - FB40	ICASE = 4 or 5, block parameters for whole matrix
FB42 - FB48	ICASE = 4, block parameters for submatrices
FB49 - FB58	ICASE = 5, block parameters for submatrices

FBLOCK

FB65 - FB71   S matrix for rotational symmetry (Equation III of Part I)

FB75 - FB88   S matrix for plane symmetry

SYMBOL DICTIONARY

ARG = $2\pi(I - 1)(J - 1)/NOP$

IMAX = number of complex numbers that can be stored in COMMON/CMB/

IMX1 = IMAX - IRNGF

IPSYM = parameter from COMMON/DATA/

IRNGF = array storage reserved for NGF

KA = number of planes of symmetry

NCOL = number of columns in matrix

NOP = number of symmetric sections

NROW = number of rows in matrix

PHAZ = $2\pi/NOP$

```
      SUBROUTINE FBLOCK (NROW,NCOL,IMAX,IRNGF,IPSYM)
C     FBLOCK SETS PARAMETERS FOR OUT-OF-CORE SOLUTION FOR THE PRIMARY
C     MATRIX (A)
      COMPLEX SSX,DETER
      COMMON /MATPAR/ ICASE,NBLOKS,NPBLK,NLAST,NBLSYM,NPSYM,NLSYM,IMAT,I
     1CASX,NBBX,NPBX,NLBX,NBBL,NPBL,NLBL
      COMMON /SMAT/ SSX(16,16)
      IMX1=IMAX-IRNGF
      IF (NROW*NCOL.GT.IMX1) GO TO 2
      NBLOKS=1
      NPBLK=NROW
      NLAST=NROW
      IMAT=NROW*NCOL
      IF (NROW.NE.NCOL) GO TO 1
      ICASE=1
      RETURN
1     ICASE=2
      GO TO 5
2     IF (NROW.NE.NCOL) GO TO 3
      ICASE=3
      NPBLK=IMAX/(2*NCOL)
      NPSYM=IMX1/NCOL
      IF (NPSYM.LT.NPBLK) NPBLK=NPSYM
      IF (NPBLK.LT.1) GO TO 12
      NBLOKS=(NROW-1)/NPBLK
      NLAST=NROW-NBLOKS*NPBLK
      NBLOKS=NBLOKS+1
      NBLSYM=NBLOKS
      NPSYM=NPBLK
      NLSYM=NLAST
      IMAT=NPBLK*NCOL
      PRINT 14, NBLOKS,NPBLK,NLAST
      GO TO 11
3     NPBLK=IMAX/NCOL
      IF (NPBLK.LT.1) GO TO 12
      IF (NPBLK.GT.NROW) NPBLK=NROW
      NBLOKS=(NROW-1)/NPBLK
      NLAST=NROW-NBLOKS*NPBLK
      NBLOKS=NBLOKS+1
      PRINT 14, NBLOKS,NPBLK,NLAST
      IF (NROW*NROW.GT.IMX1) GO TO 4
      ICASE=4
      NBLSYM=1
      NPSYM=NROW
      NLSYM=NROW
      IMAT=NROW*NROW
      PRINT 15
      GO TO 5
4     ICASE=5
      NPSYM=IMAX/(2*NROW)
      NBLSYM=IMX1/NROW
      IF (NBLSYM.LT.NPSYM) NPSYM=NBLSYM
      IF (NPSYM.LT.1) GO TO 12
      NBLSYM=(NROW-1)/NPSYM
      NLSYM=NROW-NBLSYM*NPSYM
      NBLSYM=NBLSYM+1
      PRINT 16, NBLSYM,NPSYM,NLSYM
      IMAT=NPSYM*NROW
5     NOP=NCOL/NROW
      IF (NOP*NROW.NE.NCOL) GO TO 13
      IF (IPSYM.GT.0) GO TO 7
C
C     SET UP SSX MATRIX FOR ROTATIONAL SYMMETRY.
C
```

```
      PHAZ=6.2831853072/NOP
      DO 6 I=2,NOP
      DO 6 J=I,NOP
      ARG=PHAZ*FLOAT(I-1)*FLOAT(J-1)
      SSX(I,J)=CMPLX(COS(ARG),SIN(ARG))
    6 SSX(J,I)=SSX(I,J)
      GO TO 11
C
C     SET UP SSX MATRIX FOR PLANE SYMMETRY
C
    7 KK=1
      SSX(1,1)=(1.,0.)
      IF ((NOP.EQ.2).OR.(NOP.EQ.4).OR.(NOP.EQ.8)) GO TO 8
      STOP
    8 KA=NOP/2
      IF (NOP.EQ.8) KA=3
      DO 10 K=1,KA
      DO 9 I=1,KK
      DO 9 J=1,KK
      DETER=SSX(I,J)
      SSX(I,J+KK)=DETER
      SSX(I+KK,J+KK)=-DETER
    9 SSX(I+KK,J)=DETER
   10 KK=KK*2
   11 RETURN
   12 PRINT 17, NROW,NCOL
      STOP
   13 PRINT 18, NROW,NCOL
      STOP
C
   14 FORMAT (//35H MATRIX FILE STORAGE -  NO. BLOCKS=,I5,19H COLUMNS PE
     1R BLOCK=,I5,23H COLUMNS IN LAST BLOCK=,I5)
   15 FORMAT (25H SUBMATRICIES FIT IN CORE)
   16 FORMAT (38H SUBMATRIX PARTITIONING -  NO. BLOCKS=,I5,19H COLUMNS P
     1ER BLOCK=,I5,23H COLUMNS IN LAST BLOCK=,I5)
   17 FORMAT (40H ERROR - INSUFFICIENT STORAGE FOR MATRIX,2I5)
   18 FORMAT (28H SYMMETRY ERROR - NROW,NCOL=,2I5)
      END
```

FBNGF

PURPOSE

To set parameters for storage of the matrices B, C and D for the NGF solution.

METHOD

The modes of matrix storage for the NGF solution are described in Section VIII. FBNGF choses the smallest ICASX (1 through 4) possible given the size of the matrices A, B, C and D and the space available in the array CM in COMMON/CMB/. If B, C and D must be divided into blocks (ICASX = 3 or 4) the blocks are chosen are large as possible to minimize the number of input and output requests. Parameters specifying the number and size of blocks are stored in COMMON/MATPAR/ (see Section III).

FBNGF also sets the locations in CM at which storage of B, C and D start. For example, CM(IC11) is passed from the main program to subroutines CMNGF and FACGF as the starting location of array C.

SYMBOL DICTIONARY

IB11 = location in CM at which storage of B starts

IC11 = location in CM at which storage of C starts

ID11 = location in CM at which storage of D starts

IMAT = number of complex numbers in A_F

IR = space available (complex numbers) in CM when A_F is not being used.

IRESRV = total length of CM

IRESX = space availabe in CM when A_F is being used

IX11 = location in CM at which storage of B starts when $A^{-1}B$ is computed (A_F occupies space in CM)

NBCD = number of complex numbers in B, C and D combined

NBLN = number of complex numbers in B or C

NDLN = length of D

NEQ = number of rows in B, columns in C

NEQ2 = number of columns in B or D, rows in C or D

```
      SUBROUTINE FBNGF (NEQ,NEQ2,IRESRV,IB11,IC11,ID11,IX11)
C     FBNGF SETS THE BLOCKING PARAMETERS FOR THE B, C, AND D ARRAYS FOR
C     OUT-OF-CORE STORAGE.
      COMMON /MATPAR/ ICASE,NBLOKS,NPBLK,NLAST,NBLSYM,NPSYM,NLSYM,IMAT,I
     1CASX,NBBX,NPBX,NLBX,NBBL,NPBL,NLBL
      IRESX=IRESRV-IMAT
      NBLN=NEQ*NEQ2
      NDLN=NEQ2*NEQ2
      NBCD=2*NBLN+NDLN
      IF (NBCD.GT.IRESX) GO TO 1
      ICASX=1
      IB11=IMAT+1
      GO TO 2
1     IF (ICASE.LT.3) GO TO 3
      IF (NBCD.GT.IRESRV.OR.NBLN.GT.IRESX) GO TO 3
      ICASX=2
      IB11=1
2     NBBX=1
      NPBX=NEQ
      NLBX=NEQ
      NBBL=1
      NPBL=NEQ2
      NLBL=NEQ2
      GO TO 5
3     IR=IRESRV
      IF (ICASE.LT.3) IR=IRESX
      ICASX=3
      IF (NDLN.GT.IR) ICASX=4
      NBCD=2*NEQ+NEQ2
      NPBL=IR/NBCD
      NLBL=IR/(2*NEQ2)
      IF (NLBL.LT.NPBL) NPBL=NLBL
      IF (ICASE.LT.3) GO TO 4
      NLBL=IRESX/NEQ
      IF (NLBL.LT.NPBL) NPBL=NLBL
4     IF (NPBL.LT.1) GO TO 6
      NBBL=(NEQ2-1)/NPBL
      NLBL=NEQ2-NBBL*NPBL
      NBBL=NBBL+1
      NBLN=NEQ*NPBL
      IR=IR-NBLN
      NPBX=IR/NEQ2
      IF (NPBX.GT.NEQ) NPBX=NEQ
      NBBX=(NEQ-1)/NPBX
      NLBX=NEQ-NBBX*NPBX
      NBBX=NBBX+1
      IB11=1
      IF (ICASE.LT.3) IB11=IMAT+1
5     IC11=IB11+NBLN
      ID11=IC11+NBLN
      IX11=IMAT+1
      PRINT 11, NEQ2
      IF (ICASX.EQ.1) RETURN
      PRINT 8, ICASX
      PRINT 9, NBBX,NPBX,NLBX
      PRINT 10, NBBL,NPBL,NLBL
      RETURN
6     PRINT 7, IRESRV,IMAT,NEQ,NEQ2
      STOP
C
7     FORMAT (55H ERROR - INSUFFICIENT STORAGE FOR INTERACTION MATRICIES
     1,24H  IRESRV,IMAT,NEQ,NEQ2 =,4I5)
8     FORMAT (48H FILE STORAGE FOR NEW MATRIX SECTIONS -  ICASX =,I2)
9     FORMAT (19H B FILLED BY ROWS -,15X,12HNO. BLOCKS =,I3,3X,16HROWS P
```

```
      1ER BLOCK =,I3,3X,20HROWS IN LAST BLOCK =,I3)                         FN  65
   10 FORMAT (32H B BY COLUMNS, C AND D BY ROWS -,2X,12HNO. BLOCKS =,I3,    FN  66
     14X,15HR/C PER BLOCK =,I3,4X,19HR/C IN LAST BLOCK =,I3)                FN  67
   11 FORMAT (//,35H N.G.F. - NUMBER OF NEW UNKNOWNS IS,I4)                 FN  68
      END                                                                   FN  69-
```

FFLD

PURPOSE

To calculate the radiated electric field due to the currents on wires and surfaces in free space or over ground. The range factor $\exp(-jkr_0)/(r_0/\lambda)$ is omitted.

METHOD

Equation (126) of Part I is used to evaluate the radiated field of wires and surfaces. The surface part of the equation is evaluated in subroutine FFLDS, however. For wires, the field equation is

$$\bar{E}(\bar{r}_0) = \frac{j\eta \exp(-jkr_0)}{4\pi\ r_0/\lambda} (\hat{k}\hat{k} - \bar{\bar{I}}) \cdot \bar{F}(\bar{r}_0)$$

$$\bar{F}(\bar{r}_0) = 2\pi \int_L \exp(j\bar{k}\cdot\bar{r})\ [\bar{I}(s)/\lambda]\ ds/\lambda$$

where

$r_0 = |\bar{r}_0|$

$\hat{k} = \bar{r}_0/|\bar{r}_0|$

$k = 2\pi/\lambda$

$\bar{k} = k\hat{k}$

$\bar{I}(s)$ = current on the wire at s

$\bar{\bar{I}}$ = identity dyad

L = contour of the wire

$\bar{r}$ = position of the point at s on the wire

The dot product with the dyad $\hat{k}\hat{k} - \bar{\bar{I}}$ results in the component of $\bar{F}$ transverse to $\hat{k}$. This is accomplished in the code by computing the dot products with the unit vectors $\hat{\theta}$ and $\hat{\phi}$, normal to $\hat{k}$.

For a wire structure consisting of N straight segments, $\bar{r}$ on segment i is replaced by

$$\bar{r} = \bar{r}_i + \lambda t\hat{u}_i ,$$

where

$\bar{r}_i$ = location of the center of segment i

$\bar{u}_i$ = unit vector in the direction of segment i

Then, $\overline{F}$ is evaluated as

$$\overline{F}(\overline{r}_0) = \sum_{i=1}^{N} \exp(j\overline{k} \cdot \overline{r}_i)\ \overline{Q}_i$$

$$Q_i = 2\pi\hat{u}_i \int_{-\Delta_i/2}^{\Delta_i/2} \exp[j2\pi t(\hat{k} \cdot \hat{u}_i)]\ I_i(t)/\lambda\ dt$$

where Δ_i is the length of segment i normalized to λ. With

$$I_i(t)/\lambda = A_i + B_i \sin(2\pi t) + C_i \cos(2\pi t) ,$$

the integral can be evaluated as

$$\overline{Q}_i = \hat{u}_i \left\{ A_i \frac{2 \sin(\pi w_i \Delta_i)}{w_i} - jB_i \left[\frac{\sin[\pi(1 - w_i)\Delta_i]}{(1 - w_i)} - \frac{\sin[\pi(1 + w_i)\Delta_i]}{(1 + w_i)} \right] + C_i \left[\frac{\sin[\pi(1 - w_i)\Delta_i]}{(1 - w_i)} + \frac{\sin[\pi(1 + w_i)\Delta_i]}{(1 + w_i)} \right] \right\} ,$$

where $w_i = -\hat{k} \cdot \hat{u}_i$.

The effect of a ground is included by computing the field of the image of each segment and modifying it by the Fresnel reflection coefficients. The coding here differs from section II-4 of Part I in some respects. Rather than reflecting each segment in the ground plane, the direction of observation, $\hat{k}$, is reflected for the image calculation. Thus, the sign of the z component of $\hat{k}$ is changed at the start of the image calculation. The z component of the image field must also be changed in sign at the end of the calculation. Also, the change in sign of the image field due to the change in sign of charge on the image is combined with the reflection coefficients. Thus, the reflection coefficients are the negative of those in Part I.

The code allows for a change in ground height and electrical parameters at a fixed radial distance from the origin (circular cliff) or at a fixed distance in x (linear cliff). In these cases, the reflection point of the ray from the center of each segment is computed, and the reflection coefficients and phase lag are computed for the appropriate ground. Effects from the region of change, such as diffraction from the edge, are not included, however. A radial wire ground screen may also be included by the reflection coefficient approximation described in section II-4 of Part I.

CODING

FF30 - FF164	Calculation of field due to segments.
FF34 - FF164	Loop over direct and image fields.
FF38 - FF63	Reflection coefficients computed.
FF64	$\hat{k}$ reflected in ground for image.
FF65 - FF70	Direct fields saved, and CIX, CIY, CIZ initialized before image calculation.
FF75 - FF96	Field of segment I computed.
FF102 - FF104	Summation of fields for direct field or uniform ground.
FF110 - FF149	Appropriate reflection coefficient determined and field summed for reflected field from two-medium ground or radial-wire ground screen.
FF156 - FF159	Image field multiplied by reflection coefficients for uniform ground and added to direct field.
FF161 - FF163	Reflected field added to direct field for two-medium ground or radial wire ground.
FF166 - FF167	Dot products of $\overline{F}$ with $\hat{\theta}$ and $\hat{\phi}$ for wires only.
FF169 - FF208	Calculation of field due to surface patches.
FF177 - FF203	Loop over direct and image fields.
FF179	$\hat{k}$ reflected for image.
FF180	FFLDS calculates field.
FF186 - FF202	Field multiplied by reflection coefficients for uniform ground only.

SYMBOL DICTIONARY

A	$= 2 \sin (\pi w_i \Delta_i)/w_i$ (a series is used for small w_i)
ARG	$= \overline{k} \cdot \overline{r}_i$
B	= coefficient of B_i in $\overline{Q}_i$
BOO	$= \sin [\pi(1 - w_i)\Delta_i]/[\pi(1 - w_i)\Delta_i]$
BOT	$= \pi(1 - w_i)\Delta_i$
C	= coefficient of C_i in $\overline{Q}_i$
CAB, SAB, SALP	= x, y, z components of $\hat{u}_i$

CCX, CCY, CCZ	= variables for summation of x, y, and z components of $\overline{F}$
CDP	= $(\overline{F} \cdot \hat{\phi})(R_V - R_H)$
CIX, CIY, CIZ	= variables for summation of x, y, and z components of $\overline{F}$
CONST	= CONSX = $-j\eta/4\pi$
D	= distance of ray reflection point from origin
DARG	= phase increment due to change in ground level
EL	= $\pi\Delta_i$
EPH	= ϕ component of $(r_0/\lambda)\exp(jkr_0)\ \overline{E}(\overline{r}_0)$
ETH	= θ component of $(r_0/\lambda)\exp(jkr_0)\ \overline{E}(\overline{r}_0)$
ETA	= $\eta = \sqrt{\mu/\varepsilon}$
EX, EY, EZ	= $(r_0/\lambda)\exp(jkr_0)\ \overline{E}(\overline{r}_0)$ for patches
EXA	= Q_i
GX, GY, GZ	= $(r_0/\lambda)\exp(jkr_0)\ \overline{E}(\overline{r}_0)$ for direct and reflected fields of patches
I	= segment number
OMEGA	= w_i
PHI	= ϕ
PHX, PHY	= x and y components of $\hat{\phi}$
PI	= π
RFL	= ±1 for direct or image field of patch
RI	= imaginary part of Q_i
ROX, ROY, ROZ	= x, y, and z components of $\hat{k}$
ROZS	= saved value of ROZ
RR	= real part of Q_i
RRH	= $-R_H$
RRH1	= $-R_H$ for first ground medium
RRH2	= $-R_H$ for second ground medium

RRV = $-R_V$

RRV1 = $-R_V$ for first ground medium

RRV2 = $-R_V$ for second ground medium

RRZ = z component of $\hat{k}$

SILL = $\pi w_i \Delta_i$

THET = θ (angle from vertical to $\hat{k}$)

THX, THY, THZ = $\hat{\theta}$

TIX, TIY, TIZ = Q_i for image in ground

TOO = $\sin[\pi(1 + w_i)\Delta_i]/[\pi(1 + w_i)\Delta_i]$

TOP = $\pi(1 + w_i)\Delta_i$

TP = 2π

TTHET = $\tan\theta$

ZRATI = $[\varepsilon_r - j\sigma/(\omega\varepsilon_0)]^{-1/2}$ ε_r, σ = ground parameters

ZRSIN = $[1 - (\mathrm{ZRATI})^2 \sin^2\theta]^{1/2}$

ZSCRN = surface impedance of ground with radial wire ground screen

CONSTANTS

-29.97922085 = $-j\eta/(4\pi)$

3.141592654 = π

376.73 = η

6.283185308 = 2π

```
      SUBROUTINE FFLD (THET,PHI,ETH,EPH)
C                                                                       FF   1
C     FFLD CALCULATES THE FAR ZONE RADIATED ELECTRIC FIELDS,            FF   2
C     THE FACTOR EXP(J*K*R)/(R/LAMDA) NOT INCLUDED                      FF   3
C                                                                       FF   4
      COMPLEX CIX,CIY,CIZ,EXA,ETH,EPH,CONST,CCX,CCY,CCZ,CDP,CUR         FF   5
      COMPLEX ZRATI,ZRSIN,RRV,RRH,RRV1,RRH1,RRV2,RRH2,ZRATI2,TIX,TIY,TIZ FF   6
     1,T1,ZSCRN,EX,EY,EZ,GX,GY,GZ,FRATI                                 FF   7
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300 FF   8
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX( FF   9
     2300),WLAM,IPSYM                                                   FF  10
      COMMON /ANGL/ SALP(300)                                           FF  11
      COMMON /CRNT/ AIR(300),AII(300),BIR(300),BII(300),CIR(300),CII(300 FF  12
     1),CUR(900)                                                        FF  13
      COMMON /GND/ZRATI,ZRATI2,FRATI,CL,CH,SCRWL,SCRWR,NRADL,KSYMP,IFAR, FF  14
     1IPERF,T1,T2                                                       FF  15
      DIMENSION CAB(1), SAB(1), CONSX(2)                                FF  16
      EQUIVALENCE (CAB,ALP), (SAB,BET), (CONST,CONSX)                   FF  17
      DATA PI,TP,ETA/3.141592654,6.283185308,376.73/                    FF  18
      DATA CONSX/0.,-29.97922085/                                       FF  19
      PHX=-SIN(PHI)                                                     FF  20
      PHY=COS(PHI)                                                      FF  21
      ROZ=COS(THET)                                                     FF  22
      ROZS=ROZ                                                          FF  23
      THX=ROZ*PHY                                                       FF  24
      THY=-ROZ*PHX                                                      FF  25
      THZ=-SIN(THET)                                                    FF  26
      ROX=-THZ*PHY                                                      FF  27
      ROY=THZ*PHX                                                       FF  28
      IF (N.EQ.0) GO TO 20                                              FF  29
C                                                                       FF  30
C     LOOP FOR STRUCTURE IMAGE IF ANY                                   FF  31
C                                                                       FF  32
      DO 19 K=1,KSYMP                                                   FF  33
C                                                                       FF  34
C     CALCULATION OF REFLECTION COEFFECIENTS                            FF  35
C                                                                       FF  36
      IF (K.EQ.1) GO TO 4                                               FF  37
      IF (IPERF.NE.1) GO TO 1                                           FF  38
C                                                                       FF  39
C     FOR PERFECT GROUND                                                FF  40
C                                                                       FF  41
      RRV=-(1.,0.)                                                      FF  42
      RRH=-(1.,0.)                                                      FF  43
      GO TO 2                                                           FF  44
C                                                                       FF  45
C     FOR INFINITE PLANAR GROUND                                        FF  46
C                                                                       FF  47
1     ZRSIN=CSQRT(1.-ZRATI*ZRATI*THZ*THZ)                               FF  48
      RRV=-(ROZ-ZRATI*ZRSIN)/(ROZ+ZRATI*ZRSIN)                          FF  49
      RRH=(ZRATI*ROZ-ZRSIN)/(ZRATI*ROZ+ZRSIN)                           FF  50
2     IF (IFAR.LE.1) GO TO 3                                            FF  51
C                                                                       FF  52
C     FOR THE CLIFF PROBLEM, TWO REFLCTION COEFFICIENTS CALCULATED      FF  53
C                                                                       FF  54
      RRV1=RRV                                                          FF  55
      RRH1=RRH                                                          FF  56
      TTHET=TAN(THET)                                                   FF  57
      IF (IFAR.EQ.4) GO TO 3                                            FF  58
      ZRSIN=CSQRT(1.-ZRATI2*ZRATI2*THZ*THZ)                             FF  59
      RRV2=-(ROZ-ZRATI2*ZRSIN)/(ROZ+ZRATI2*ZRSIN)                       FF  60
      RRH2=(ZRATI2*ROZ-ZRSIN)/(ZRATI2*ROZ+ZRSIN)                        FF  61
      DARG=-TP*2.*CH*ROZ                                                FF  62
3     ROZ=-ROZ                                                          FF  63
```

```
      CCX=CIX
      CCY=CIY
      CCZ=CIZ
4     CIX=(0.,0.)
      CIY=(0.,0.)
      CIZ=(0.,0.)
C
C     LOOP OVER STRUCTURE SEGMENTS
C
      DO 17 I=1,N
      OMEGA=-(ROX*CAB(I)+ROY*SAB(I)+ROZ*SALP(I))
      EL=PI*SI(I)
      SILL=OMEGA*EL
      TOP=EL+SILL
      BOT=EL-SILL
      IF (ABS(OMEGA).LT.1.E-7) GO TO 5
      A=2.*SIN(SILL)/OMEGA
      GO TO 6
5     A=(2.-OMEGA*OMEGA*EL*EL/3.)*EL
6     IF (ABS(TOP).LT.1.E-7) GO TO 7
      TOO=SIN(TOP)/TOP
      GO TO 8
7     TOO=1.-TOP*TOP/6.
8     IF (ABS(BOT).LT.1.E-7) GO TO 9
      BOO=SIN(BOT)/BOT
      GO TO 10
9     BOO=1.-BOT*BOT/6.
10    B=EL*(BOO-TOO)
      C=EL*(BOO+TOO)
      RR=A*AIR(I)+B*BII(I)+C*CIR(I)
      RI=A*AII(I)-B*BIR(I)+C*CII(I)
      ARG=TP*(X(I)*ROX+Y(I)*ROY+Z(I)*ROZ)
      IF (K.EQ.2.AND.IFAR.GE.2) GO TO 11
      EXA=CMPLX(COS(ARG),SIN(ARG))*CMPLX(RR,RI)
C
C     SUMMATION FOR FAR FIELD INTEGRAL
C
      CIX=CIX+EXA*CAB(I)
      CIY=CIY+EXA*SAB(I)
      CIZ=CIZ+EXA*SALP(I)
      GO TO 17
C
C     CALCULATION OF IMAGE CONTRIBUTION IN CLIFF AND GROUND SCREEN
C     PROBLEMS.
C
11    DR=Z(I)*TTHET
C
C     SPECULAR POINT DISTANCE
C
      D=DR*PHY+X(I)
      IF (IFAR.EQ.2) GO TO 13
      D=SQRT(D*D+(Y(I)-DR*PHX)**2)
      IF (IFAR.EQ.3) GO TO 13
      IF ((SCRWL-D).LT.0.) GO TO 12
C
C     RADIAL WIRE GROUND SCREEN REFLECTION COEFFICIENT
C
      D=D+T2
      ZSCRN=T1*D*ALOG(D/T2)
      ZSCRN=(ZSCRN*ZRATI)/(ETA*ZRATI+ZSCRN)
      ZRSIN=CSQRT(1.-ZSCRN*ZSCRN*THZ*THZ)
      RRV=(ROZ+ZSCRN*ZRSIN)/(-ROZ+ZSCRN*ZRSIN)
      RRH=(ZSCRN*ROZ+ZRSIN)/(ZSCRN*ROZ-ZRSIN)
      GO TO 16
```

```
   12 IF (IFAR.EQ.4) GO TO 14
      IF (IFAR.EQ.5) D=DR*PHY+X(I)
   13 IF ((CL-D).LE.0.) GO TO 15
   14 RRV=RRV1
      RRH=RRH1
      GO TO 16
   15 RRV=RRV2
      RRH=RRH2
      ARG=ARG+DARG
   16 EXA=CMPLX(COS(ARG),SIN(ARG))*CMPLX(RR,RI)
C
C     CONTRIBUTION OF EACH IMAGE SEGMENT MODIFIED BY REFLECTION COEF.
C     FOR CLIFF AND GROUND SCREEN PROBLEMS
C
      TIX=EXA*CAB(I)
      TIY=EXA*SAB(I)
      TIZ=EXA*SALP(I)
      CDP=(TIX*PHX+TIY*PHY)*(RRH-RRV)
      CIX=CIX+TIX*RRV+CDP*PHX
      CIY=CIY+TIY*RRV+CDP*PHY
      CIZ=CIZ-TIZ*RRV
   17 CONTINUE
      IF (K.EQ.1) GO TO 19
      IF (IFAR.GE.2) GO TO 18
C
C     CALCULATION OF CONTRIBUTION OF STRUCTURE IMAGE FOR INFINITE GROUND
C
      CDP=(CIX*PHX+CIY*PHY)*(RRH-RRV)
      CIX=CCX+CIX*RRV+CDP*PHX
      CIY=CCY+CIY*RRV+CDP*PHY
      CIZ=CCZ-CIZ*RRV
      GO TO 19
   18 CIX=CIX+CCX
      CIY=CIY+CCY
      CIZ=CIZ+CCZ
   19 CONTINUE
      IF (M.GT.0) GO TO 21
      ETH=(CIX*THX+CIY*THY+CIZ*THZ)*CONST
      EPH=(CIX*PHX+CIY*PHY)*CONST
      RETURN
   20 CIX=(0.,0.)
      CIY=(0.,0.)
      CIZ=(0.,0.)
   21 ROZ=ROZS
C
C     ELECTRIC FIELD COMPONENTS
C
      RFL=-1.
      DO 25 IP=1,KSYMP
      RFL=-RFL
      RRZ=ROZ*RFL
      CALL FFLDS (ROX,ROY,RRZ,CUR(N+1),GX,GY,GZ)
      IF (IP.EQ.2) GO TO 22
      EX=GX
      EY=GY
      EZ=GZ
      GO TO 25
   22 IF (IPERF.NE.1) GO TO 23
      GX=-GX
      GY=-GY
      GZ=-GZ
      GO TO 24
   23 RRV=CSQRT(1.-ZRATI*ZRATI*THZ*THZ)
      RRH=ZRATI*ROZ
```

```
      RRH=(RRH-RRV)/(RRH+RRV)
      RRV=ZRATI*RRV
      RRV=-(ROZ-RRV)/(ROZ+RRV)
      ETH=(GX*PHX+GY*PHY)*(RRH-RRV)
      GX=GX*RRV+ETH*PHX
      GY=GY*RRV+ETH*PHY
      GZ=GZ*RRV
 24   EX=EX+GX
      EY=EY+GY
      EZ=EZ-GZ
 25   CONTINUE
      EX=EX+CIX*CONST
      EY=EY+CIY*CONST
      EZ=EZ+CIZ*CONST
      ETH=EX*THX+EY*THY+EZ*THZ
      EPH=EX*PHX+EY*PHY
      RETURN
      END
```

FFLDS

PURPOSE

To calculate the x, y, z components of the far electric field due to surface currents. The term $\exp(-jkr_0)/(r_0/\lambda)$ is omitted.

METHOD

The field is computed using the surface portion of equation (126) in Part I. With lengths normalized to the wavelength, the equation is

$$\bar{E}(\bar{r}_0) = \frac{j\eta}{2}\,\frac{\exp(-jkr_0)}{r_0/\lambda}\,(\hat{k}\hat{k} - \bar{\bar{I}}) \cdot \int_S \bar{J}_S(\bar{r})\exp(j\bar{k}\cdot\bar{r})\,dA/\lambda^2 \;,$$

where

$r_0 = |\bar{r}_0|$
$\hat{k} = \bar{r}_0/|\bar{r}_0|$
$k = 2\pi/\lambda$
$\bar{k} = k\hat{k}$
$\bar{J}_S$ = surface current on surface S
$\bar{\bar{I}}$ = identity dyad

The dot product with the dyad $\hat{k}\hat{k} - \bar{\bar{I}}$ results in the component of the integral

$$\bar{F}(\bar{r}_0) = \int_S \bar{J}_S(\bar{r})\exp(j\bar{k}\cdot\bar{r})\,dA/\lambda^2$$

transverse to $\hat{k}$. The integral is evaluated by summation over the patches with the current assumed constant over each patch.

SYMBOL DICTIONARY

ARG = $\bar{k}\cdot\bar{r}_i$, $\bar{r}_i$ = center of patch I
CONS = CONSX = $j\eta/2$
CT = $\exp(j\bar{k}\cdot\bar{r}_i)\,dA/\lambda^2$ at FL18
= $\hat{k}\cdot\bar{F}(\bar{r}_0)$ at FL24
EX, EY, EZ = x, y, z components of $\bar{F}(\bar{r}_0)$ at FL22
= $(r_0/\lambda)\exp(jkr_0)\bar{E}(\bar{r}_0)$ at FL27
I = array location of patch data
J = patch number
K = current array index

ROX, ROY, ROZ = x, y, z components of $\hat{k}$

S(I) = (area of patch I)/λ^2

SCUR = array containing surface current components

TPI = 2π

XS, YS, ZS = arrays containing center point coordinates of patches normalized to wavelength.

CODE LISTING

```
      SUBROUTINE FFLDS (ROX,ROY,ROZ,SCUR,EX,EY,EZ)
C     CALCULATES THE XYZ COMPONENTS OF THE ELECTRIC FIELD DUE TO
C     SURFACE CURRENTS
      COMPLEX CT,CONS,SCUR,EX,EY,EZ
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      DIMENSION XS(1), YS(1), ZS(1), S(1), SCUR(1), CONSX(2)
      EQUIVALENCE (XS,X), (YS,Y), (ZS,Z), (S,BI), (CONS,CONSX)
      DATA TPI/6.283185308/,CONSX/0.,188.365/
      EX=(0.,0.)
      EY=(0.,0.)
      EZ=(0.,0.)
      I=LD+1
      DO 1 J=1,M
      I=I-1
      ARG=TPI*(ROX*XS(I)+ROY*YS(I)+ROZ*ZS(I))
      CT=CMPLX(COS(ARG)*S(I),SIN(ARG)*S(I))
      K=3*J
      EX=EX+SCUR(K-2)*CT
      EY=EY+SCUR(K-1)*CT
      EZ=EZ+SCUR(K)*CT
    1 CONTINUE
      CT=ROX*EX+ROY*EY+ROZ*EZ
      EX=CONS*(CT*ROX-EX)
      EY=CONS*(CT*ROY-EY)
      EZ=CONS*(CT*ROZ-EZ)
      RETURN
      END
```

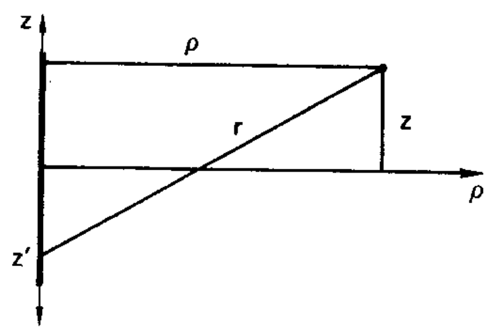
GF

PURPOSE

To supply values of the integrated function exp(jkr)/(kr) to the numerical integration routine INTX.

METHOD

The geometry parameters for integration over a segment are shown in the following diagram.



in which

$$r(z') = [\rho^2 + (z' - z)^2]^{1/2} .$$

If the field point (ρ,z) is not on the source segment, the integrand value is

$$G(z') = \frac{\exp[jkr(z')]}{kr(z')} .$$

If the field point is on the source segment (ρ = 0, z = 0), the integrand value is

$$G(z') = \frac{\exp[jkr(z')] - 1}{kr(z')} .$$

In the latter case, if kr is less than 0.2, then (cos kr)/kr is evaluated by the first three terms of its Taylor's series to reduce numerical error.

SYMBOL DICTIONARY

CO = real part of G(z')

COS = external function (cosine)

IJ = flag to indicate when field point is on source segment (by IJ = 0)

RK = kr

RKB2 = $(k\rho)^2$

SI = imaginary part of G(z')

SIN = external function (sine)

SQRT = external function (square root)

ZDK = kz' - kz

ZK = kz'

ZPK = kz

CONSTANTS

$$\left.\begin{array}{l}-1.38888889E-3 \\ 4.16666667E-2 \\ 0.5\end{array}\right\} = \text{constants in series for } (\cos kr - 1)/kr$$

CODE LISTING

```
      SUBROUTINE GF (ZK,CO,SI)                                          GF   1
C                                                                       GF   2
C     GF COMPUTES THE INTEGRAND EXP(JKR)/(KR) FOR NUMERICAL INTEGRATION. GF   3
C                                                                       GF   4
      COMMON /TMI/ ZPK,RKB2,IJ                                          GF   5
      ZDK=ZK-ZPK                                                        GF   6
      RK=SQRT(RKB2+ZDK*ZDK)                                             GF   7
      SI=SIN(RK)/RK                                                     GF   8
      IF (IJ) 1,2,1                                                     GF   9
1     CO=COS(RK)/RK                                                     GF  10
      RETURN                                                            GF  11
2     IF (RK.LT..2) GO TO 3                                             GF  12
      CO=(COS(RK)-1.)/RK                                                GF  13
      RETURN                                                            GF  14
3     RKS=RK*RK                                                         GF  15
      CO=((-1.38888889E-3*RKS+4.16666667E-2)*RKS-.5)*RK                 GF  16
      RETURN                                                            GF  17
      END                                                               GF  18-
```

PURPOSE

To read the NGF file and store parameters in the proper arrays.

METHOD

GI22	Miscellaneous parameters are read.
GI30 - GI48	Segment coordinates were converted to the form involving the segment center, segment length, and orientation (see Section III, COMMON/DATA/) with dimensions of wavelength. They must be converted back to the coordinates of the segment ends so that subroutine CONNECT can locate connections. Dimensions are converted to meters.
GI52 - GI62	Patch coordinates are converted from units of wavelength to meters since they will be scaled back to wavelengths along with the new segments and patches.
GI63	Matrix blocking parameters are read.
GI64	Interpolation tables for the Sommerfeld integrals are read if the Sommerfeld/Norton ground treatment was used.
GI74	Matrix A_F is read for in-core storage (ICASE = 1 or 2).
GI78 - GI81	A_F is read for ICASE = 4.
GI83 - GI88	A_F is read for ICASE = 3 or 5.
GI92 - GI113	A heading summarizing the NGF file is printed.

SYMBOL DICTIONARY

DX	= half segment length (meters)
IGFL	= file number for NGF file
IOUT	= number of elements in matrix
IPRT	= 1 to print coordinates of ends of segments
NBL2	= two times number of blocks in matrix A_F (since A_F is stored twice, in ascending and descending order)
NEQ	= order of the NGF matrix
NOP	= number of symmetric sections
NPEQ	= number of unknowns for a symmetric section
XI, YI, ZI	= coordinates of the center of a segment or patch

GFIL

```
      SUBROUTINE GFIL (IPRT)                                            GI   1
C                                                                       GI   2
C     GFIL READS THE N.G.F. FILE                                        GI   3
C                                                                       GI   4
      COMPLEX CM,SSX,ZRATI,ZRATI2,T1,ZARRAY,AR1,AR2,AR3,EPSCF,FRATI     GI   5
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300 GI   6
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX( GI   7
     2300),WLAM,IPSYM                                                   GI   8
      COMMON /CMB/ CM(4000)                                             GI   9
      COMMON /ANGL/ SALP(300)                                           GI  10
      COMMON /GND/ZRATI,ZRATI2,FRATI,CL,CH,SCRWL,SCRWR,NRADL,KSYMP,IFAR, GI  11
     1IPERF,T1,T2                                                       GI  12
      COMMON /GGRID/ AR1(11,10,4),AR2(17,5,4),AR3(9,8,4),EPSCF,DXA(3),DY GI  13
     1A(3),XSA(3),YSA(3),NXA(3),NYA(3)                                  GI  14
      COMMON /MATPAR/ ICASE,NBLOKS,NPBLK,NLAST,NBLSYM,NPSYM,NLSYM,IMAT,I GI  15
     1CASX,NBBX,NPBX,NLBX,NBBL,NPBL,NLBL                                GI  16
      COMMON /SMAT/ SSX(16,16)                                          GI  17
      COMMON /ZLOAD/ ZARRAY(300),NLOAD,NLODF                            GI  18
      COMMON /SAVE/ IP(600),KCOM,COM(13,5),EPSR,SIG,SCRWLT,SCRWRT,FMHZ  GI  19
      DATA IGFL/20/                                                     GI  20
      REWIND IGFL                                                       GI  21
      READ (IGFL) N1,NP,M1,MP,WLAM,FMHZ,IPSYM,KSYMP,IPERF,NRADL,EPSR,SIG GI  22
     1,SCRWLT,SCRWRT,NLODF,KCOM                                         GI  23
      N=N1                                                              GI  24
      M=M1                                                              GI  25
      N2=N1+1                                                           GI  26
      M2=M1+1                                                           GI  27
      IF (N1.EQ.0) GO TO 2                                              GI  28
C     READ SEG. DATA AND CONVERT BACK TO END COORD. IN UNITS OF METERS  GI  29
      READ (IGFL) (X(I),I=1,N1),(Y(I),I=1,N1),(Z(I),I=1,N1)             GI  30
      READ (IGFL) (SI(I),I=1,N1),(BI(I),I=1,N1),(ALP(I),I=1,N1)         GI  31
      READ (IGFL) (BET(I),I=1,N1),(SALP(I),I=1,N1)                      GI  32
      READ (IGFL) (ICON1(I),I=1,N1),(ICON2(I),I=1,N1)                   GI  33
      READ (IGFL) (ITAG(I),I=1,N1)                                      GI  34
      IF (NLODF.NE.0) READ (IGFL) (ZARRAY(I),I=1,N1)                    GI  35
      DO 1 I=1,N1                                                       GI  36
      XI=X(I)*WLAM                                                      GI  37
      YI=Y(I)*WLAM                                                      GI  38
      ZI=Z(I)*WLAM                                                      GI  39
      DX=SI(I)*.5*WLAM                                                  GI  40
      X(I)=XI-ALP(I)*DX                                                 GI  41
      Y(I)=YI-BET(I)*DX                                                 GI  42
      Z(I)=ZI-SALP(I)*DX                                                GI  43
      SI(I)=XI+ALP(I)*DX                                                GI  44
      ALP(I)=YI+BET(I)*DX                                               GI  45
      BET(I)=ZI+SALP(I)*DX                                              GI  46
      BI(I)=BI(I)*WLAM                                                  GI  47
1     CONTINUE                                                          GI  48
2     IF (M1.EQ.0) GO TO 4                                              GI  49
      J=LD-M1+1                                                         GI  50
C     READ PATCH DATA AND CONVERT TO METERS                             GI  51
      READ (IGFL) (X(I),I=J,LD),(Y(I),I=J,LD),(Z(I),I=J,LD)             GI  52
      READ (IGFL) (SI(I),I=J,LD),(BI(I),I=J,LD),(ALP(I),I=J,LD)         GI  53
      READ (IGFL) (BET(I),I=J,LD),(SALP(I),I=J,LD)                      GI  54
      READ (IGFL) (ICON1(I),I=J,LD),(ICON2(I),I=J,LD)                   GI  55
      READ (IGFL) (ITAG(I),I=J,LD)                                      GI  56
      DX=WLAM*WLAM                                                      GI  57
      DO 3 I=J,LD                                                       GI  58
      X(I)=X(I)*WLAM                                                    GI  59
      Y(I)=Y(I)*WLAM                                                    GI  60
      Z(I)=Z(I)*WLAM                                                    GI  61
3     BI(I)=BI(I)*DX                                                    GI  62
4     READ (IGFL) ICASE,NBLOKS,NPBLK,NLAST,NBLSYM,NPSYM,NLSYM,IMAT      GI  63
      IF (IPERF.EQ.2) READ (IGFL) AR1,AR2,AR3,EPSCF,DXA,DYA,XSA,YSA,NXA, GI  64
```

```
     1NYA
      NEQ=N1+2*M1
      NPEQ=NP+2*MP
      NOP=NEQ/NPEQ
      IF (NOP.GT.1) READ (IGFL) ((SSX(I,J),I=1,NOP),J=1,NOP)
      READ (IGFL) (IP(I),I=1,NEQ),COM
C     READ MATRIX A AND WRITE TAPE13 FOR OUT OF CORE
      IF (ICASE.GT.2) GO TO 5
      IOUT=NEQ*NPEQ
      READ (IGFL) (CM(I),I=1,IOUT)
      GO TO 10
5     REWIND 13
      IF (ICASE.NE.4) GO TO 7
      IOUT=NPEQ*NPEQ
      DO 6 K=1,NOP
      READ (IGFL) (CM(J),J=1,IOUT)
6     WRITE (13) (CM(J),J=1,IOUT)
      GO TO 9
7     IOUT=NPSYM*NPEQ*2
      NBL2=2*NBLSYM
      DO 8 IOP=1,NOP
      DO 8 I=1,NBL2
      CALL BLCKIN (CM,IGFL,1,IOUT,1,206)
8     CALL BLCKOT (CM,13,1,IOUT,1,205)
9     REWIND 13
10    REWIND IGFL
C     PRINT N.G.F. HEADING
      PRINT 16
      PRINT 14
      PRINT 14
      PRINT 17
      PRINT 18, N1,M1
      IF (NOP.GT.1) PRINT 19, NOP
      PRINT 20, IMAT,ICASE
      IF (ICASE.LT.3) GO TO 11
      NBL2=NEQ*NPEQ
      PRINT 21, NBL2
11    PRINT 22, FMHZ
      IF (KSYMP.EQ.2.AND.IPERF.EQ.1) PRINT 23
      IF (KSYMP.EQ.2.AND.IPERF.EQ.0) PRINT 27
      IF (KSYMP.EQ.2.AND.IPERF.EQ.2) PRINT 28
      IF (KSYMP.EQ.2.AND.IPERF.NE.1) PRINT 24, EPSR,SIG
      PRINT 17
      DO 12 J=1,KCOM
12    PRINT 15, (COM(I,J),I=1,13)
      PRINT 17
      PRINT 14
      PRINT 14
      PRINT 16
      IF (IPRT.EQ.0) RETURN
      PRINT 25
      DO 13 I=1,N1
13    PRINT 26, I,X(I),Y(I),Z(I),SI(I),ALP(I),BET(I)
      RETURN
C
14    FORMAT (5X,50H**************************************************,3
     14H**********************************)
15    FORMAT (5X,3H** ,13A6,3H **)
16    FORMAT (////)
17    FORMAT (5X,2H**,80X,2H**)
18    FORMAT (5X,29H** NUMERICAL GREEN'S FUNCTION,53X,2H**,/,5X,17H** NO
     1. SEGMENTS =,I4,10X,13HNO. PATCHES =,I4,34X,2H**)
19    FORMAT (5X,27H** NO. SYMMETRIC SECTIONS =,I4,51X,2H**)
20    FORMAT (5X,34H** N.G.F. MATRIX -  CORE STORAGE =,I7,23H COMPLEX NU
```

```
     1MBERS,  CASE,I2,16X,2H**)
  21  FORMAT (5X,2H**,19X,13HMATRIX SIZE =,I7,16H COMPLEX NUMBERS,25X,2H
     1**)
  22  FORMAT (5X,14H** FREQUENCY =,E12.5,5H MHZ.,51X,2H**)
  23  FORMAT (5X,17H** PERFECT GROUND,65X,2H**)
  24  FORMAT (5X,44H** GROUND PARAMETERS - DIELECTRIC CONSTANT =,E12.5,2
     16X,2H**,/,5X,2H**,21X,14HCONDUCTIVITY =,E12.5,8H MHOS/M.,25X,2H**)
  25  FORMAT (39X,31HNUMERICAL GREEN'S FUNCTION DATA,/,41X,27HCOORDINATE
     1S OF SEGMENT ENDS,/,51X,8H(METERS),/,5X,4HSEG.,11X,19H- - - END ON
     2E - - -,26X,19H- - - END TWO - - -,/,6X,3HNO.,6X,1HX,14X,1HY,14X,1
     3HZ,14X,1HX,14X,1HY,14X,1HZ)
  26  FORMAT (1X,I7,6E15.6)
  27  FORMAT (5X,55H** FINITE GROUND.  REFLECTION COEFFICIENT APPROXIMAT
     1ION,27X,2H**)
  28  FORMAT (5X,38H** FINITE GROUND.  SOMMERFELD SOLUTION,44X,2H**)
      END
```

GFLD

PURPOSE

To compute the electric field at intermediate distances from a radiating structure over ground, including the surface-wave field component.

METHOD

Approximate expressions for the field of a horizontal or vertical current element over a ground plane were derived by K. A. Norton (ref. 2). These expressions are used to evaluate the field of each segment in a structure and the components summed for the total field of the structure. To evaluate Norton's expressions for segment i, a local coordinate system (x', y', z') is defined (fig. 6a) with origin on the ground plane and the vertical z axis passing through segment i. In the (x, y, z) coordinate system (fig. 6 b) the location and orientation of segment i are

$$\bar{r}_i = x_i\hat{x} + y_i\hat{y} + z_i\hat{z}$$

$$\hat{i} = \cos\alpha \cos\beta\ \hat{x} + \cos\alpha \sin\beta\ \hat{y} + \sin\alpha\ \hat{z}$$

and the field observation point is at (ρ, ϕ, z). The origin of the primed coordinate system is at (x_i, y_i, 0) in the umprimed coordinates, and the x' axis is along the projection of the segment on the ground plane.

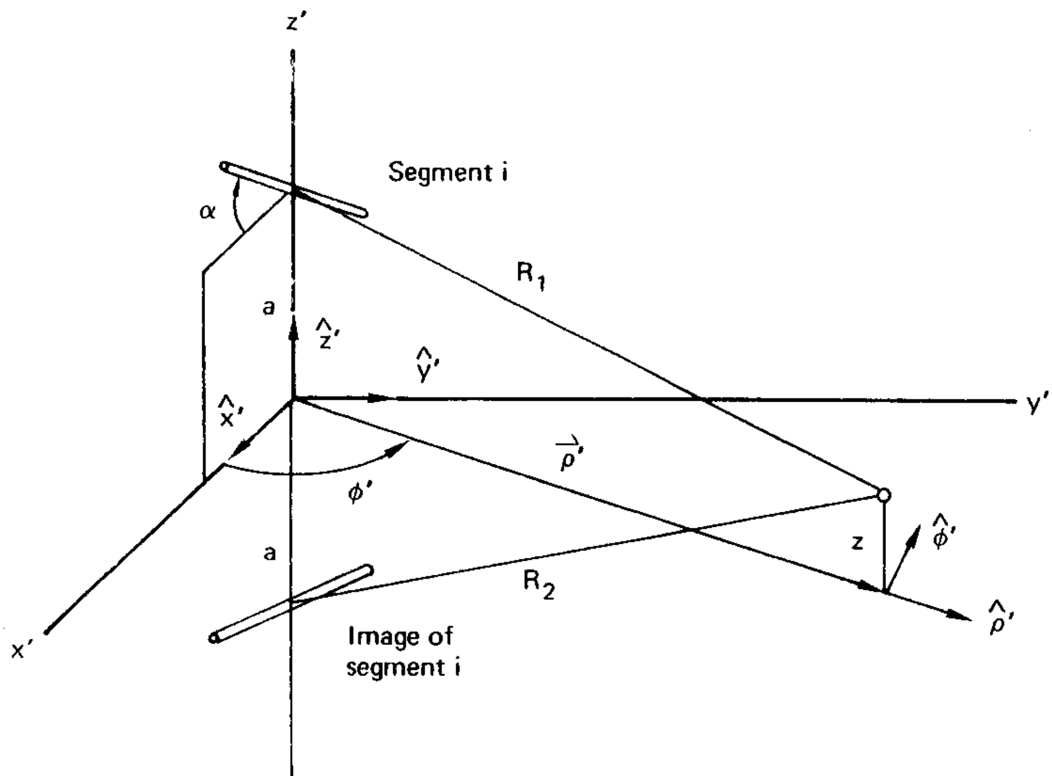
Norton's expressions give the electric field in ρ', ϕ', and z' components for infinitesimal current elements either vertical or horizontal, and directed along the x' axis. To evaluate the field of a segment, the segment current is decomposed into horizontal and vertical components, and the fields of the infinitesimal current elements are integrated over the segment. Each field component for the infinitesimal current element has the form

$$E_\Lambda(\rho', \phi', z') = F_1(\rho', \phi', z')\exp(-jkR_1) + F_2(\rho', \phi', z')\exp(-jkR_2)\ ,$$

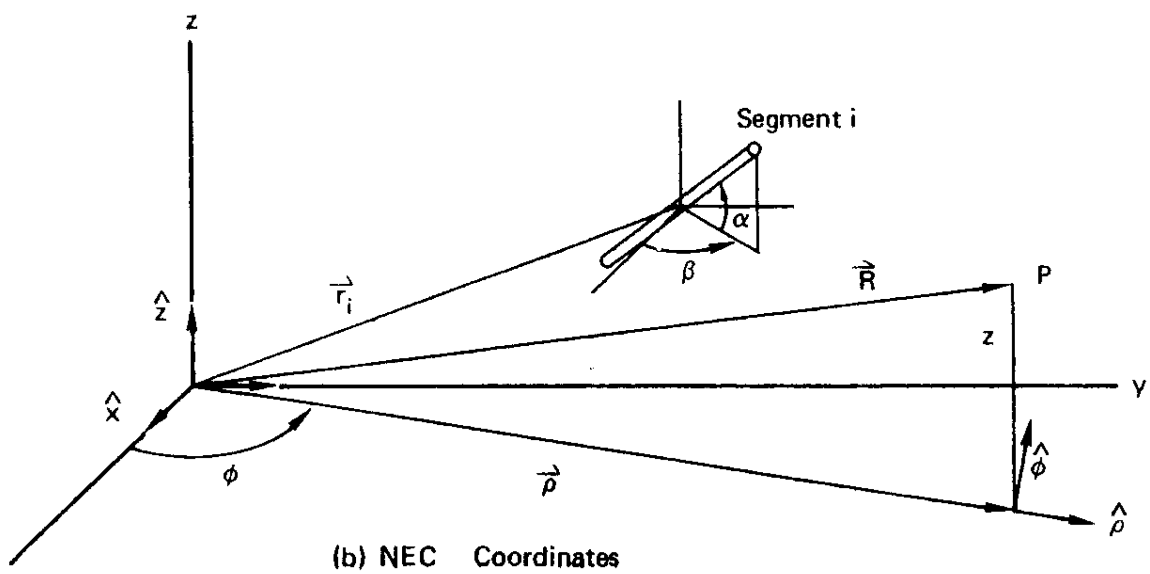
for

$$R_1 = |\bar{R}_1|$$

$$R_2 = |\bar{R}_2|$$



(a) Norton's Coordinates



(b) NEC Coordinates

Figure 6. Coordinate Systems Used to Evaluate Norton's Expressions for the Ground Wave Fields in the NEC Program.

where F_1 and F_2 are algebraic functions of R_1 and R_2 and can be considered constant for integration over the segment as long as R_1 and R_2 are much greater than the segment length. To integrate the exponential factors over the segment, R_1 and R_2 are approximated as

$$R_1 \approx R - \hat{R}_1 \cdot (\bar{r}_i + \hat{i}s)$$

$$R_2 \approx R - \hat{R}_2 \cdot (\bar{r}_i' + \hat{i}'s)$$

where $R = |\bar{R}|$, $\hat{R}_1 = \bar{R}_1/|\bar{R}_1|$, $\hat{R}_2 = \bar{R}_2/|\bar{R}_2|$; $\bar{r}_i'$, $\hat{i}'$ = position and orientation of image of segment i, and s = variable of length along the segment (s = 0 at segment center). The current on the segment is

$$I_i(s) = A_i + B_i \sin ks + C_i \cos ks .$$

With F_1 and F_2 considered constant, each vector component of the field produced by segment i involves an integral of the form

$$E = F_1' \int_{-\Delta/2\lambda}^{\Delta/2\lambda} \frac{I_i(s)}{\lambda} \exp(-jks\omega)\, d\frac{s}{\lambda} + F_2' \int_{-\Delta/2\lambda}^{\Delta/2\lambda} \frac{I_i(s)}{\lambda} \exp(-jks\omega')\, d(s/\lambda) ,$$

where

$$F_1' = \lambda^2 F_1 \exp[-jk(R - \hat{R}_1 \cdot \bar{r}_i)]$$

$$F_2' = \lambda^2 F_2 \exp[-jk(R - \hat{R}_2 \cdot \bar{r}_i')]$$

$$\omega = -\hat{R}_1 \cdot \hat{i}$$

$$\omega' = -\hat{R}_2 \cdot \hat{i}'$$

Δ = segment length

The integrals can be evaluated as

$$G_1 = \int_{-\Delta/2\lambda}^{\Delta/2\lambda} \frac{I_i(s)}{\lambda} \exp(-j2\pi\, \omega\, s/\lambda)\, d\frac{s}{\lambda}$$

$$2\pi G_1 = \frac{A_i}{\lambda}\,\frac{2 \sin \pi\,\omega\,d}{\omega}$$

$$- j\,\frac{B_i}{\lambda}\left\{\frac{\sin[\pi\,(1-\omega)d]}{(1-\omega)} - \frac{\sin[\pi\,(1+\omega)d]}{(1+\omega)}\right\}$$

$$+ \frac{C_i}{\lambda}\left\{\frac{\sin[\pi\,(1-\omega)d]}{(1-\omega)} + \frac{\sin[\pi\,(1+\omega)d]}{(1+\omega)}\right\}$$

where $d = \Lambda/\lambda$. The integral for G_2 (the coefficient of F_2') is the same with $\bar{r}_i$ and $\hat{i}$ reflected in the ground plane. The terms G_1 and G_2 and other necessary quantities are passed to subroutine GWAVE through COMMON/GWAV/. GWAVE returns the field components

E_ρ^v = ρ' component of field due to vertical current component

E_z^v = z component of field due to vertical current component

E_ρ^h = ρ' component of field due to horizontal current component

E_ϕ^h = ϕ' component of field due to horizontal current component

E_z^h = z component of field due to horizontal current component

The common factor exp(-jkR) occurring in F_1' and F_2' is omitted from the field components and included in the total field after summation.

These field components are then combined to form the total field in x, y, z components and summed for each segment. The field is finally converted to r, θ, ϕ components in a spherical coordinate system coinciding with the x, y, z coordinate system.

The approximations involved in the calculation of the surface wave are valid to second order in u^2, where

$u = k/k_2$

k = wave number in free space

k_2 = wave number in ground medium

The approximations are valid for practical ground parameters. To ensure that the expressions are not used in an invalid range, however, the surface wave is not computed if $|u|$ is greater than 0.5. Rather, subroutine FFLD is called, and the resulting space wave is multiplied by the range factor exp(-jkR)/(R/λ). The radial field component will be zero in this case. FFLD is also called if R/λ is greater than 10^5, or if there is no ground present.

SYMBOL DICTIONARY

A	=	coefficient of A_i/λ in $2\pi G_1$ and $2\pi G_2$
ABS	=	external routine (absolute value)
ARG	=	argument of exp( ) for phase factor
ATAN	=	external routine (arctangent)
B	=	coefficient of B_i/λ in $2\pi G_1$ and $2\pi G_2$
BOO	=	sin (BOT)/BOT
BOT	=	$\pi(1 - \omega)d$
C	=	coefficient of C_i/λ in $2\pi G_1$ and $2\pi G_2$
CAB(I)	=	cos α cos β for segment I
CABS	=	external routine (magnitude of complex number)
CALP	=	cos α
CBET	=	cos β
CIX, CIY, CIZ	=	x, y, z components in summation for field
CMPLX	=	external routine (forms complex number)
COS	=	external routine (cosine)
CPH	=	$\cos \phi'$
DX, DY, DZ	=	x, y, z components of $\hat{i}$
EL	=	πd
EPH	=	E^h_ϕ or $E^h_\phi \cos \alpha$ (ϕ' component of total field of segment i)
EPI	=	φ component of field of structure
ERD	=	R component of field of structure
ERH	=	E^h_ρ and ρ' component of total field of segment i
ERV	=	E^v_ρ
ETH	=	θ component of field of structure
EX	=	x component of field for segment i
EXA	=	phase factor at GD30 and GD130: $G_1 \exp(jk\hat{R}_1 \cdot \bar{r}_i)$ or $G_2 \exp(jk\hat{R}_2 \cdot \bar{r}'_i)$ at GD109
EY	=	y component of field for segment i
EZH	=	E^h_z and z component of total field of segment i
EZV	=	E^v_z

FFLD = external routine (computes space wave)
GWAVE = external routine (computes E_ρ^v, E_ρ^h ,...)
I = DO loop index (i)
K = DO loop index (loop over segment and image)
KSYMP = 1 if ground is present; 0 otherwise
OMEGA = ω
PHI = ϕ
PHX = x component of $\hat{\phi}$
PHY = y component of $\hat{\phi}$
PI = π
R = R/λ
RFL = sign factor to reflect segment coordinates in ground
RHO = ρ/λ
RHP = ρ'/λ
RHS = $(\rho'/\lambda)^2$
RHX = x component of $\hat{\rho}'$
RHY = y component of $\hat{\rho}'$
RI = imaginary part of $2\pi G_1$ or $2\pi G_2$
RIX = x component of $\bar{R}_1/\lambda$ or $\bar{R}_2/\lambda$
RIY = y component of $\bar{R}_1/\lambda$ or $\bar{R}_2/\lambda$
RIZ = z component of $\bar{R}_1/\lambda$ or $\bar{R}_2/\lambda$
RNX, RNY, RNZ = x, y, z components of $\hat{R}_1$ or $\hat{R}_2$ or $\hat{R}$
RR = real part of $2\pi G_1$ or $2\pi G_2$
RX = x component of $\bar{\rho}/\lambda$
RXYZ = R_1/λ or R_2/λ (for s = 0)
RY = y component of $\bar{\rho}/\lambda$
RZ = z/λ
SAB(I) = cos α sin β
SBET = sin β
SILL = $\pi d\omega$
SIN = external routine (sine)
SPH = sin ϕ'

SQRT = external routine (square root)

THET = θ in spherical coordinate system

THX = x component of $\hat{\theta}$

THY = y component of $\hat{\theta}$

THZ = z component of $\hat{\theta}$

TOO = sin (TOP)/TOP

TOP = $\pi(1 + \omega)d$

TP = 2π

U = u

UX = u

U2 = u^2

XX1 = $G_1 \exp(jk\hat{R}_1 \cdot \bar{r}_i)$

XX2 = $G_2 \exp(jk\hat{R}_2 \cdot \bar{r}'_i)$

CONSTANTS

1.E-20 = tolerance in test for zero

1.E-7 = tolerance in test for zero

1.E-6 = tolerance in test for zero

0.5 = upper limit for $|u|$

3.141592654 = π

6.283185308 = 2π

1.E+5 = upper limit for R/λ

```
      SUBROUTINE GFLD (RHO,PHI,RZ,ETH,EPI,ERD,UX,KSYMP)
C
C     GFLD COMPUTES THE RADIATED FIELD INCLUDING GROUND WAVE.
C
      COMPLEX CUR,EPI,CIX,CIY,CIZ,EXA,XX1,XX2,U,U2,ERV,EZV,ERH,EPH
      COMPLEX EZH,EX,EY,ETH,UX,ERD
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /ANGL/ SALP(300)
      COMMON /CRNT/ AIR(300),AII(300),BIR(300),BII(300),CIR(300),CII(300
     1),CUR(900)
      COMMON /GWAV/ U,U2,XX1,XX2,R1,R2,ZMH,ZPH
      DIMENSION CAB(1), SAB(1)
      EQUIVALENCE (CAB(1),ALP(1)), (SAB(1),BET(1))
      DATA PI,TP/3.141592654,6.283185308/
      R=SQRT(RHO*RHO+RZ*RZ)
      IF (KSYMP.EQ.1) GO TO 1
      IF (CABS(UX).GT..5) GO TO 1
      IF (R.GT.1.E5) GO TO 1
      GO TO 4
C
C     COMPUTATION OF SPACE WAVE ONLY
C
1     IF (RZ.LT.1.E-20) GO TO 2
      THET=ATAN(RHO/RZ)
      GO TO 3
2     THET=PI*.5
3     CALL FFLD (THET,PHI,ETH,EPI)
      ARG=-TP*R
      EXA=CMPLX(COS(ARG),SIN(ARG))/R
      ETH=ETH*EXA
      EPI=EPI*EXA
      ERD=(0.,0.)
      RETURN
C
C     COMPUTATION OF SPACE AND GROUND WAVES.
C
4     U=UX
      U2=U*U
      PHX=-SIN(PHI)
      PHY=COS(PHI)
      RX=RHO*PHY
      RY=-RHO*PHX
      CIX=(0.,0.)
      CIY=(0.,0.)
      CIZ=(0.,0.)
C
C     SUMMATION OF FIELD FROM INDIVIDUAL SEGMENTS
C
      DO 17 I=1,N
      DX=CAB(I)
      DY=SAB(I)
      DZ=SALP(I)
      RIX=RX-X(I)
      RIY=RY-Y(I)
      RHS=RIX*RIX+RIY*RIY
      RHP=SQRT(RHS)
      IF (RHP.LT.1.E-6) GO TO 5
      RHX=RIX/RHP
      RHY=RIY/RHP
      GO TO 6
5     RHX=1.
      RHY=0.
```

```
    6 CALP=1.-DZ*DZ                                                     GD  65
      IF (CALP.LT.1.E-6) GO TO 7                                        GD  66
      CALP=SQRT(CALP)                                                   GD  67
      CBET=DX/CALP                                                      GD  68
      SBET=DY/CALP                                                      GD  69
      CPH=RHX*CBET+RHY*SBET                                             GD  70
      SPH=RHY*CBET-RHX*SBET                                             GD  71
      GO TO 8                                                           GD  72
    7 CPH=RHX                                                           GD  73
      SPH=RHY                                                           GD  74
    8 EL=PI*SI(I)                                                       GD  75
      RFL=-1.                                                           GD  76
C                                                                       GD  77
C     INTEGRATION OF (CURRENT)*(PHASE FACTOR) OVER SEGMENT AND IMAGE FOR GD  78
C     CONSTANT, SINE, AND COSINE CURRENT DISTRIBUTIONS                  GD  79
C                                                                       GD  80
      DO 16 K=1,2                                                       GD  81
      RFL=-RFL                                                          GD  82
      RIZ=RZ-Z(I)*RFL                                                   GD  83
      RXYZ=SQRT(RIX*RIX+RIY*RIY+RIZ*RIZ)                                GD  84
      RNX=RIX/RXYZ                                                      GD  85
      RNY=RIY/RXYZ                                                      GD  86
      RNZ=RIZ/RXYZ                                                      GD  87
      OMEGA=-(RNX*DX+RNY*DY+RNZ*DZ*RFL)                                 GD  88
      SILL=OMEGA*EL                                                     GD  89
      TOP=EL+SILL                                                       GD  90
      BOT=EL-SILL                                                       GD  91
      IF (ABS(OMEGA).LT.1.E-7) GO TO 9                                  GD  92
      A=2.*SIN(SILL)/OMEGA                                              GD  93
      GO TO 10                                                          GD  94
    9 A=(2.-OMEGA*OMEGA*EL*EL/3.)*EL                                    GD  95
   10 IF (ABS(TOP).LT.1.E-7) GO TO 11                                   GD  96
      TOO=SIN(TOP)/TOP                                                  GD  97
      GO TO 12                                                          GD  98
   11 TOO=1.-TOP*TOP/6.                                                 GD  99
   12 IF (ABS(BOT).LT.1.E-7) GO TO 13                                   GD 100
      BOO=SIN(BOT)/BOT                                                  GD 101
      GO TO 14                                                          GD 102
   13 BOO=1.-BOT*BOT/6.                                                 GD 103
   14 B=EL*(BOO-TOO)                                                    GD 104
      C=EL*(BOO+TOO)                                                    GD 105
      RR=A*AIR(I)+B*BII(I)+C*CIR(I)                                     GD 106
      RI=A*AII(I)-B*BIR(I)+C*CII(I)                                     GD 107
      ARG=TP*(X(I)*RNX+Y(I)*RNY+Z(I)*RNZ*RFL)                           GD 108
      EXA=CMPLX(COS(ARG),SIN(ARG))*CMPLX(RR,RI)/TP                      GD 109
      IF (K.EQ.2) GO TO 15                                              GD 110
      XX1=EXA                                                           GD 111
      R1=RXYZ                                                           GD 112
      ZMH=RIZ                                                           GD 113
      GO TO 16                                                          GD 114
   15 XX2=EXA                                                           GD 115
      R2=RXYZ                                                           GD 116
      ZPH=RIZ                                                           GD 117
   16 CONTINUE                                                          GD 118
C                                                                       GD 119
C     CALL SUBROUTINE TO COMPUTE THE FIELD OF SEGMENT INCLUDING GROUND  GD 120
C     WAVE.                                                             GD 121
C                                                                       GD 122
      CALL GWAVE (ERV,EZV,ERH,EZH,EPH)                                  GD 123
      ERH=ERH*CPH*CALP+ERV*DZ                                           GD 124
      EPH=EPH*SPH*CALP                                                  GD 125
      EZH=EZH*CPH*CALP+EZV*DZ                                           GD 126
      EX=ERH*RHX-EPH*RHY                                                GD 127
      EY=ERH*RHY+EPH*RHX                                                GD 128
```

```
      CIX=CIX+EX                                                        GD 129
      CIY=CIY+EY                                                        GD 130
   17 CIZ=CIZ+EZH                                                       GD 131
      ARG=-TP*R                                                         GD 132
      EXA=CMPLX(COS(ARG),SIN(ARG))                                      GD 133
      CIX=CIX*EXA                                                       GD 134
      CIY=CIY*EXA                                                       GD 135
      CIZ=CIZ*EXA                                                       GD 136
      RNX=RX/R                                                          GD 137
      RNY=RY/R                                                          GD 138
      RNZ=RZ/R                                                          GD 139
      THX=RNZ*PHY                                                       GD 140
      THY=-RNZ*PHX                                                      GD 141
      THZ=-RHO/R                                                        GD 142
      ETH=CIX*THX+CIY*THY+CIZ*THZ                                       GD 143
      EPI=CIX*PHX+CIY*PHY                                               GD 144
      ERD=CIX*RNX+CIY*RNY+CIZ*RNZ                                       GD 145
      RETURN                                                            GD 146
      END                                                               GD 147-
```

GFOUT

PURPOSE

To write the NGF file.

METHOD

The contents of the COMMON blocks in GFOUT are written to file 20. If ICASE is 3 or 5 the blocks of the LU decomposition of matrix A are on file 13 in ascending order and on file 14 in descending order. Both files are written to file 20.

SYMBOL DICTIONARY

IGFL = NGF file number

IOUT = number of elements in matrix

NEQ = order of matrix A

NOP = number of symmetric sections

NPEQ = number of unknowns for a symmetric section

```
      SUBROUTINE GFOUT
C
C     WRITE N.G.F. FILE
C
      COMPLEX CM,SSX,ZRATI,ZRATI2,T1,ZARRAY,AR1,AR2,AR3,EPSCF,FRATI
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /CMB/ CM(4000)
      COMMON /ANGL/ SALP(300)
      COMMON /GND/ZRATI,ZRATI2,FRATI,CL,CH,SCRWL,SCRWR,NRADL,KSYMP,IFAR,
     1IPERF,T1,T2
      COMMON /GGRID/ AR1(11,10,4),AR2(17,5,4),AR3(9,8,4),EPSCF,DXA(3),DY
     1A(3),XSA(3),YSA(3),NXA(3),NYA(3)
      COMMON /MATPAR/ ICASE,NBLOKS,NPBLK,NLAST,NBLSYM,NPSYM,NLSYM,IMAT,I
     1CASX,NBBX,NPBX,NLBX,NBBL,NPBL,NLBL
      COMMON /SMAT/ SSX(16,16)
      COMMON /ZLOAD/ ZARRAY(300),NLOAD,NLODF
      COMMON /SAVE/ IP(600),KCOM,COM(13,5),EPSR,SIG,SCRWLT,SCRWRT,FMHZ
      DATA IGFL/20/
      NEQ=N+2*M
      NPEQ=NP+2*MP
      NOP=NEQ/NPEQ
      WRITE (IGFL) N,NP,M,MP,WLAM,FMHZ,IPSYM,KSYMP,IPERF,NRADL,EPSR,SIG,
     1SCRWLT,SCRWRT,NLOAD,KCOM
      IF (N.EQ.0) GO TO 1
      WRITE (IGFL) (X(I),I=1,N),(Y(I),I=1,N),(Z(I),I=1,N)
      WRITE (IGFL) (SI(I),I=1,N),(BI(I),I=1,N),(ALP(I),I=1,N)
      WRITE (IGFL) (BET(I),I=1,N),(SALP(I),I=1,N)
      WRITE (IGFL) (ICON1(I),I=1,N),(ICON2(I),I=1,N)
      WRITE (IGFL) (ITAG(I),I=1,N)
      IF (NLOAD.GT.0) WRITE (IGFL) (ZARRAY(I),I=1,N)
1     IF (M.EQ.0) GO TO 2
      J=LD-M+1
      WRITE (IGFL) (X(I),I=J,LD),(Y(I),I=J,LD),(Z(I),I=J,LD)
      WRITE (IGFL) (SI(I),I=J,LD),(BI(I),I=J,LD),(ALP(I),I=J,LD)
      WRITE (IGFL) (BET(I),I=J,LD),(SALP(I),I=J,LD)
      WRITE (IGFL) (ICON1(I),I=J,LD),(ICON2(I),I=J,LD)
      WRITE (IGFL) (ITAG(I),I=J,LD)
2     WRITE (IGFL) ICASE,NBLOKS,NPBLK,NLAST,NBLSYM,NPSYM,NLSYM,IMAT
      IF (IPERF.EQ.2) WRITE (IGFL) AR1,AR2,AR3,EPSCF,DXA,DYA,XSA,YSA,NXA
     1,NYA
      IF (NOP.GT.1) WRITE (IGFL) ((SSX(I,J),I=1,NOP),J=1,NOP)
      WRITE (IGFL) (IP(I),I=1,NEQ),COM
      IF (ICASE.GT.2) GO TO 3
      IOUT=NEQ*NPEQ
      WRITE (IGFL) (CM(I),I=1,IOUT)
      GO TO 12
3     IF (ICASE.NE.4) GO TO 5
      REWIND 13
      I=NPEQ*NPEQ
      DO 4 K=1,NOP
      READ (13) (CM(J),J=1,I)
4     WRITE (IGFL) (CM(J),J=1,I)
      REWIND 13
      GO TO 12
5     REWIND 13
      REWIND 14
      IF (ICASE.EQ.5) GO TO 8
      IOUT=NPBLK*NEQ*2
      DO 6 I=1,NBLOKS
      CALL BLCKIN (CM,13,1,IOUT,1,201)
6     CALL BLCKOT (CM,IGFL,1,IOUT,1,202)
      DO 7 I=1,NBLOKS
```

```
      CALL BLCKIN (CM,14,1,IOUT,1,203)
7     CALL BLCKOT (CM,IGFL,1,IOUT,1,204)
      GO TO 12
8     IOUT=NPSYM*NPEQ*2
      DO 11 IOP=1,NOP
      DO 9 I=1,NBLSYM
      CALL BLCKIN (CM,13,1,IOUT,1,205)
9     CALL BLCKOT (CM,IGFL,1,IOUT,1,206)
      DO 10 I=1,NBLSYM
      CALL BLCKIN (CM,14,1,IOUT,1,207)
10    CALL BLCKOT (CM,IGFL,1,IOUT,1,208)
11    CONTINUE
      REWIND 13
      REWIND 14
12    REWIND IGFL
      PRINT 13, IGFL,IMAT
      RETURN
C
13    FORMAT (///,44H ****NUMERICAL GREEN'S FUNCTION FILE ON TAPE,I3,5H
     1****,/,5X,16HMATRIX STORAGE -,I7,16H COMPLEX NUMBERS,///)
      END
```

GH

PURPOSE

To compute the function that is numerically integrated for the near H field of a segment.

METHOD

The value returned by GH is

$$G = \left[\frac{1}{(kr)^3} + \frac{j}{(kr)^2}\right] \exp(-jkr) \ ,$$

where

$r = \left[\rho'^2 + (z - z')^2\right]^{1/2}$

ρ' = ρ coordinate of the field observation point in a cylindrical coordinate system with origin at the center of the source segment and z axis oriented along the source segment

z' = z coordinate of the field observation point in the cylindrical coordinate system

z = z coordinate of the integration point on the source segment

$k = 2\pi/\lambda$

SYMBOL DICTIONARY

CKR = cos kr

HR = real part of G

HI = imaginary part of G

R = kr

RHKS = $(k\rho')^2$

RR2 = $1/(kr)^2$

RR3 = $1/(kr)^3$

RS = $(kr)^2$

SKR = sin kr

ZK = kz

ZPK = kz'

```
      SUBROUTINE GH (ZK,HR,HI)
C     INTEGRAND FOR H FIELD OF A WIRE
      COMMON /TMH/ ZPK,RHKS
      RS=ZK-ZPK
      RS=RHKS+RS*RS
      R=SQRT(RS)
      CKR=COS(R)
      SKR=SIN(R)
      RR2=1./RS
      RR3=RR2/R
      HR=SKR*RR2+CKR*RR3
      HI=CKR*RR2-SKR*RR3
      RETURN
      END
```

GWAVE

PURPOSE

To compute the components of electric field due to an electric current element over a ground plane at intermediate distances, including the surface wave field.

METHOD

Approximate expressions for the electric field of a vertical or horizontal infinitesimal current element above a ground plane, including surface wave, were derived by K. A. Norton (ref. 2). The geometry is shown in figure 6a for a current element at height a above the ground plane and field observation point at p. The current element is located on the z' axis, and the horizontal current element is directed along the x' axis. The vertical current element produces z' and ρ' field components given by

$$E_z^v = -\frac{j\eta I d\ell}{2\lambda}\left\{\cos^2\psi' \frac{\exp(-jkR_1)}{R_1} + R_v \cos^2\psi \frac{\exp(-jkR_2)}{R_2}\right.$$

$$+ (1 - R_v)\cos^2\psi \; F \frac{\exp(-jkR_2)}{R_2}$$

$$+ u\sqrt{1 - u^2\cos^2\psi}\,\sin\psi\; 2\,\frac{\exp(-jkR_2)}{jkR_2^2}$$

$$+ \frac{\exp(-jkR_1)}{R_1}\left(\frac{1}{jkR_1} + \frac{1}{(jkR_1)^2}\right)(1 - 3\sin^2\psi')$$

$$\left. + \frac{\exp(-jkR_2)}{R_2}\left(\frac{1}{jkR_2} + \frac{1}{(jkR_2)^2}\right)(1 - 3\sin^2\psi)\right\} ,$$

$$E_{\rho}^{v} = \frac{j\eta I d\ell}{2\lambda} \left\{ \sin\psi' \cos\psi' \frac{\exp(-jkR_1)}{R_1} + R_v \sin\psi \cos\psi \frac{\exp(-jkR_2)}{R_2} \right.$$

$$- \cos\psi (1 - R_v)\, u \sqrt{1 - u^2 \cos^2\psi}\, F \frac{\exp(-jkR_2)}{R_2}$$

$$- \sin\psi \cos\psi\, (1 - R_v) \frac{\exp(-jkR_2)}{jkR_2^2}$$

$$+ 3 \sin\psi' \cos\psi' \left(\frac{1}{jkR_1} + \frac{1}{(jkR_1)^2} \right) \frac{\exp(-jkR_1)}{R_1}$$

$$- \cos\psi\, u \sqrt{1 - u^2 \cos^2\psi}\, (1 - R_v) \frac{\exp(-jkR_2)}{2jkR_2^2}$$

$$\left. + 3 \sin\psi \cos\psi \left(\frac{1}{jkR_2} + \frac{1}{(jkR_2)^2} \right) \frac{\exp(-jkR_2)}{R_2} \right\} ,$$

where

$F = 1 - j \sqrt{\pi w} \exp(-w) \operatorname{erfc}(j\sqrt{w})$

$\operatorname{erfc}(z) = 1 - \operatorname{erf}(z)$

$\operatorname{erf}(z) = 2/\sqrt{\pi} \int_0^z \exp(-t^2)\, dt$ (error function)

$w = 4p_1/(1 - R_v)^2$

$p_1 = -jkR_2 u^2 (1 - u^2 \cos^2\psi)/(2\cos^2\psi)$

$R_v = \dfrac{\sin\psi - u\sqrt{1 - u^2\cos^2\psi}}{\sin\psi + u\sqrt{1 - u^2\cos^2\psi}}$

$u = k/k_2$

k = wave number in free space

k_2 = wave number in lower medium

$\sin\psi = (z + a)/R_2$

$\sin\psi' = (z - a)/R_1$

The horizontal current element directed along the x' axis produces ρ', φ', and z' field components given by

$$E_z^h = \frac{j\eta I d\ell}{2\lambda} \cos\phi' \Bigg\{ \sin\psi' \cos\psi' \frac{\exp(-jkR_1)}{R_1}$$

$$- R_v \sin\psi \cos\psi \cdot \frac{\exp(-jkR_2)}{R_2}$$

$$+ \cos\psi\,(1 - R_v)\, u\sqrt{1 - u^2 \cos^2\psi}\; F \frac{\exp(-jkR_2)}{R_2}$$

$$+ \sin\psi \cos\psi\,(1 - R_v) \frac{\exp(-jkR_2)}{jkR_2^2}$$

$$+ 3 \sin\psi' \cos\psi' \left(\frac{1}{jkR_1} + \frac{1}{(jkR_1)^2}\right) \frac{\exp(-jkR_1)}{R_1}$$

$$+ \cos\psi\,(1 - R_v)\, u\sqrt{1 - u^2 \cos^2\psi}\; \frac{\exp(-jkR_2)}{2jkR_2^2}$$

$$- 3 \sin\psi \cos\psi \left(\frac{1}{jkR_2} + \frac{1}{(jkR_2)^2}\right) \frac{\exp(-jkR_2)}{R_2} \Bigg\} ,$$

$$E_{\rho}^{h} = \frac{-j\eta I d\ell}{2\lambda} \cos\phi' \left\{ \sin^2\psi' \frac{\exp(-jkR_1)}{R_1} - R_v \sin^2\psi \frac{\exp(-jkR_2)}{R_2} \right.$$

$$- (1 - u^2\cos^2\psi)\, u^2 (1 - R_v)\, F \frac{\exp(-jkR_2)}{R_2}$$

$$+ \left(\frac{1}{jkR_1} + \frac{1}{(jkR_1)^2} \right) (1 - 3\cos^2\psi') \frac{\exp(-jkR_1)}{R_1}$$

$$- \left(\frac{1}{jkR_2} + \frac{1}{(jkR_2)^2} \right) (1 - 3\cos^2\psi) \left[1 - u^2 (1 + R_v) - u^2 (1 - R_v) F \right]$$

$$\times \frac{\exp(-jkR_2)}{R_2} + u^2 \cos^2\psi\, (1 - R_v) \left(1 + \frac{1}{jkR_2} \right)$$

$$\left. \times \left[F\left(u^2 (1 - u^2\cos^2\psi) - \sin^2\psi + \frac{1}{jkR_2} \right) - \frac{1}{jkR_2} \right] \frac{\exp(-jkR_2)}{R_2} \right\} ,$$

$$E_{\phi}^{h} = \frac{j\eta I d\ell}{2\lambda} \sin\phi' \left\{ \frac{\exp(-jkR_1)}{R_1} - R_h \frac{\exp(-jkR_2)}{R_2} \right.$$

$$+ (R_h + 1)\, G \frac{\exp(-jkR_2)}{R_2} + \left(1 + \frac{1}{jkR_1} \right) \frac{\exp(-jkR_1)}{jkR_1^2}$$

$$- \left(1 + \frac{1}{jkR_2} \right) [1 - u^2 (1 + R_v) - u^2 (1 - R_v) F] \frac{\exp(-jkR_2)}{jkR_2^2}$$

$$- \frac{u^2 (1 - R_v)}{2} \left[F\left(u^2 (1 - u^2\cos^2\psi) - \sin^2\psi + \frac{1}{jkR_2} \right) - \frac{1}{jkR_2} \right]$$

$$\left. \times \frac{\exp(-jkR_2)}{jkR_2^2} \right\} ,$$

where

$$G = [1 - j\sqrt{\pi v}\,\exp(-v)\,\mathrm{erfc}\,(j\sqrt{v})],$$

$$v = 4q_1/(1 + R_h)^2$$

$$q_1 = -jkR_2\,(1 - u^2\cos^2\psi)/(2u^2\cos^2\psi)$$

$$R_h = \frac{\sqrt{1 - u^2\cos^2\psi} - u\sin\psi}{\sqrt{1 - u^2\cos^2\psi} + u\sin\psi}$$

The approximations in these expressions are valid for R_1 and R_2 greater than about a wavelength and to second order in u^2. In each equation, the first term represents the direct space wave field of the current element, the second term is the space wave field reflected from the ground, and the following higher order terms involving F and G represent the ground wave. It may be noted that the coefficients R_v and R_h are the Fresnel reflection coefficients for vertical and horizontal polarization, respectively.

To obtain the field due to a structure, these expressions are integrated over each segment and the fields of the segments are summed in subroutine GFLD. For integration, R_1 and R_2 are the distances from the integration point ℓ on the segment to point p. Since R_1 and R_2 are assumed large compared to the segment length, R_1, R_2, ψ, and ψ' are considered constant during integration over the segment except where jkR_1 and jkR_2 occur in exponential functions. Thus, if s represents distance along the segment, the integral of each expression over the segment is obtained by replacing $(Id\ell/\lambda^2)\exp(-jkR_1)$ and $(Id\ell/\lambda^2)\exp(-jkR_2)$ by XX1 and XX2 from subroutine GFLD. A factor of exp(-jkR) is omitted from the fields and is included after summation in GFLD. Including a factor of $1/\lambda^2$ in XX1 and XX2 makes a factor of λ available to normalize R_1 and R_2 in the denominators of the field expressions. The factors sin ϕ' or cos ϕ' are omitted from the fields due to a horizontal current element in GWAVE and are supplied later.

SYMBOL DICTIONARY

CPP = $\cos \psi$

CPPP = $\cos \psi'$

CPPP2 = $\cos^2 \psi'$

CPP2 = $\cos^2 \psi$

ECON = $-j\eta/2$ (η = impedance of free space)

EPH = $E_{\phi}^{h}/\sin \phi'$

ERH = $E_{\rho}^{h}/\cos \phi'$

ERV = E_{ρ}^{v}

EZH = $E_{z}^{h}/\cos \phi'$

EZV = E_{z}^{v}

F = F

FJ = $j = \sqrt{-1}$

G = G

OMR = $1 - R_v$

PI = π

P1 = p_1

Q1 = q_1

RH = R_h

RK1 = $-jkR_1$

RK2 = $-jkR_2$

RV = R_v

R1 = R_1/λ

R2 = R_2/λ

SPP = $\sin \psi$

SPPP = $\sin \psi'$

SPPP2 = $\sin^2 \psi'$

SPP2 = $\sin^2 \psi$

TPJ = $2\pi j$

T1 = $1 - u^2 \cos^2 \psi$

T2 = $\sqrt{T1}$

T3 = $-[1/(jkR_1) + 1/(jkR_1)^2]$

T4	=	$-[1/(jkR_2) + 1/(jkR_2)^2]$
U	=	u
U2	=	u^2
V	=	v
W	=	w
XR1	=	$XX1/(R/\lambda)$
XR2	=	$XX2/(R/\lambda)$
XX1	=	$G_1 \exp(jk\hat{R}_1 \cdot \bar{r}_i)$
XX2	=	$G_2 \exp(jk\hat{R}_2 \cdot \bar{r}_i')$
X1, X2, X3, X4, X5, X6, X7	=	first, second, ..., seventh term in each field expression
ZMH	=	z - a
ZPH	=	z + a

CONSTANTS

(0., 1.)	=	$j = \sqrt{-1}$
(0., 6.2831853)	=	$2\pi j$
(0., -188.363)	=	$-j\eta/2$
3.1415926	=	π

```
      SUBROUTINE GWAVE (ERV,EZV,ERH,EZH,EPH)
C
C     GWAVE COMPUTES THE ELECTRIC FIELD, INCLUDING GROUND WAVE, OF A
C     CURRENT ELEMENT OVER A GROUND PLANE USING FORMULAS OF K.A. NORTON
C     (PROC. IRE, SEPT., 1937, PP.1203,1236.)
C
      COMPLEX FJ,TPJ,U2,U,RK1,RK2,T1,T2,T3,T4,P1,RV,OMR,W,F,Q1,RH,V,G,XR
     1,XR2,X1,X2,X3,X4,X5,X6,X7,EZV,ERV,EZH,ERH,EPH,XX1,XX2,ECON,FBAR
      COMMON /GWAV/ U,U2,XX1,XX2,R1,R2,ZMH,ZPH
      DIMENSION FJX(2), TPJX(2), ECONX(2)
      EQUIVALENCE (FJ,FJX), (TPJ,TPJX), (ECON,ECONX)
      DATA PI/3.141592654/,FJX/0.,1./,TPJX/0.,6.283185308/
      DATA ECONX/0.,-188.367/
      SPPP=ZMH/R1
      SPPP2=SPPP*SPPP
      CPPP2=1.-SPPP2
      IF (CPPP2.LT.1.E-20) CPPP2=1.E-20
      CPPP=SQRT(CPPP2)
      SPP=ZPH/R2
      SPP2=SPP*SPP
      CPP2=1.-SPP2
      IF (CPP2.LT.1.E-20) CPP2=1.E-20
      CPP=SQRT(CPP2)
      RK1=-TPJ*R1
      RK2=-TPJ*R2
      T1=1.-U2*CPP2
      T2=CSQRT(T1)
      T3=(1.-1./RK1)/RK1
      T4=(1.-1./RK2)/RK2
      P1=RK2*U2*T1/(2.*CPP2)
      RV=(SPP-U*T2)/(SPP+U*T2)
      OMR=1.-RV
      W=1./OMR
      W=(4.,0.)*P1*W*W
      F=FBAR(W)
      Q1=RK2*T1/(2.*U2*CPP2)
      RH=(T2-U*SPP)/(T2+U*SPP)
      V=1./(1.+RH)
      V=(4.,0.)*Q1*V*V
      G=FBAR(V)
      XR1=XX1/R1
      XR2=XX2/R2
      X1=CPPP2*XR1
      X2=RV*CPP2*XR2
      X3=OMR*CPP2*F*XR2
      X4=U*T2*SPP*2.*XR2/RK2
      X5=XR1*T3*(1.-3.*SPPP2)
      X6=XR2*T4*(1.-3.*SPP2)
      EZV=(X1+X2+X3-X4-X5-X6)*ECON
      X1=SPPP*CPPP*XR1
      X2=RV*SPP*CPP*XR2
      X3=CPP*OMR*U*T2*F*XR2
      X4=SPP*CPP*OMR*XR2/RK2
      X5=3.*SPPP*CPPP*T3*XR1
      X6=CPP*U*T2*OMR*XR2/RK2*.5
      X7=3.*SPP*CPP*T4*XR2
      ERV=-(X1+X2-X3+X4-X5+X6-X7)*ECON
      EZH=-(X1-X2+X3-X4-X5-X6+X7)*ECON
      X1=SPPP2*XR1
      X2=RV*SPP2*XR2
      X4=U2*T1*OMR*F*XR2
      X5=T3*(1.-3.*CPPP2)*XR1
      X6=T4*(1.-3.*CPP2)*(1.-U2*(1.+RV)-U2*OMR*F)*XR2
      X7=U2*CPP2*OMR*(1.-1./RK2)*(F*(U2*T1-SPP2-1./RK2)+1./RK2)*XR2
```

```
      ERH=(X1-X2-X4-X5+X6+X7)*ECON                                      GW  65
      X1=XR1                                                            GW  66
      X2=RH*XR2                                                         GW  67
      X3=(RH+1.)*G*XR2                                                  GW  68
      X4=T3*XR1                                                         GW  69
      X5=T4*(1.-U2*(1.+RV)-U2*OMR*F)*XR2                                GW  70
      X6=.5*U2*OMR*(F*(U2*T1-SPP2-1./RK2)+1./RK2)*XR2/RK2               GW  71
      EPH=-(X1-X2+X3-X4+X5+X6)*ECON                                     GW  72
      RETURN                                                            GW  73
      END                                                               GW  74-
```

GX

PURPOSE

To evaluate terms for the field contribution due to segment ends in the thin wire kernel.

SYMBOL DICTIONARY

GZ = $\exp(-jkr)/r = G_0$

GZP = $-(1 + jkr)\exp(-jkr)/r^3$

R = r

R2 = $r^2 = \rho^2 + z^2$

RH = ρ

RK = kR

XK = $2\pi/\lambda$

ZZ = z

CODE LISTING

```
      SUBROUTINE GX (ZZ,RH,XK,GZ,GZP)
C     SEGMENT END CONTRIBUTIONS FOR THIN WIRE APPROX.
      COMPLEX GZ,GZP
      R2=ZZ*ZZ+RH*RH
      R=SQRT(R2)
      RK=XK*R
      GZ=CMPLX(COS(RK),-SIN(RK))/R
      GZP=-CMPLX(1.,RK)*GZ/R2
      RETURN
      END
```

GXX

PURPOSE

To evaluate terms for the field contribution due to segment ends in the extended thin wire kernel.

METHOD

Equations 89 through 94 in Part I are evaluated for $\rho > a$, and equations 99 through 103 for $\rho < a$. Several variables are used for storage of intermediate results before being set to their final values.

SYMBOL DICTIONARY

A = radius of source segment, a

A2 = a^2

C1 = $1 + jkr_0$

C2 = $3(1 + jkr_0) - k^2r_0^2$

C3 = $(6 + jkr_0)k^2r_0^2 - 15(1 + jkr_0)$

G1 = G_1

G1P = $\partial G_1/\partial z'$

G2 = G_2

G2P = $\partial G_2/\partial z'$

G3 = $\partial G_1/\partial \rho$

GZ = G_0

GZP = $\partial G_0/\partial z'$

IRA = 1 to indicate $\rho < a$

R = r_0

R2 = r_0^2

R4 = r_0^4

RH = ρ

RH2 = ρ^2

RK = kr_0

RK2 = $k^2r_0^2$

T1 = $a^2\rho^2/4r^4$

T2 = $a^2/2r^2$

XK = $k = 2\pi/\lambda$

ZZ = $z' - z$

```
      SUBROUTINE GXX (ZZ,RH,A,A2,XK,IRA,G1,G1P,G2,G2P,G3,GZP)
C     SEGMENT END CONTRIBUTIONS FOR EXT. THIN WIRE APPROX.
      COMPLEX GZ,C1,C2,C3,G1,G1P,G2,G2P,G3,GZP
      R2=ZZ*ZZ+RH*RH
      R=SQRT(R2)
      R4=R2*R2
      RK=XK*R
      RK2=RK*RK
      RH2=RH*RH
      T1=.25*A2*RH2/R4
      T2=.5*A2/R2
      C1=CMPLX(1.,RK)
      C2=3.*C1-RK2
      C3=CMPLX(6.,RK)*RK2-15.*C1
      GZ=CMPLX(COS(RK),-SIN(RK))/R
      G2=GZ*(1.+T1*C2)
      G1=G2-T2*C1*GZ
      GZ=GZ/R2
      G2P=GZ*(T1*C3-C1)
      GZP=T2*C2*GZ
      G3=G2P+GZP
      G1P=G3*ZZ
      IF (IRA.EQ.1) GO TO 2
      G3=(G3+GZP)*RH
      GZP=-ZZ*C1*GZ
      IF (RH.GT.1.E-10) GO TO 1
      G2=0.
      G2P=0.
      RETURN
1     G2=G2/RH
      G2P=G2P*ZZ/RH
      RETURN
2     T2=.5*A
      G2=-T2*C1*GZ
      G2P=T2*GZ*C2/R2
      G3=RH2*G2P-A*GZ*C1
      G2P=G2P*ZZ
      GZP=-ZZ*C1*GZ
      RETURN
      END
```

HFK

PURPOSE

To compute the near H field of a uniform current filament by numerical integration.

METHOD

The H field of a current filament of length Δ with uniform current distribution of magnitude I = λ is

$$H_\phi = \frac{k\rho'}{2} \int_{-k\Delta/2}^{k\Delta/2} \left[\frac{1}{(kr)^3} + \frac{j}{(kr)^2} \right] \exp(-jkr)\, d(kz) \ ,$$

where r, ρ', z' and z are defined in the description of subroutine GH. The numerical integration is performed by the method of Romberg quadrature with variable interval width, which is described in the discussion of subroutine INTX. The integral is multiplied by kρ'/2 at HF79 and HF80 in the code.

SYMBOL DICTIONARY

This listing excludes those variables used in the numerical quadrature algorithm, which are defined under subroutine INTX.

RHK = kρ'

RHKS = $(k\rho')^2$

SGI = imaginary part of H_ϕ

SGR = real part of H_ϕ .

ZPK = kz' (z' = z coordinate of observation point)

ZPKX = ZPK

```
      SUBROUTINE HFK (EL1,EL2,RHK,ZPKX,SGR,SGI)
C     HFK COMPUTES THE H FIELD OF A UNIFORM CURRENT FILAMENT BY
C     NUMERICAL INTEGRATION
      COMMON /TMH/ ZPK,RHKS
      DATA NX,NM,NTS,RX/1,65536,4,1.E-4/
      ZPK=ZPKX
      RHKS=RHK*RHK
      Z=EL1
      ZE=EL2
      S=ZE-Z
      EP=S/(10.*NM)
      ZEND=ZE-EP
      SGR=0.0
      SGI=0.0
      NS=NX
      NT=0
      CALL GH (Z,G1R,G1I)
1     DZ=S/NS
      ZP=Z+DZ
      IF (ZP-ZE) 3,3,2
2     DZ=ZE-Z
      IF (ABS(DZ)-EP) 17,17,3
3     DZOT=DZ*.5
      ZP=Z+DZOT
      CALL GH (ZP,G3R,G3I)
      ZP=Z+DZ
      CALL GH (ZP,G5R,G5I)
4     T00R=(G1R+G5R)*DZOT
      T00I=(G1I+G5I)*DZOT
      T01R=(T00R+DZ*G3R)*0.5
      T01I=(T00I+DZ*G3I)*0.5
      T10R=(4.0*T01R-T00R)/3.0
      T10I=(4.0*T01I-T00I)/3.0
      CALL TEST (T01R,T10R,TE1R,T01I,T10I,TE1I,0.)
      IF (TE1I-RX) 5,5,6
5     IF (TE1R-RX) 8,8,6
6     ZP=Z+DZ*0.25
      CALL GH (ZP,G2R,G2I)
      ZP=Z+DZ*0.75
      CALL GH (ZP,G4R,G4I)
      T02R=(T01R+DZOT*(G2R+G4R))*0.5
      T02I=(T01I+DZOT*(G2I+G4I))*0.5
      T11R=(4.0*T02R-T01R)/3.0
      T11I=(4.0*T02I-T01I)/3.0
      T20R=(16.0*T11R-T10R)/15.0
      T20I=(16.0*T11I-T10I)/15.0
      CALL TEST (T11R,T20R,TE2R,T11I,T20I,TE2I,0.)
      IF (TE2I-RX) 7,7,14
7     IF (TE2R-RX) 9,9,14
8     SGR=SGR+T10R
      SGI=SGI+T10I
      NT=NT+2
      GO TO 10
9     SGR=SGR+T20R
      SGI=SGI+T20I
      NT=NT+1
10    Z=Z+DZ
      IF (Z-ZEND) 11,17,17
11    G1R=G5R
      G1I=G5I
      IF (NT-NTS) 1,12,12
12    IF (NS-NX) 1,1,13
13    NS=NS/2
      NT=1
```

```
      GO TO 1
   14 NT=0
      IF (NS-NM) 16,15,15
   15 PRINT 18, Z
      GO TO 9
   16 NS=NS*2
      DZ=S/NS
      DZOT=DZ*0.5
      G5R=G3R
      G5I=G3I
      G3R=G2R
      G3I=G2I
      GO TO 4
   17 CONTINUE
      SGR=SGR*RHK*.5
      SGI=SGI*RHK*.5
      RETURN
C
   18 FORMAT (24H STEP SIZE LIMITED AT Z=,F10.5)
      END
```

HINTG

PURPOSE

To compute the near magnetic field due to a single patch in free space or over ground.

METHOD

The magnetic field is computed at the point, XI, YI, ZI due to the patch defined by parameters in COMMON/DATAJ/. The H field at $\bar{r} = (XI)\hat{x} + (YI)\hat{y} + (ZI)\hat{z}$ due to patch i, centered at $\bar{r}_i$, is approximated as:

$$\bar{H}(\bar{r}) = -\frac{1}{4\pi}\left[(1 + jkR)\,\frac{\exp(-jkR)}{(R/\lambda)^3}\right]\left[(\bar{R}/\lambda) \times \bar{J}_i\right] A_i/\lambda^2$$

where $\bar{R} = \bar{r} - \bar{r}_i$, and A_i is the area of patch i. This expression treats the surface currents as lumped at the center of the patch. $\bar{H}$ is computed for unit currents along the surface vectors $\hat{t}_{1i}$ and $\hat{t}_{2i}$.

When a ground is present, the code is executed twice in a loop. In the second pass, the field of the image of the patch is computed, multiplied by the reflection coefficients, and added to the direct field.

SYMBOL DICTIONARY

CR = cos (kR)

CTH = cos θ, θ = angle between the reflected ray and the normal to the ground

EXC, EYC, EZC = x, y, and z components of $\bar{H}$ excluding $(\times \bar{J}_i)$ term

EXK, EYK, EZK = $\bar{H}$ for $\bar{J}_i = \hat{t}_{1i}$

EXS, EYS, EZS = $\bar{H}$ for $\bar{J}_i = \hat{t}_{2i}$

F1X, F1Y, F1Z = $\bar{H}$ for $\bar{J}_i = \hat{t}_{1i}$; direct or reflected field contribution

F2X, F2Y, F2Z = $\overline{H}$ for $\overline{J}_i = \hat{t}_{2i}$; direct or reflected field contribution

FPI = 4π

GAM = $\overline{H}$ excluding the term $(\overline{R}/\lambda) \times \overline{J}_i$

IP = 1 for direct field, 2 for reflected field

IPERF = 1 for perfect ground, 0 otherwise

KSYMP = 1 for free space, 2 for ground

PX, PY = unit vector normal to plane of incidence for reflected ray $\hat{\rho}$

R = R/λ

RFL = +1 for direct field, -1 for reflected field

RK = kR; $k = 2\pi/\lambda$

RRH = R_H

RRV = R_V

RSQ = R^2/λ^2

RX, RY, RZ = $\overline{R}/\lambda$

S = A_i/λ^2

SR = sin (kR)

T1XJ, T1YJ, T1ZJ = $\hat{t}_{1i}$

T2XJ, T2YJ, T2ZJ = $\hat{t}_{2i}$

T1ZR = z component of $\hat{t}_{1i}$ for patch i or for the image of patch i reflected in the ground

T2ZR = same as T1ZR for the $\hat{t}_{2i}$

TP = 2π

XI, YI, ZI = field evaluation point $\overline{r}/\lambda$

XJ, YJ, ZJ = position of center of patch $\overline{r}_i/\lambda$

XYMAG = magnitude of $\overline{R}/\lambda$ projected on the x-y plane

CONSTANTS

12.56637062 = 4π

6.283185308 = 2π

```
      SUBROUTINE HINTG (XI,YI,ZI)                                       HI   1
C     HINTG COMPUTES THE H FIELD OF A PATCH CURRENT                     HI   2
      COMPLEX EXK,EYK,EZK,EXS,EYS,EZS,EXC,EYC,EZC,ZRATI,ZRATI2,GAM,F1X,F HI   3
     11Y,F1Z,F2X,F2Y,F2Z,RRV,RRH,T1,FRATI                               HI   4
      COMMON /DATAJ/ S,B,XJ,YJ,ZJ,CABJ,SABJ,SALPJ,EXK,EYK,EZK,EXS,EYS,EZ HI   5
     1S,EXC,EYC,EZC,RKH,IEXK,IND1,IND2,IPGND                            HI   6
      COMMON /GND/ZRATI,ZRATI2,FRATI,CL,CH,SCRWL,SCRWR,NRADL,KSYMP,IFAR, HI   7
     1IPERF,T1,T2                                                       HI   8
      EQUIVALENCE (T1XJ,CABJ), (T1YJ,SABJ), (T1ZJ,SALPJ), (T2XJ,B), (T2Y HI   9
     1J,IND1), (T2ZJ,IND2)                                              HI  10
      DATA FPI/12.56637062/,TP/6.283185308/                             HI  11
      RX=XI-XJ                                                          HI  12
      RY=YI-YJ                                                          HI  13
      RFL=-1.                                                           HI  14
      EXK=(0.,0.)                                                       HI  15
      EYK=(0.,0.)                                                       HI  16
      EZK=(0.,0.)                                                       HI  17
      EXS=(0.,0.)                                                       HI  18
      EYS=(0.,0.)                                                       HI  19
      EZS=(0.,0.)                                                       HI  20
      DO 5 IP=1,KSYMP                                                   HI  21
      RFL=-RFL                                                          HI  22
      RZ=ZI-ZJ*RFL                                                      HI  23
      RSQ=RX*RX+RY*RY+RZ*RZ                                             HI  24
      IF (RSQ.LT.1.E-20) GO TO 5                                        HI  25
      R=SQRT(RSQ)                                                       HI  26
      RK=TP*R                                                           HI  27
      CR=COS(RK)                                                        HI  28
      SR=SIN(RK)                                                        HI  29
      GAM=-(CMPLX(CR,-SR)+RK*CMPLX(SR,CR))/(FPI*RSQ*R)*S                HI  30
      EXC=GAM*RX                                                        HI  31
      EYC=GAM*RY                                                        HI  32
      EZC=GAM*RZ                                                        HI  33
      T1ZR=T1ZJ*RFL                                                     HI  34
      T2ZR=T2ZJ*RFL                                                     HI  35
      F1X=EYC*T1ZR-EZC*T1YJ                                             HI  36
      F1Y=EZC*T1XJ-EXC*T1ZR                                             HI  37
      F1Z=EXC*T1YJ-EYC*T1XJ                                             HI  38
      F2X=EYC*T2ZR-EZC*T2YJ                                             HI  39
      F2Y=EZC*T2XJ-EXC*T2ZR                                             HI  40
      F2Z=EXC*T2YJ-EYC*T2XJ                                             HI  41
      IF (IP.EQ.1) GO TO 4                                              HI  42
      IF (IPERF.NE.1) GO TO 1                                           HI  43
      F1X=-F1X                                                          HI  44
      F1Y=-F1Y                                                          HI  45
      F1Z=-F1Z                                                          HI  46
      F2X=-F2X                                                          HI  47
      F2Y=-F2Y                                                          HI  48
      F2Z=-F2Z                                                          HI  49
      GO TO 4                                                           HI  50
1     XYMAG=SQRT(RX*RX+RY*RY)                                           HI  51
      IF (XYMAG.GT.1.E-6) GO TO 2                                       HI  52
      PX=0.                                                             HI  53
      PY=0.                                                             HI  54
      CTH=1.                                                            HI  55
      RRV=(1.,0.)                                                       HI  56
      GO TO 3                                                           HI  57
2     PX=-RY/XYMAG                                                      HI  58
      PY=RX/XYMAG                                                       HI  59
      CTH=RZ/R                                                          HI  60
      RRV=CSQRT(1.-ZRATI*ZRATI*(1.-CTH*CTH))                            HI  61
3     RRH=ZRATI*CTH                                                     HI  62
      RRH=(RRH-RRV)/(RRH+RRV)                                           HI  63
      RRV=ZRATI*RRV                                                     HI  64
```

```
      RRV=-(CTH-RRV)/(CTH+RRV)
      GAM=(F1X*PX+F1Y*PY)*(RRV-RRH)
      F1X=F1X*RRH+GAM*PX
      F1Y=F1Y*RRH+GAM*PY
      F1Z=F1Z*RRH
      GAM=(F2X*PX+F2Y*PY)*(RRV-RRH)
      F2X=F2X*RRH+GAM*PX
      F2Y=F2Y*RRH+GAM*PY
      F2Z=F2Z*RRH
    4 EXK=EXK+F1X
      EYK=EYK+F1Y
      EZK=EZK+F1Z
      EXS=EXS+F2X
      EYS=EYS+F2Y
      EZS=EZS+F2Z
    5 CONTINUE
      RETURN
      END
```

HSFLD

PURPOSE

To compute the near magnetic field due to constant, sine, and cosine current distributions on a segment in free space or over ground.

METHOD

The magnetic field is computed at the point XI, YI, ZI due to the segment defined by parameters in COMMON/DATAJ/. The fields computed by routine HSFLX are stored in /DATAJ/. When a ground is present, the code is executed twice in a loop. In the second pass, the field of the image of the segment is computed, multiplied by the reflection coefficients, and added to the direct field.

The field is evaluated in a cylindrical coordinate system with the source segment at the origin. The radius of a segment on which the field is evaluated is treated in the same way as for the electric field in subroutine EFLD. When the field evaluation point is not on a segment, the observation segment radius is set to zero in the call to HSFLD. Thus, as for the electric field, the ρ coordinate of the field evaluation point is computed for the surface of the observation segment as $\rho' = (\rho^2 + a^2)^{1/2}$, where ρ is the distance from the axis of the source segment to (XI, YI, ZI) and a is the radius of the observation segment. The resulting H field is multiplied by ρ/ρ'.

SYMBOL DICTIONARY

AI = radius of observation segment, if any

CTH = cos θ, θ = angle between the ray reflected from the ground and vertical

ETA = $\eta = \sqrt{\mu/\varepsilon}$

HPC, HPK, HPS = H_ϕ due to cosine, constant, and sine current, respectively

PHX, PHY, PHZ = $(\rho/\rho')\hat{\phi}$ in the cylindrical coordinates of the source segment or its image

PX, PY = unit vector normal to the plane of incidence of the reflected ray, $\hat{p}$

QX, QY, QZ = ρ/ρ' $[R_H\hat{\phi} + (R_V - R_H)(\hat{\phi} \cdot \hat{p})\hat{p}]$ for reflected ray

RFL = +1 for direct field, -1 for reflected field

RH = ρ'

RHOSPC = distance from coordinate origin to the point where the ray from the source to (XI, YI, ZI) reflects from the ground

RHOX, RHOY, RHOZ = $\bar{\rho}$ or $\bar{\rho}/\rho'$

RMAG = distance from the field evaluation point to the center of the source segment

RRH = R_H

RRV = R_V

SALPR = z component of unit vector in the direction of the source segment or its image

XI, YI, ZI = x, y, z coordinates of the field evaluation point

XIJ, YIJ, ZIJ = x, y, z components of distance from center of source segment to field observation point

XSPEC, YSPEC = x, y coordinates of the ground plane reflection point

XYMAG = horizontal distance from the source segment to the field observation point

ZP = projection of the vector (XIJ, YIJ, ZIJ) on the axis of the source segment

ZRATX = temporary storage for ZRATI

```
      SUBROUTINE HSFLD (XI,YI,ZI,AI)
C     HSFLD COMPUTES THE H FIELD FOR CONSTANT, SINE, AND COSINE CURRENT
C     ON A SEGMENT INCLUDING GROUND EFFECTS.
      COMPLEX EXK,EYK,EZK,EXS,EYS,EZS,EXC,EYC,EZC,ZRATI,ZRATI2,T1,HPK,HP
     1S,HPC,QX,QY,QZ,RRV,RRH,ZRATX,FRATI
      COMMON /DATAJ/ S,B,XJ,YJ,ZJ,CABJ,SABJ,SALPJ,EXK,EYK,EZK,EXS,EYS,EZ
     1S,EXC,EYC,EZC,RKH,IEXK,IND1,IND2,IPGND
      COMMON /GND/ZRATI,ZRATI2,FRATI,CL,CH,SCRWL,SCRWR,NRADL,KSYMP,IFAR,
     1IPERF,T1,T2
      DATA ETA/376.73/
      XIJ=XI-XJ
      YIJ=YI-YJ
      RFL=-1.
      DO 7 IP=1,KSYMP
      RFL=-RFL
      SALPR=SALPJ*RFL
      ZIJ=ZI-RFL*ZJ
      ZP=XIJ*CABJ+YIJ*SABJ+ZIJ*SALPR
      RHOX=XIJ-CABJ*ZP
      RHOY=YIJ-SABJ*ZP
      RHOZ=ZIJ-SALPR*ZP
      RH=SQRT(RHOX*RHOX+RHOY*RHOY+RHOZ*RHOZ+AI*AI)
      IF (RH.GT.1.E-10) GO TO 1
      EXK=0.
      EYK=0.
      EZK=0.
      EXS=0.
      EYS=0.
      EZS=0.
      EXC=0.
      EYC=0.
      EZC=0.
      GO TO 7
1     RHOX=RHOX/RH
      RHOY=RHOY/RH
      RHOZ=RHOZ/RH
      PHX=SABJ*RHOZ-SALPR*RHOY
      PHY=SALPR*RHOX-CABJ*RHOZ
      PHZ=CABJ*RHOY-SABJ*RHOX
      CALL HSFLX (S,RH,ZP,HPK,HPS,HPC)
      IF (IP.NE.2) GO TO 6
      IF (IPERF.EQ.1) GO TO 5
      ZRATX=ZRATI
      RMAG=SQRT(ZP*ZP+RH*RH)
      XYMAG=SQRT(XIJ*XIJ+YIJ*YIJ)
C
C     SET PARAMETERS FOR RADIAL WIRE GROUND SCREEN.
C
      IF (NRADL.EQ.0) GO TO 2
      XSPEC=(XI*ZJ+ZI*XJ)/(ZI+ZJ)
      YSPEC=(YI*ZJ+ZI*YJ)/(ZI+ZJ)
      RHOSPC=SQRT(XSPEC*XSPEC+YSPEC*YSPEC+T2*T2)
      IF (RHOSPC.GT.SCRWL) GO TO 2
      RRV=T1*RHOSPC*ALOG(RHOSPC/T2)
      ZRATX=(RRV*ZRATI)/(ETA*ZRATI+RRV)
2     IF (XYMAG.GT.1.E-6) GO TO 3
C
C     CALCULATION OF REFLECTION COEFFICIENTS WHEN GROUND IS SPECIFIED.
C
      PX=0.
      PY=0.
      CTH=1.
      RRV=(1.,0.)
      GO TO 4
```

```
    3     PX=-YIJ/XYMAG
          PY=XIJ/XYMAG
          CTH=ZIJ/RMAG
          RRV=CSQRT(1.-ZRATX*ZRATX*(1.-CTH*CTH))
    4     RRH=ZRATX*CTH
          RRH=-(RRH-RRV)/(RRH+RRV)
          RRV=ZRATX*RRV
          RRV=(CTH-RRV)/(CTH+RRV)
          QY=(PHX*PX+PHY*PY)*(RRV-RRH)
          QX=QY*PX+PHX*RRH
          QY=QY*PY+PHY*RRH
          QZ=PHZ*RRH
          EXK=EXK-HPK*QX
          EYK=EYK-HPK*QY
          EZK=EZK-HPK*QZ
          EXS=EXS-HPS*QX
          EYS=EYS-HPS*QY
          EZS=EZS-HPS*QZ
          EXC=EXC-HPC*QX
          EYC=EYC-HPC*QY
          EZC=EZC-HPC*QZ
          GO TO 7
    5     EXK=EXK-HPK*PHX
          EYK=EYK-HPK*PHY
          EZK=EZK-HPK*PHZ
          EXS=EXS-HPS*PHX
          EYS=EYS-HPS*PHY
          EZS=EZS-HPS*PHZ
          EXC=EXC-HPC*PHX
          EYC=EYC-HPC*PHY
          EZC=EZC-HPC*PHZ
          GO TO 7
    6     EXK=HPK*PHX
          EYK=HPK*PHY
          EZK=HPK*PHZ
          EXS=HPS*PHX
          EYS=HPS*PHY
          EZS=HPS*PHZ
          EXC=HPC*PHX
          EYC=HPC*PHY
          EZC=HPC*PHZ
    7     CONTINUE
          RETURN
          END
```

HSFLX

PURPOSE

To compute the near H field of filamentary currents of sine, cosine, and constant distribution on a segment.

METHOD

The wire segment is considered to be located at the origin of a local cylindrical coordinate system with the point at which the H field is computed being (ρ, ϕ, z). The coordinate geometry for a filament of current of length Δ is shown in figure 7. For a sine or cosine current distribution, the field can be written in closed form. For a current

$$I_0 \begin{bmatrix} \sin kz' \\ \cos kz' \end{bmatrix},$$

the field is

$$H_\phi\ (\rho,\ z) = \frac{-jI_0/\lambda}{2k\rho} \Bigg\{ \exp(-jkr_2) \begin{bmatrix} \cos(k\Delta/2) \\ -\sin(k\Delta/2) \end{bmatrix} - \exp(-jkr_1) \begin{bmatrix} \cos(k\Delta/2) \\ \sin(k\Delta/2) \end{bmatrix}$$

$$- j(kz - k\Delta/2)\ \frac{\exp(-jkr_2)}{kr_2} \begin{bmatrix} \sin\ (k\Delta/2) \\ \cos\ (k\Delta/2) \end{bmatrix}$$

$$+ j(kz + k\Delta/2)\ \frac{\exp(-jkr_1)}{kr_1} \begin{bmatrix} -\sin\ (k\Delta/2) \\ \cos\ (k\Delta/2) \end{bmatrix} \Bigg\}\ .$$

$I_0/\lambda = 1$ is assumed in this routine.

For small values of ρ with $|z| > \Delta/2$, this equation may produce large numerical errors due to cancellation of large terms. Hence, for $z > 0$ and $\rho/(z - \Delta/2) < 10^{-3}$, a more stable approximation for small $\rho/(z \pm \Delta/2)$ is used:

$$H_\phi(\rho,z) = \frac{(\rho/\lambda)(Io/\lambda)}{8\pi} \exp(-jkz) \Bigg\{ \left[\frac{2\pi}{(z + \Delta/2)/\lambda} - \frac{2\pi}{(z - \Delta/2)/\lambda} \right] \begin{bmatrix} 1 \\ -j \end{bmatrix}$$

$$+ \left[\frac{\exp(jk\Delta/2)}{(z-\Delta/2)^2/\lambda^2} \begin{pmatrix} \sin\ (k\Delta/2) \\ \cos\ (k\Delta/2) \end{pmatrix} - \frac{\exp(-jk\Delta/2)}{(z+\Delta/2)^2/\lambda^2} \begin{pmatrix} -\sin\ (k\Delta/2) \\ \cos\ (k\Delta/2) \end{pmatrix} \right] \Bigg\}$$

For $z < 0$, the above equation is evaluated for $H_\phi(\rho,\ -z)$. The field of a sin kz' current is multiplied by -1 in this case, since it is an odd function of z.

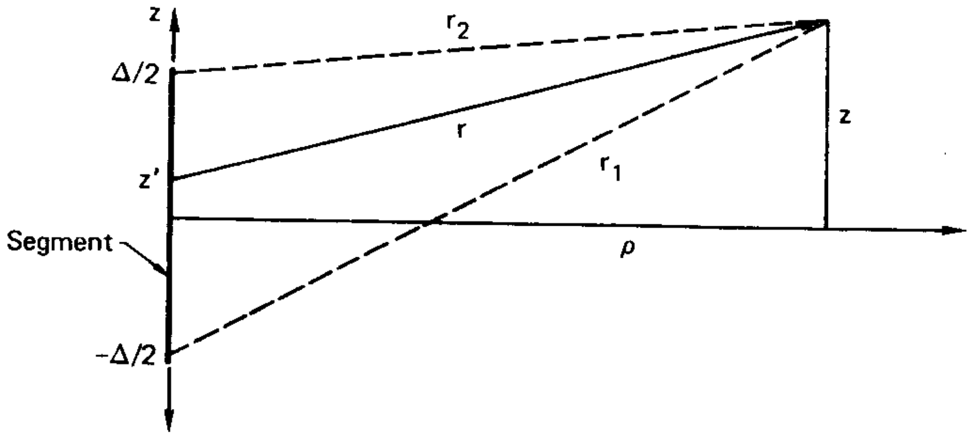


Figure 7. Coordinates for Evaluating H Field of a Segment.

The field due to a constant current is obtained by numerical integration, which is performed by subroutine HFK. If ρ is zero, all field quantities are set to zero, since H_ϕ is undefined.

SYMBOL DICTIONARY

CDK	= cos(kΔ/2)
CONS	= -j/(2kρ)
DH	= Δ/2
DK	= kΔ/2
EKR1	= $\exp(-jkr_1)$
EKR2	= $\exp(-jkr_2)$
FJ	= j
FJK	= -j2π
HKR, HKI	= real and imaginary parts of H_ϕ due to a constant current
HPC, HPK, HPS	= H_ϕ due to cosine, constant, and sine currents, respectively
HSS	= sign of z
PI8	= 8π
R1	= r_1
R2	= r_2
RH	= ρ
RH2	= ρ^2
RHZ	= ρ/(z - Δ/2)

S	=	Δ
SDK	=	$\sin(k\Delta/2)$
TP	=	2π
Z1	=	$z + \Delta/2$
Z2	=	$z - \Delta/2$
ZP	=	z

```
      SUBROUTINE HSFLX (S,RH,ZPX,HPK,HPS,HPC)                           HX   1
C     CALCULATES H FIELD OF SINE COSINE, AND CONSTANT CURRENT OF SEGMENT HX   2
      COMPLEX FJ,FJK,EKR1,EKR2,T1,T2,CONS,HPS,HPC,HPK                   HX   3
      DIMENSION FJX(2), FJKX(2)                                         HX   4
      EQUIVALENCE (FJ,FJX), (FJK,FJKX)                                  HX   5
      DATA TP/6.283185308/,FJX/0.,1./,FJKX/0.,-6.283185308/             HX   6
      DATA PI8/25.13274123/                                             HX   7
      IF (RH.LT.1.E-10) GO TO 6                                         HX   8
      IF (ZPX.LT.0.) GO TO 1                                            HX   9
      ZP=ZPX                                                            HX  10
      HSS=1.                                                            HX  11
      GO TO 2                                                           HX  12
1     ZP=-ZPX                                                           HX  13
      HSS=-1.                                                           HX  14
2     DH=.5*S                                                           HX  15
      Z1=ZP+DH                                                          HX  16
      Z2=ZP-DH                                                          HX  17
      IF (Z2.LT.1.E-7) GO TO 3                                          HX  18
      RHZ=RH/Z2                                                         HX  19
      GO TO 4                                                           HX  20
3     RHZ=1.                                                            HX  21
4     DK=TP*DH                                                          HX  22
      CDK=COS(DK)                                                       HX  23
      SDK=SIN(DK)                                                       HX  24
      CALL HFK (-DK,DK,RH*TP,ZP*TP,HKR,HKI)                             HX  25
      HPK=CMPLX(HKR,HKI)                                                HX  26
      IF (RHZ.LT.1.E-3) GO TO 5                                         HX  27
      RH2=RH*RH                                                         HX  28
      R1=SQRT(RH2+Z1*Z1)                                                HX  29
      R2=SQRT(RH2+Z2*Z2)                                                HX  30
      EKR1=CEXP(FJK*R1)                                                 HX  31
      EKR2=CEXP(FJK*R2)                                                 HX  32
      T1=Z1*EKR1/R1                                                     HX  33
      T2=Z2*EKR2/R2                                                     HX  34
      HPS=(CDK*(EKR2-EKR1)-FJ*SDK*(T2+T1))*HSS                          HX  35
      HPC=-SDK*(EKR2+EKR1)-FJ*CDK*(T2-T1)                               HX  36
      CONS=-FJ/(2.*TP*RH)                                               HX  37
      HPS=CONS*HPS                                                      HX  38
      HPC=CONS*HPC                                                      HX  39
      RETURN                                                            HX  40
5     EKR1=CMPLX(CDK,SDK)/(Z2*Z2)                                       HX  41
      EKR2=CMPLX(CDK,-SDK)/(Z1*Z1)                                      HX  42
      T1=TP*(1./Z1-1./Z2)                                               HX  43
      T2=CEXP(FJK*ZP)*RH/PI8                                            HX  44
      HPS=T2*(T1+(EKR1+EKR2)*SDK)*HSS                                   HX  45
      HPC=T2*(-FJ*T1+(EKR1-EKR2)*CDK)                                   HX  46
      RETURN                                                            HX  47
6     HPS=(0.,0.)                                                       HX  48
      HPC=(0.,0.)                                                       HX  49
      HPK=(0.,0.)                                                       HX  50
      RETURN                                                            HX  51
      END                                                               HX  52-
```

INTRP

PURPOSE

To evaluate the Sommerfeld integral contributions to the field of a source over ground by interpolation in precomputed tables.

METHOD

The interpolation region in R_1 and θ is covered by three grids as shown in Figure 12 of Part I. The interpolation tables and the number of data points and the boundaries of each grid are read from file 21 and stored in COMMON/GGRID/ by the main program. In subroutine INTRP the variable x corresponds to R_1 and y to θ.

The three interpolation tables are stored in the arrays AR1, AR2 and AR3 in COMMON/GGRID/. For grid i, ARi(I,J,K) is the value at

$$x_I = s_i + (I - 1)\,\Delta x_i\,, \qquad I = 1, \ldots N_i$$

$$y_J = t_i + (J - 1)\,\Delta y_i\,, \qquad J = 1, \ldots M_i$$

where s_i = XSA(i) , Δx_i = DXA(i) , N_i = NXA(i)

t_i = YSA(i) , Δy_i = DYA(i) , M_i = NYA(i)

Each array contains values for I^V_ρ, I^V_z, I^H_ρ and I^H_ϕ from equations 156 through 159 of Part I for K equal to 1 through 4, respectively. The grid boundaries and density of points can be varied but the relative positions of the three grids must be as shown in Figure 12 of Part I for the logic for choosing the correct grid to work correctly. In particular, XSA(1), YSA(1) and YSA(2) must be zero; and XSA(2) and XSA(3) must be equal.

For a given x and y the values of I^V_ρ, I^V_z, I^H_ρ and I^H_ϕ are found by bivariate cubic interpolation and returned in the variables F1, F2, F3 and F4. The grid containing (x,y) is determined and a four by four point region containing (x,y) is selected. If x_i and y_k are the minimum values of x and y in the four by four point region then four interpolation polynomials in x are computed for $y = y_j$ with $j = k, k + 1, k + 2, k + 3$. These are

$$f_{ij}(x) = a_{ij}\xi_i^3 + b_{ij}\xi_i^2 + c_{ij}\xi_i + d_{ij}$$

where $\xi_i = (x - x_{i+1})/\Delta x$

$$a_{ij} = \frac{1}{6}\,[F_{i+3,j} - F_{i,j} + 3(F_{i+1,j} - F_{i+2,j})]$$

$$b_{ij} = \frac{1}{2}\,[F_{i,j} - 2F_{i+1,j} + F_{i+2,j}]$$

$$c_{ij} = F_{i+2,j} - \frac{1}{6}\,[2F_{i,j} + 3F_{i+1,j} + F_{i+3,j}]$$

$$d_{ij} = F_{i+1,j}$$

$$F_{i,j} = F(x_i,\ y_j)$$

A cubic polynomial in y, fit to the points $f_{ij}(x)$ for $j = k, \ldots k + 3$ is then evaluated for the given y to obtain the interpolated value $\hat{F}(x,y)$

$$\hat{F}(x,y) = \frac{1}{6}\,(p_1\eta_k^3 + p_2\eta_k^2 + p_3\eta_k) + p_4$$

$$\eta_k = (y - y_{k+1})/\Delta y$$

$$p_1 = f_{i,k+3}(x) - f_{ik}(x) + 3\,[f_{i,k+1}(x) - f_{i,k+2}(x)]$$

$$p_2 = 3[f_{i,k}(x) - 2f_{i,k+1}(x) + f_{i,k+2}(x)]$$

$$p_3 = 6f_{i,k+2}(x) - 2f_{i,k}(x) - 3f_{i,k+1}(x) - f_{i,k+3}(x)$$

$$p_4 = f_{i,k+1}$$

To reduce computation time the coefficients a_{ij}, b_{ij}, c_{ij} and d_{ij} are saved as long as successive points (x,y) fall in the same four by four point region of a grid. In addition the four by four point interpolation regions are restricted to starting indices i and k with values $3n + 1$, $n = 0, 1 \ldots.$ Thus the regions do not overlap. This is less accurate than centering the region on each x,y point but requires less frequent computation of the coefficients. At the outer edges of a grid the regions are chosen to extend to the edge but not beyond. If x,y is out of the entire three grid region the nearest four by four point region is used for extrapolation.

The coefficients a_{ij}, b_{ij}, c_{ij} and d_{ij} are stored in two dimensional arrays from IT 106 to IT 109. When they are used, from IT 118 to

IT 149 they are used as simple variables (A(1,1) ≡ A11) to save time. Also the three dimensional arrays AR1, AR2, and AR3 are used as linear arrays from IT 92 to IT 105. The equivalent three subscripts are shown in the comment at IT 91.

SYMBOL DICTIONARY

Aij	= A(i,j) = a_{ij}
AR1	= ARL1 = grid 1
AR2	= ARL2 = grid 2
AR3	= ARL3 = grid 3
Bij	= B(i,j) = b_{ij}
Cij	= C(i,j) = c_{ij}
Dij	= D(i,j) = d_{ij}
DX	= Δx for grid being used
DXA	= array of Δx values for the three grids
DY	= Δy for grid being used
DYA	= array of Δy values
EPSCF	= $\varepsilon_1 - j\sigma/\omega\varepsilon_0$
F1	= I_ρ^V
F2	= I_Z^V
F3	= I_ρ^H
F4	= I_ϕ^H
FX1	= $f_{i,j}(x)$
FX2	= $f_{i,j+1}(x)$
FX3	= $f_{i,j+2}(x)$
FX4	= $f_{i,j+3}(x)$
IADD	= index for linear arrays ARL1, etc.
IADZ	= initial value for IADD
IGR	= grid number for present x,y
IGRS	= grid number for last x,y
IX	= x index of the grid coordinate just less than x
IXEG	= x index of the upper edge of the last normally located interpolation patch when a patch out of the

normal locations is used at the outer edge of a grid, -10000 otherwise

IXS	= 1 plus the x index of the lower edge of 4 by 4 point interpolation patch
IY, IYEG, IYS	= same for y as IX, IXEG and IXS
K	= 1, 2, 3, 4 for I^V_ρ, I^V_z, I^H_ρ, I^H_ϕ
ND	= NDA for the particular grid
NDA	= array containing the first dimensions of AR1, AR2 and AR3
NDP	= NDPA for a particular grid
NDPA	= array containing the product of the first two dimensions in AR1, AR2 and AR3
NXA	= number of x values in each grid
NXM2	= NXA - 2 for a particular grid
NXMS	= upper x index of the last normally located patch at the edge of a grid
NYA, NYM2, NYMS	= same for y as NXA, NXM2 and NXMS
P1, P2, P3, P4	= p_1, p_2, p_3, p_4
X	= x
XS	= XSA for the present grid
XS2	= XSA(2) through equivalence
XSA	= array of values of x at lower edge of each grid (s_i)
XX	= ξ_i
XZ	= x_{i+1} for computing ξ_i
Y	= y
YS	= YSA for present grid
YS3	= YSA(3) through equivalence
YSA	= array of values of y at lower edge of each grid (t_i)
YY	= η_k
YZ	= y_{k+1} for computing η_k

```
      SUBROUTINE INTRP (X,Y,F1,F2,F3,F4)
C
C     INTRP USES BIVARIATE CUBIC INTERPOLATION TO OBTAIN THE VALUES OF
C     4 FUNCTIONS AT THE POINT (X,Y).
C
      COMPLEX F1,F2,F3,F4,A,B,C,D,FX1,FX2,FX3,FX4,P1,P2,P3,P4,A11,A12,A1
     13,A14,A21,A22,A23,A24,A31,A32,A33,A34,A41,A42,A43,A44,B11,B12,B13,
     2B14,B21,B22,B23,B24,B31,B32,B33,B34,B41,B42,B43,B44,C11,C12,C13,C1
     34,C21,C22,C23,C24,C31,C32,C33,C34,C41,C42,C43,C44,D11,D12,D13,D14,
     4D21,D22,D23,D24,D31,D32,D33,D34,D41,D42,D43,D44
      COMPLEX AR1,AR2,AR3,ARL1,ARL2,ARL3,EPSCF
      COMMON /GGRID/ AR1(11,10,4),AR2(17,5,4),AR3(9,8,4),EPSCF,DXA(3),DY
     1A(3),XSA(3),YSA(3),NXA(3),NYA(3)
      DIMENSION NDA(3), NDPA(3)
      DIMENSION A(4,4), B(4,4), C(4,4), D(4,4), ARL1(1), ARL2(1), ARL3(1
     1)
      EQUIVALENCE (A(1,1),A11), (A(1,2),A12), (A(1,3),A13), (A(1,4),A14)
      EQUIVALENCE (A(2,1),A21), (A(2,2),A22), (A(2,3),A23), (A(2,4),A24)
      EQUIVALENCE (A(3,1),A31), (A(3,2),A32), (A(3,3),A33), (A(3,4),A34)
      EQUIVALENCE (A(4,1),A41), (A(4,2),A42), (A(4,3),A43), (A(4,4),A44)
      EQUIVALENCE (B(1,1),B11), (B(1,2),B12), (B(1,3),B13), (B(1,4),B14)
      EQUIVALENCE (B(2,1),B21), (B(2,2),B22), (B(2,3),B23), (B(2,4),B24)
      EQUIVALENCE (B(3,1),B31), (B(3,2),B32), (B(3,3),B33), (B(3,4),B34)
      EQUIVALENCE (B(4,1),B41), (B(4,2),B42), (B(4,3),B43), (B(4,4),B44)
      EQUIVALENCE (C(1,1),C11), (C(1,2),C12), (C(1,3),C13), (C(1,4),C14)
      EQUIVALENCE (C(2,1),C21), (C(2,2),C22), (C(2,3),C23), (C(2,4),C24)
      EQUIVALENCE (C(3,1),C31), (C(3,2),C32), (C(3,3),C33), (C(3,4),C34)
      EQUIVALENCE (C(4,1),C41), (C(4,2),C42), (C(4,3),C43), (C(4,4),C44)
      EQUIVALENCE (D(1,1),D11), (D(1,2),D12), (D(1,3),D13), (D(1,4),D14)
      EQUIVALENCE (D(2,1),D21), (D(2,2),D22), (D(2,3),D23), (D(2,4),D24)
      EQUIVALENCE (D(3,1),D31), (D(3,2),D32), (D(3,3),D33), (D(3,4),D34)
      EQUIVALENCE (D(4,1),D41), (D(4,2),D42), (D(4,3),D43), (D(4,4),D44)
      EQUIVALENCE (ARL1,AR1), (ARL2,AR2), (ARL3,AR3), (XS2,XSA(2)), (YS3
     1,YSA(3))
      DATA IXS,IYS,IGRS/-10,-10,-10/,DX,DY,XS,YS/1.,1.,0.,0./
      DATA NDA/11,17,9/,NDPA/110,85,72/,IXEG,IYEG/0,0/
      IF (X.LT.XS.OR.Y.LT.YS) GO TO 1
      IX=INT((X-XS)/DX)+1
      IY=INT((Y-YS)/DY)+1
C
C     IF POINT LIES IN SAME 4 BY 4 POINT REGION AS PREVIOUS POINT, OLD
C     VALUES ARE REUSED
C
      IF (IX.LT.IXEG.OR.IY.LT.IYEG) GO TO 1
      IF (IABS(IX-IXS).LT.2.AND.IABS(IY-IYS).LT.2) GO TO 12
C
C     DETERMINE CORRECT GRID AND GRID REGION
C
1     IF (X.GT.XS2) GO TO 2
      IGR=1
      GO TO 3
2     IGR=2
      IF (Y.GT.YS3) IGR=3
3     IF (IGR.EQ.IGRS) GO TO 4
      IGRS=IGR
      DX=DXA(IGRS)
      DY=DYA(IGRS)
      XS=XSA(IGRS)
      YS=YSA(IGRS)
      NXM2=NXA(IGRS)-2
      NYM2=NYA(IGRS)-2
      NXMS=((NXM2+1)/3)*3+1
      NYMS=((NYM2+1)/3)*3+1
      ND=NDA(IGRS)
```

```
      NDP=NDPA(IGRS)
      IX=INT((X-XS)/DX)+1
      IY=INT((Y-YS)/DY)+1
    4 IXS=((IX-1)/3)*3+2
      IF (IXS.LT.2) IXS=2
      IXEG=-10000
      IF (IXS.LE.NXM2) GO TO 5
      IXS=NXM2
      IXEG=NXMS
    5 IYS=((IY-1)/3)*3+2
      IF (IYS.LT.2) IYS=2
      IYEG=-10000
      IF (IYS.LE.NYM2) GO TO 6
      IYS=NYM2
      IYEG=NYMS
C
C     COMPUTE COEFFICIENTS OF 4 CUBIC POLYNOMIALS IN X FOR THE 4 GRID
C     VALUES OF Y FOR EACH OF THE 4 FUNCTIONS
C
    6 IADZ=IXS+(IYS-3)*ND-NDP
      DO 11 K=1,4
      IADZ=IADZ+NDP
      IADD=IADZ
      DO 11 I=1,4
      IADD=IADD+ND
      GO TO (7,8,9), IGRS
C     P1=AR1(IXS-1,IYS-2+I,K)
    7 P1=ARL1(IADD-1)
      P2=ARL1(IADD)
      P3=ARL1(IADD+1)
      P4=ARL1(IADD+2)
      GO TO 10
    8 P1=ARL2(IADD-1)
      P2=ARL2(IADD)
      P3=ARL2(IADD+1)
      P4=ARL2(IADD+2)
      GO TO 10
    9 P1=ARL3(IADD-1)
      P2=ARL3(IADD)
      P3=ARL3(IADD+1)
      P4=ARL3(IADD+2)
   10 A(I,K)=(P4-P1+3.*(P2-P3))*.1666666667
      B(I,K)=(P1-2.*P2+P3)*.5
      C(I,K)=P3-(2.*P1+3.*P2+P4)*.1666666667
   11 D(I,K)=P2
      XZ=(IXS-1)*DX+XS
      YZ=(IYS-1)*DY+YS
C
C     EVALUATE POLYMOMIALS IN X AND THEN USE CUBIC INTERPOLATION IN Y
C     FOR EACH OF THE 4 FUNCTIONS.
C
   12 XX=(X-XZ)/DX
      YY=(Y-YZ)/DY
      FX1=((A11*XX+B11)*XX+C11)*XX+D11
      FX2=((A21*XX+B21)*XX+C21)*XX+D21
      FX3=((A31*XX+B31)*XX+C31)*XX+D31
      FX4=((A41*XX+B41)*XX+C41)*XX+D41
      P1=FX4-FX1+3.*(FX2-FX3)
      P2=3.*(FX1-2.*FX2+FX3)
      P3=6.*FX3-2.*FX1-3.*FX2-FX4
      F1=((P1*YY+P2)*YY+P3)*YY*.1666666667+FX2
      FX1=((A12*XX+B12)*XX+C12)*XX+D12
      FX2=((A22*XX+B22)*XX+C22)*XX+D22
      FX3=((A32*XX+B32)*XX+C32)*XX+D32
```

```
      FX4=((A42*XX+B42)*XX+C42)*XX+D42
      P1=FX4-FX1+3.*(FX2-FX3)
      P2=3.*(FX1-2.*FX2+FX3)
      P3=6.*FX3-2.*FX1-3.*FX2-FX4
      F2=((P1*YY+P2)*YY+P3)*YY*.1666666667+FX2
      FX1=((A13*XX+B13)*XX+C13)*XX+D13
      FX2=((A23*XX+B23)*XX+C23)*XX+D23
      FX3=((A33*XX+B33)*XX+C33)*XX+D33
      FX4=((A43*XX+B43)*XX+C43)*XX+D43
      P1=FX4-FX1+3.*(FX2-FX3)
      P2=3.*(FX1-2.*FX2+FX3)
      P3=6.*FX3-2.*FX1-3.*FX2-FX4
      F3=((P1*YY+P2)*YY+P3)*YY*.1666666667+FX2
      FX1=((A14*XX+B14)*XX+C14)*XX+D14
      FX2=((A24*XX+B24)*XX+C24)*XX+D24
      FX3=((A34*XX+B34)*XX+C34)*XX+D34
      FX4=((A44*XX+B44)*XX+C44)*XX+D44
      P1=FX4-FX1+3.*(FX2-FX3)
      P2=3.*(FX1-2.*FX2+FX3)
      P3=6.*FX3-2.*FX1-3.*FX2-FX4
      F4=((P1*YY+P2)*YY+P3)*YY*.1666666667+FX2
      RETURN
      END
```

INTX

PURPOSE

To numerically compute the integral of the function exp(jkr)/kr.

METHOD

For evaluation of the field due to a segment, a local cylindrical coordinate system is defined with origin at the center of the segment and z axis in the segment direction. This geometry is illustrated in the discussion of subroutine GF. Subroutine INTX is called by subroutine EFLD to evaluate the integral

$$G = \int_{-k\Delta/2}^{k\Delta/2} \frac{\exp(-jkr)}{kr}\, d(kz) \ ,$$

where

$$r = [\rho'^2 + (z - z')^2]^{1/2} \ ,$$

and other symbols are defined in the discussion of subroutine GF.

The numerical integration technique of Romberg integration with variable interval width is used (refs. 3 and 4). The Romberg integration formula is obtained from the trapezoidal formula by an iterative procedure (ref. 1). The trapezoidal rule for integration of the function f(x) over an interval (a, b) using 2^k subintervals is

$$T_{0k} = [(b - a)/N][(1/2)\ f_0 + f_1 + \ldots + f_{N-1} + (1/2)f_N] \ ,$$

where

$$N = 2^k$$
$$f_i = f(x_i)$$
$$x_i = a + i(b - a)/N$$

These trapezoidal-rule answers are then used in the iterative formula

$$T_{m,n} = \left(4^m T_{m-1,n+1} - T_{m-1,n}\right)/(4^m - 1) \ .$$

The results $T_{m,n}$ may be arranged in a triangular matrix of the form

$$\begin{matrix} T_{0,0} & & \\ T_{0,1} & T_{1,0} & \\ T_{0,2} & T_{1,1} & T_{2,0} \\ \vdots & \vdots & \vdots \end{matrix}$$

where the elements in the first column, T_{0k}, represent the trapezoidal rule results, and the elements in the diagonal, T_{k0}, are the Romberg integration results for 2^k subintervals.

Convergence to increasingly more accurate answers takes place down the first column and the diagonal, as well as towards the right along the rows. The row convergence generally provides a more realistic indication of error magnitude than two successive trapezoidal-rule or Romberg answers.

This convergence along the rows is used to determine the interval width in the variable interval-width scheme. The complete integration interval is first divided into a minimum number of subintervals (presently set to 1) and T_{00}, T_{01}, and T_{10} are computed on the first subinterval. The relative difference of T_{01} and T_{10} is then computed, and if less than the error criterion, R_x, T_{10} is accepted as the integral over that interval, and integration proceeds to the next interval. If the difference of T_{01} and T_{10} is too great, T_{02}, T_{11} and T_{20} are computed. The relative difference of T_{11} and T_{20} is then computed, and if less than R_x, T_{20} is accepted as the integral over the subinterval. If the difference of T_{11} and T_{20} is too great, the subinterval is divided in half and the process repeated starting with T_{00} for the left hand, new subinterval. The subinterval is repeatedly halved until convergence to less than R_x is found. The process is repeated for successive subintervals until the right-hand side of the integration interval is reached. When convergence has been obtained with a given subinterval size for a few times, the routine attempts doubling the subinterval size to maintain the largest subinterval size that will give the required accuracy. Thus, the routine will use many points in a rapidly changing region of a function and fewer points where the function is smoothly varying.

Since the function to be integrated is complex, the convergence of both real and imaginary parts is tested and both must be less than R_x. The same subinterval sizes are used for real and imaginary parts.

When the field of a segment is being computed at the segment's own center, the length r becomes

$$r = [b^2 + (z - z')^2]^{1/2} ,$$

where b is the wire radius. For small values of b, the real part of the integrand is sharply peaked and, hence, difficult to integrate numerically. Hence, the integral is divided into the components

$$G' = \int_{-k\Delta/2}^{k\Delta/2} \frac{\exp(-jkr) - 1}{kr} d(kz)$$

$$G'' = \int_{-k\Delta/2}^{k\Delta/2} \frac{1}{kr} d(kz)$$

$$G = G' + G''$$

G' must be computed numerically; however, the integrand is no longer peaked. G", which contains the sharp peak, can be computed as

$$G'' = 2 \log \left(\frac{\sqrt{b^2 + \Delta^2} + \Delta}{b} \right)$$

To further reduce integration time for the self term, the integral of G' is computed from $-k\Delta/2$ to 0, and the result doubled to obtain G'.

SYMBOL DICTIONARY

ABS = external routine (absolute value)
ALOG = external routine (natural log)
B = wire radius, b/λ
DZ = subinterval size on which T_{00}, T_{01}, ... are computed
DZOT = 0.5 DZ
EL1 = $-k\Delta/2$
EL2 = $k\Delta/2$
EP = tolerance for ending the integration interval
FNM = real number equivalent of NM
FNS = real number equivalent of NS
GF = external routine (integrand)
G1I = imaginary part of f_1
G1R = real part of f_1

G2I = imaginary part of f_2
G2R = real part of f_2
G3I = imaginary part of f_3
G3R = real part of f_3
G4I = imaginary part of f_4
G4R = real part of f_4
G5I = imaginary part of f_5
G5R = real part of f_5
IJ = indication of self term integration when equal to zero
NM = minimum allowed subinterval size is $k\Delta/NM$
NS = present subinterval size is $k\Delta/NS$
NT = counter to control increasing of subinterval size
NTS = larger values retard increasing of subinterval size
NX = maximum allowed subinterval size is $k\Delta/NX$
RX = R_x
S = Δ/λ
SGI = imaginary part of G
SGR = real part of G
SQRT = external routine (square root)
TEST = external routine (computes relative convergence)
TE1I = relative difference of T_{01} and T_{10} for imaginary part
TE1R = relative difference of T_{01} and T_{10} for real part
TE2I = relative difference of T_{11} and T_{20} for imaginary part
TE2R = relative difference of T_{11} and T_{20} for real part
T00I = imaginary part T_{00}
T00R = real part T_{00}
T01I = imaginary part T_{01}
T01R = real part T_{01}
T02I = imaginary part T_{02}
T02R = real part T_{02}
T10I = imaginary part T_{10}
T10R = real part of T_{10}
T11I = imaginary part of T_{11}
T11R = real part of T_{11}
T20I = imaginary part of T_{20}
T20R = real part of T_{20}

Z = integration variable at left-hand side of subinterval

ZE = $k\Delta/2$

ZEND = $k\Delta/2$ - EP; EP = tolerance term

ZP = integration variable

CONSTANTS

65536 = 2^{16} = limit of minimum subinterval size (NM)

1.E-4 = error criterion, R_x

```
      SUBROUTINE INTX (EL1,EL2,B,IJ,SGR,SGI)
C
C     INTX PERFORMS NUMERICAL INTEGRATION OF EXP(JKR)/R BY THE METHOD OF
C     VARIABLE INTERVAL WIDTH ROMBERG INTEGRATION.  THE INTEGRAND VALUE
C     IS SUPPLIED BY SUBROUTINE GF.
C
      DATA NX,NM,NTS,RX/1,65536,4,1.E-4/
      Z=EL1
      ZE=EL2
      IF (IJ.EQ.0) ZE=0.
      S=ZE-Z
      FNM=NM
      EP=S/(10.*FNM)
      ZEND=ZE-EP
      SGR=0.
      SGI=0.
      NS=NX
      NT=0
      CALL GF (Z,G1R,G1I)
1     FNS=NS
      DZ=S/FNS
      ZP=Z+DZ
      IF (ZP-ZE) 3,3,2
2     DZ=ZE-Z
      IF (ABS(DZ)-EP) 17,17,3
3     DZOT=DZ*.5
      ZP=Z+DZOT
      CALL GF (ZP,G3R,G3I)
      ZP=Z+DZ
      CALL GF (ZP,G5R,G5I)
4     T00R=(G1R+G5R)*DZOT
      T00I=(G1I+G5I)*DZOT
      T01R=(T00R+DZ*G3R)*0.5
      T01I=(T00I+DZ*G3I)*0.5
      T10R=(4.0*T01R-T00R)/3.0
      T10I=(4.0*T01I-T00I)/3.0
C
C     TEST CONVERGENCE OF 3 POINT ROMBERG RESULT.
C
      CALL TEST (T01R,T10R,TE1R,T01I,T10I,TE1I,0.)
      IF (TE1I-RX) 5,5,6
5     IF (TE1R-RX) 8,8,6
6     ZP=Z+DZ*0.25
      CALL GF (ZP,G2R,G2I)
      ZP=Z+DZ*0.75
      CALL GF (ZP,G4R,G4I)
      T02R=(T01R+DZOT*(G2R+G4R))*0.5
      T02I=(T01I+DZOT*(G2I+G4I))*0.5
      T11R=(4.0*T02R-T01R)/3.0
      T11I=(4.0*T02I-T01I)/3.0
      T20R=(16.0*T11R-T10R)/15.0
      T20I=(16.0*T11I-T10I)/15.0
C
C     TEST CONVERGENCE OF 5 POINT ROMBERG RESULT.
C
      CALL TEST (T11R,T20R,TE2R,T11I,T20I,TE2I,0.)
      IF (TE2I-RX) 7,7,14
7     IF (TE2R-RX) 9,9,14
8     SGR=SGR+T10R
      SGI=SGI+T10I
      NT=NT+2
      GO TO 10
9     SGR=SGR+T20R
      SGI=SGI+T20I
```

```
      NT=NT+1
   10 Z=Z+DZ
      IF (Z-ZEND) 11,17,17
   11 G1R=G5R
      G1I=G5I
      IF (NT-NTS) 1,12,12
   12 IF (NS-NX) 1,1,13
C
C     DOUBLE STEP SIZE
C
   13 NS=NS/2
      NT=1
      GO TO 1
   14 NT=0
      IF (NS-NM) 16,15,15
   15 PRINT 20, Z
      GO TO 9
C
C     HALVE STEP SIZE
C
   16 NS=NS*2
      FNS=NS
      DZ=S/FNS
      DZOT=DZ*0.5
      G5R=G3R
      G5I=G3I
      G3R=G2R
      G3I=G2I
      GO TO 4
   17 CONTINUE
      IF (IJ) 19,18,19
C
C     ADD CONTRIBUTION OF NEAR SINGULARITY FOR DIAGONAL TERM
C
   18 SGR=2.*(SGR+ALOG((SQRT(B*B+S*S)+S)/B))
      SGI=2.*SGI
   19 CONTINUE
      RETURN
C
   20 FORMAT (24H STEP SIZE LIMITED AT Z=,F10.5)
      END
```

ISEGNO

PURPOSE

To determine the segment number of the m^{th} segment ordered by increasing segment numbers in the set of segments with tag numbers equal to the given tag number. With a given tag of zero, segment number m is returned.

METHOD

Search segments consecutively and check their tag numbers against a given tag.

SYMBOL DICTIONARY

I = DO loop index

ICNT = counter

ITAGI = input tag number (given tag)

M = input quantity specifying the position in the set of segments with the given tag

CODE LISTING

```
      FUNCTION ISEGNO (ITAGI,MX)
C
C     ISEGNO RETURNS THE SEGMENT NUMBER OF THE MTH SEGMENT HAVING THE
C     TAG NUMBER ITAGI.  IF ITAGI=0 SEGMENT NUMBER M IS RETURNED.
C
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      IF (MX.GT.0) GO TO 1
      PRINT 6
      STOP
1     ICNT=0
      IF (ITAGI.NE.0) GO TO 2
      ISEGNO=MX
      RETURN
2     IF (N.LT.1) GO TO 4
      DO 3 I=1,N
      IF (ITAG(I).NE.ITAGI) GO TO 3
      ICNT=ICNT+1
      IF (ICNT.EQ.MX) GO TO 5
3     CONTINUE
4     PRINT 7, ITAGI
      STOP
5     ISEGNO=I
      RETURN
C
6     FORMAT (4X,91HCHECK DATA, PARAMETER SPECIFYING SEGMENT POSITION IN
     1 A GROUP OF EQUAL TAGS MUST NOT BE ZERO)
7     FORMAT (///,10X,26HNO SEGMENT HAS AN ITAG OF ,I5)
      END
```

LFACTR

PURPOSE

To perform the Gauss-Doolittle factorization calculations on two blocks of the matrix in core storage. This routine in conjunction with FACIO factors a matrix that is too large for core storage into an upper and lower triangular matrix using the Gauss-Doolittle technique. The factored matrix is used by LUNSCR and LTSOLV to determine the solution of the transposed matrix equation $x^T A^T = B^T$.

METHOD

The basic algorithm used in this routine is presented by Ralston in ref. 1 on pages 411-416. A brief discussion is also given under FACTR in this manual. The main difference between LFACTR and FACTR is that LFACTR is set up to perform the calculations on two blocks of columns of the transposed matrix that reside in core storage. This situation arises when the matrix is too large to fit in core at one time; thus, the matrix is divided into blocks of columns and stored on files. This matrix is then factored into a lower triangular matrix and an upper triangular matrix by the subroutines FACIO and LFACTR. The function of these two subroutines is closely tied together: LFACTR performs the mathematical computations involved in the factorization, while FACIO controls the input and output of matrix blocks in core storage, and, thus, controls the necessary block ordering input to LFACTR. For clarification of the ordering of matrix blocks during factorization, refer to FACIO.

The computations performed in LFACTR are slightly different for three matrix block conditions: (1) block numbers 1 and 2, (2) adjacent matrix blocks, and (3) non-adjacent matrix blocks. If the blocks are numbers 1 and 2, both blocks are factored, and the computations proceed exactly as in FACTR. The only difference between LFACTR and FACTR here is that the two blocks do not form a square matrix, and the row and column indices in LFACTR have not been interchanged as in FACTR. At the end of this stage, both blocks 1 and 2 are completely factored. For case 2, where the blocks are adjacent in the matrix and other than 1 and 2, the first block is assumed factored and is used to complete the factorization of the partially factored second block. The computations start with the first column of the second block and proceed as in FACTR (with the exceptions noted above). If the blocks are not adjacent (case 3), the first block is assumed factored and is used to partially factor the second block. Computations start with the first column of the second block. Factorization cannot be completed, since values from the intervening columns are necessary.

CODING

- LF20 - LF39 Initialization of loop parameters for the various matrix block conditions.
- LF40 - LF99 Loop over columns to be factored or partially factored.
- LF44 - LF46 Write column of A in scratch vector D.
- LF49 - LF62 Computations for u_{ir} (see FACTR), where positioning for size is taken into account. The range of i is determined by the matrix blocks used.
- LF69 - LF71 For case 3, the partially factored column is stored in A, and a jump to LF100 is made.
- LF73 - LF87 For cases 1 and 2, the maximum value in the column is found for positioning.
- LF92 - LF94 For cases 1 and 2, ℓ_{ir} (see FACTR) is calculated; limits on i are dependent on blocks.

SYMBOL DICTIONARY

- A = array which contains the two blocks of columns of the transposed matrix in some state of factorization
- CONJG = external routine (conjugate of complex numbers)
- D = scratch vector, temporary storage of one column
- DMAX = maximum value in column
- ELMAG = intermediate variable
- I = DO loop index
- IFLG = small pivot value flag
- IP = array containing positioning information
- IXJ = index
- IX1 = first block number, input
- IX2 = second block number, input
- J = DO loop index
- JP1 = J + 1
- J1, J2 = DO loop limits
- J2P1 = J2 + 1

J2P2 = J2 + 2

K = DO loop index

L1, L2, L3 = logical variables for testing

NCOL = number of columns

NROW = number of rows

PJ, PR = intermediate variables

R = DO loop index

REAL = external routine (real part of a complex number)

R1, R2 = DO loop limits, relative column number limits for calculations

In programs using double precision accumulation in the matrix solution, the following double precision variables are used in LFACTR.

DAR1, DAI1, DAR2, DAI2 = real and imaginary parts of a number for temporary storage

DR, DI = real and imaginary vectors replacing the complex vector D in single precision programs

CONSTANT

1.E-10 = small value test

```
      SUBROUTINE LFACTR (A,NROW,IX1,IX2,IP)
C
C     LFACTR PERFORMS GAUSS-DOOLITTLE MANIPULATIONS ON THE TWO BLOCKS OF
C     THE TRANSPOSED MATRIX IN CORE STORAGE.  THE GAUSS-DOOLITTLE
C     ALGORITHM IS PRESENTED ON PAGES 411-416 OF A. RALSTON -- A FIRST
C     COURSE IN NUMERICAL ANALYSIS.  COMMENTS BELOW REFER TO COMMENTS IN
C     RALSTONS TEXT.
C
      COMPLEX A,D,AJR
      INTEGER R,R1,R2,PJ,PR
      LOGICAL L1,L2,L3
      COMMON /MATPAR/ ICASE,NBLOKS,NPBLK,NLAST,NBLSYM,NPSYM,NLSYM,IMAT,I
     1CASX,NBBX,NPBX,NLBX,NBBL,NPBL,NLBL
      COMMON /SCRATM/ D(600)
      DIMENSION A(NROW,1), IP(NROW)
      IFLG=0
C
C     INITIALIZE R1,R2,J1,J2
C
      L1=IX1.EQ.1.AND.IX2.EQ.2
      L2=(IX2-1).EQ.IX1
      L3=IX2.EQ.NBLSYM
      IF (L1) GO TO 1
      GO TO 2
1     R1=1
      R2=2*NPSYM
      J1=1
      J2=-1
      GO TO 5
2     R1=NPSYM+1
      R2=2*NPSYM
      J1=(IX1-1)*NPSYM+1
      IF (L2) GO TO 3
      GO TO 4
3     J2=J1+NPSYM-2
      GO TO 5
4     J2=J1+NPSYM-1
5     IF (L3) R2=NPSYM+NLSYM
      DO 16 R=R1,R2
C
C     STEP 1
C
      DO 6 K=J1,NROW
      D(K)=A(K,R)
6     CONTINUE
C
C     STEPS 2 AND 3
C
      IF (L1.OR.L2) J2=J2+1
      IF (J1.GT.J2) GO TO 9
      IXJ=0
      DO 8 J=J1,J2
      IXJ=IXJ+1
      PJ=IP(J)
      AJR=D(PJ)
      A(J,R)=AJR
      D(PJ)=D(J)
      JP1=J+1
      DO 7 I=JP1,NROW
      D(I)=D(I)-A(I,IXJ)*AJR
7     CONTINUE
8     CONTINUE
9     CONTINUE
C
```

```
C       STEP 4
C
        J2P1=J2+1
        IF (L1.OR.L2) GO TO 11
        IF (NROW.LT.J2P1) GO TO 16
        DO 10 I=J2P1,NROW
        A(I,R)=D(I)
   10   CONTINUE
        GO TO 16
   11   DMAX=REAL(D(J2P1)*CONJG(D(J2P1)))
        IP(J2P1)=J2P1
        J2P2=J2+2
        IF (J2P2.GT.NROW) GO TO 13
        DO 12 I=J2P2,NROW
        ELMAG=REAL(D(I)*CONJG(D(I)))
        IF (ELMAG.LT.DMAX) GO TO 12
        DMAX=ELMAG
        IP(J2P1)=I
   12   CONTINUE
   13   CONTINUE
        IF (DMAX.LT.1.E-10) IFLG=1
        PR=IP(J2P1)
        A(J2P1,R)=D(PR)
        D(PR)=D(J2P1)
C
C       STEP 5
C
        IF (J2P2.GT.NROW) GO TO 15
        AJR=1./A(J2P1,R)
        DO 14 I=J2P2,NROW
        A(I,R)=D(I)*AJR
   14   CONTINUE
   15   CONTINUE
        IF (IFLG.EQ.0) GO TO 16
        PRINT 17, J2,DMAX
        IFLG=0
   16   CONTINUE
        RETURN
C
   17   FORMAT (1H ,6HPIVOT(,I3,2H)=,E16.8)
        END
```

LOAD

PURPOSE

To compute the impedances at a given frequency for the loading specified by LD cards.

METHOD

The value of $\lambda Z/\Delta$, where Z is the total impedance on a segment and Δ is the length of the segment, is computed for each loaded segment and stored in the array ZARRAY. The proper impedance formula is chosen by the value of the input quantity LDTYP. These computations are performed from the sequence L074 to L096 of the program, and the formulas are:

LDTYP = 0 (series R, L, and C):

$$Z = R + j\omega L + \frac{1}{j\omega C}$$

$$Z' = \frac{\lambda Z}{\Delta} = \frac{R}{\frac{\Delta}{\lambda}} + j2\pi c \left(\frac{L}{\Delta}\right) + \frac{1}{j2\pi c \left(\frac{\Delta}{\lambda}\right)^2 \left(\frac{C}{\Delta}\right)}$$

where c is the speed of light and R, L, and C are input.

LDTYP = 1 (parallel R, L, and C; R, L, and C input):

$$Z' = \frac{1}{\left(\frac{\Delta}{\lambda}\right) \frac{1}{R} + \frac{\Delta}{j2\pi cL} + j2\pi c \left(\frac{\Delta}{\lambda}\right)^2 \left(\frac{C}{\Delta}\right)}$$

LDTYP = 2 and 3 (same as above, but R/Δ, L/Δ, C/Δ are input)

LDTYP = 4 (resistance and reactance input):

$$Z' = \frac{\text{resistance} + j\ \text{reactance}}{\frac{\Delta}{\lambda}}$$

LDTYP = 5 (call another subroutine for wire conductivity calculation)

SYMBOL DICTIONARY

ABS = external routine (absolute value of a real number)

AIMAG = external routine (imaginary part of a complex number)

CMPLX = external routine (forms a complex number)

ICHK = check flag in diagnosing data errors

ISTEP = loading card subscript

IWARN = flag checking for multiply loaded segments

JUMP = LDTYP + 1

LDTAG = tag number, input quantity

LDTAGF = input quantity

LDTAGS = LDTAG(ISTEP)

LDTAGT = input quantity

LDTYP = input quantity specifying loading type

NLOAD = number of input loading data cards

PRNT = external routine (prints the impedance data in a table)

REAL = external routine (takes the real part of a complex number)

TPCJ = $j2\pi c$, where c is the speed of light

ZARRAY = array containing $\lambda Z/\Delta$ for each segment, dimensioned to the maximum number of segments

ZINT = external routine (calculates the internal impedance of a finitely conducting wire)

ZLC, ZLI, ZLR = input quantities, the definitions are a function of the type of loading specified. For the case of series RLC (LDTYP = 0): ZLC = capacitance (farads), ZLI = inductance (henrys), and ZLR = resistance (ohms). For the remaining cases, see Part III.

ZT = $Z' = \lambda Z/\Delta$ for one segment; however, variable name is used during the calculation of this quantity

CONSTANTS

1.E-20 = floating point zero test

(0., 1.88365371E+9) = $j2\pi c$, where c is the velocity of light

```
      SUBROUTINE LOAD (LDTYP,LDTAG,LDTAGF,LDTAGT,ZLR,ZLI,ZLC)
C
C     LOAD CALCULATES THE IMPEDANCE OF SPECIFIED SEGMENTS FOR VARIOUS
C     TYPES OF LOADING
C
      COMPLEX ZARRAY,ZT,TPCJ,ZINT
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /ZLOAD/ ZARRAY(300),NLOAD,NLODF
      DIMENSION LDTYP(1), LDTAG(1), LDTAGF(1), LDTAGT(1), ZLR(1), ZLI(1)
     1, ZLC(1), TPCJX(2)
      EQUIVALENCE (TPCJ,TPCJX)
      DATA TPCJX/0.,1.883698955E+9/
C
C     PRINT HEADING
C
      PRINT 25
C
C     INITIALIZE D ARRAY, USED FOR TEMPORARY STORAGE OF LOADING
C     INFORMATION.
C
      DO 1 I=N2,N
    1 ZARRAY(I)=(0.,0.)
      IWARN=0
C
C     CYCLE OVER LOADING CARDS
C
      ISTEP=0
    2 ISTEP=ISTEP+1
      IF (ISTEP.LE.NLOAD) GO TO 5
      IF (IWARN.EQ.1) PRINT 26
      IF (N1+2*M1.GT.0) GO TO 4
      NOP=N/NP
      IF (NOP.EQ.1) GO TO 4
      DO 3 I=1,NP
      ZT=ZARRAY(I)
      L1=I
      DO 3 L2=2,NOP
      L1=L1+NP
    3 ZARRAY(L1)=ZT
    4 RETURN
    5 IF (LDTYP(ISTEP).LE.5) GO TO 6
      PRINT 27, LDTYP(ISTEP)
      STOP
    6 LDTAGS=LDTAG(ISTEP)
      JUMP=LDTYP(ISTEP)+1
      ICHK=0
C
C     SEARCH SEGMENTS FOR PROPER ITAGS
C
      L1=N2
      L2=N
      IF (LDTAGS.NE.0) GO TO 7
      IF (LDTAGF(ISTEP).EQ.0.AND.LDTAGT(ISTEP).EQ.0) GO TO 7
      L1=LDTAGF(ISTEP)
      L2=LDTAGT(ISTEP)
      IF (L1.GT.N1) GO TO 7
      PRINT 29
      STOP
    7 DO 17 I=L1,L2
      IF (LDTAGS.EQ.0) GO TO 8
      IF (LDTAGS.NE.ITAG(I)) GO TO 17
```

```
      IF (LDTAGF(ISTEP).EQ.0) GO TO 8
      ICHK=ICHK+1
      IF (ICHK.GE.LDTAGF(ISTEP).AND.ICHK.LE.LDTAGT(ISTEP)) GO TO 9
      GO TO 17
    8 ICHK=1
C
C     CALCULATION OF LAMDA*IMPED. PER UNIT LENGTH, JUMP TO APPROPRIATE
C     SECTION FOR LOADING TYPE
C
    9 GO TO (10,11,12,13,14,15), JUMP
   10 ZT=ZLR(ISTEP)/SI(I)+TPCJ*ZLI(ISTEP)/(SI(I)*WLAM)
      IF (ABS(ZLC(ISTEP)).GT.1.E-20) ZT=ZT+WLAM/(TPCJ*SI(I)*ZLC(ISTEP))
      GO TO 16
   11 ZT=TPCJ*SI(I)*ZLC(ISTEP)/WLAM
      IF (ABS(ZLI(ISTEP)).GT.1.E-20) ZT=ZT+SI(I)*WLAM/(TPCJ*ZLI(ISTEP))
      IF (ABS(ZLR(ISTEP)).GT.1.E-20) ZT=ZT+SI(I)/ZLR(ISTEP)
      ZT=1./ZT
      GO TO 16
   12 ZT=ZLR(ISTEP)*WLAM+TPCJ*ZLI(ISTEP)
      IF (ABS(ZLC(ISTEP)).GT.1.E-20) ZT=ZT+1./(TPCJ*SI(I)*SI(I)*ZLC(ISTE
     1P))
      GO TO 16
   13 ZT=TPCJ*SI(I)*SI(I)*ZLC(ISTEP)
      IF (ABS(ZLI(ISTEP)).GT.1.E-20) ZT=ZT+1./(TPCJ*ZLI(ISTEP))
      IF (ABS(ZLR(ISTEP)).GT.1.E-20) ZT=ZT+1./(ZLR(ISTEP)*WLAM)
      ZT=1./ZT
      GO TO 16
   14 ZT=CMPLX(ZLR(ISTEP),ZLI(ISTEP))/SI(I)
      GO TO 16
   15 ZT=ZINT(ZLR(ISTEP)*WLAM,BI(I))
   16 IF ((ABS(REAL(ZARRAY(I)))+ABS(AIMAG(ZARRAY(I)))).GT.1.E-20) IWARN=
     11
      ZARRAY(I)=ZARRAY(I)+ZT
   17 CONTINUE
      IF (ICHK.NE.0) GO TO 18
      PRINT 28, LDTAGS
      STOP
C
C     PRINTING THE SEGMENT LOADING DATA, JUMP TO PROPER PRINT
C
   18 GO TO (19,20,21,22,23,24), JUMP
   19 CALL PRNT (LDTAGS,LDTAGF(ISTEP),LDTAGT(ISTEP),ZLR(ISTEP),ZLI(ISTEP
     1),ZLC(ISTEP),0.,0.,0.,7H SERIES,7)
      GO TO 2
   20 CALL PRNT (LDTAGS,LDTAGF(ISTEP),LDTAGT(ISTEP),ZLR(ISTEP),ZLI(ISTEP
     1),ZLC(ISTEP),0.,0.,0.,8HPARALLEL,8)
      GO TO 2
   21 CALL PRNT (LDTAGS,LDTAGF(ISTEP),LDTAGT(ISTEP),ZLR(ISTEP),ZLI(ISTEP
     1),ZLC(ISTEP),0.,0.,0.,18HSERIES (PER METER),18)
      GO TO 2
   22 CALL PRNT (LDTAGS,LDTAGF(ISTEP),LDTAGT(ISTEP),ZLR(ISTEP),ZLI(ISTEP
     1),ZLC(ISTEP),0.,0.,0.,20HPARALLEL (PER METER),20)
      GO TO 2
   23 CALL PRNT (LDTAGS,LDTAGF(ISTEP),LDTAGT(ISTEP),0.,0.,0.,ZLR(ISTEP),
     1ZLI(ISTEP),0.,15HFIXED IMPEDANCE,15)
      GO TO 2
   24 CALL PRNT (LDTAGS,LDTAGF(ISTEP),LDTAGT(ISTEP),0.,0.,0.,0.,0.,ZLR(I
     1STEP),6H  WIRE,6)
      GO TO 2
C
   25 FORMAT (//,7X,8HLOCATION,10X,10HRESISTANCE,3X,10HINDUCTANCE,2X,11H
     1CAPACITANCE,7X,16HIMPEDANCE (OHMS),5X,12HCONDUCTIVITY,4X,4HTYPE,/,
     24X,4HITAG,10H FROM THRU,10X,4HOHMS,8X,6HHENRYS,7X,6HFARADS,8X,4HRE
     3AL,6X,9HIMAGINARY,4X,10HMHOS/METER)
```

```
26    FORMAT (/,10X,74HNOTE, SOME OF THE ABOVE SEGMENTS HAVE BEEN LOADED LO 128
     1 TWICE - IMPEDANCES ADDED)                                        LO 129
27    FORMAT (/,10X,46HIMPROPER LOAD TYPE CHOOSEN, REQUESTED TYPE IS ,I3 LO 130
     1)                                                                 LO 131
28    FORMAT (/,10X,50HLOADING DATA CARD ERROR, NO SEGMENT HAS AN ITAG = LO 132
     1 ,I5)                                                             LO 133
29    FORMAT (63H ERROR - LOADING MAY NOT BE ADDED TO SEGMENTS IN N.G.F. LO 134
     1 SECTION)                                                         LO 135
      END                                                               LO 136-
```

LTSOLV

PURPOSE

To solve the matrix equation $X^R LU = B^R$, where R denotes a row vector and L and U are the lower and upper triangular matrices stored as blocks on files.

METHOD

The L and U triangular matrices are written in a square array, where the 1's on the diagonal of the L matrix are suppressed. The array is stored by blocks of columns in ascending order on file IFL1 and descending order on file IFL2. The solution procedure is as follows. First solve the equation

$$Y^R U = B^R \tag{1}$$

then

$$X^R L = Y^R \ , \tag{2}$$

since $X^R LU = B^R$. The solutions of equations (1) and (2) are straightforward, since both matrices are triangular. In particular for equation (1),

$$y_j^R = \frac{1}{u_{jj}} \left(b_j^R - \sum_{i=1}^{j-1} y_i^R \, u_{ij} \right) \qquad j = 1, \ldots, n$$

and similarly for equation (2).

Several right-hand side vectors may be stored in the two dimensional array B. The forward and backward substitution is then done on each vector in the loops from LT 23 to LT 34 and LT 43 to LT 56. This can be much faster than calling LTSOLV for each vector since the files IFL1 and IFL2 are read only once. This feature is used in computing $A^{-1}B$ for the NGF solution. It is not used with the multiple excitations for a receiving pattern or to compute the driving point interaction matrix in NETWK but could reduce the out-of-core solution time in these cases.

Row interchanges were used to position elements for size in factoring the transposed structure matrix; therefore, the elements in the solution vector X^R are not in the original locations. Using the IX array (filled by LUNSCR), the vector can be put back into the original order. The integer contained in IX(J) is the index of the original location of the parameter now in the j^{th} location. The solution vector is overwritten on the input right-hand side vector B^R.

SYMBOL DICTIONARY

A	=	array for matrix blocks
B	=	B^R, right-hand side and solution
I2	=	number of words in a block
IFL1	=	file with blocks in normal order
IFL2	=	file with blocks in reversed order
IX	=	solution unscramble vector
IXBLK1	=	block number
J	=	row index
JST	=	initial value for J
K2	=	number of columns in a block
KP	=	column index
NEQ	=	total number of equations
NRH	=	number of right-hand side vectors in B
NROW	=	row dimension of A (number of equations in a symmetric section)
SUM	=	summation result

```
      SUBROUTINE LTSOLV (A,NROW,IX,B,NEQ,NRH,IFL1,IFL2)                 LT   1
C                                                                       LT   2
C     LTSOLV SOLVES THE MATRIX EQ. Y(R)*LU(T)=B(R) WHERE (R) DENOTES ROW LT   3
C     VECTOR AND LU(T) DENOTES THE LU DECOMPOSITION OF THE TRANSPOSE OF  LT   4
C     THE ORIGINAL COEFFICIENT MATRIX.  THE LU(T) DECOMPOSITION IS       LT   5
C     STORED ON TAPE 5 IN BLOCKS IN ASCENDING ORDER AND ON FILE 3 IN     LT   6
C     BLOCKS OF DESCENDING ORDER.                                        LT   7
C                                                                       LT   8
      COMPLEX A,B,Y,SUM                                                  LT   9
      COMMON /MATPAR/ ICASE,NBLOKS,NPBLK,NLAST,NBLSYM,NPSYM,NLSYM,IMAT,I LT  10
     1CASX,NBBX,NPBX,NLBX,NBBL,NPBL,NLBL                                 LT  11
      COMMON /SCRATM/ Y(600)                                             LT  12
      DIMENSION A(NROW,NROW), B(NEQ,NRH), IX(NEQ)                        LT  13
C                                                                       LT  14
C     FORWARD SUBSTITUTION                                               LT  15
C                                                                       LT  16
      I2=2*NPSYM*NROW                                                    LT  17
      DO 4 IXBLK1=1,NBLSYM                                               LT  18
      CALL BLCKIN (A,IFL1,1,I2,1,121)                                    LT  19
      K2=NPSYM                                                           LT  20
      IF (IXBLK1.EQ.NBLSYM) K2=NLSYM                                     LT  21
      JST=(IXBLK1-1)*NPSYM                                               LT  22
      DO 4 IC=1,NRH                                                      LT  23
      J=JST                                                              LT  24
      DO 3 K=1,K2                                                        LT  25
      JM1=J                                                              LT  26
      J=J+1                                                              LT  27
      SUM=(0.,0.)                                                        LT  28
      IF (JM1.LT.1) GO TO 2                                              LT  29
      DO 1 I=1,JM1                                                       LT  30
1     SUM=SUM+A(I,K)*B(I,IC)                                             LT  31
2     B(J,IC)=(B(J,IC)-SUM)/A(J,K)                                       LT  32
3     CONTINUE                                                           LT  33
4     CONTINUE                                                           LT  34
C                                                                       LT  35
C     BACKWARD SUBSTITUTION                                              LT  36
C                                                                       LT  37
      JST=NROW+1                                                         LT  38
      DO 8 IXBLK1=1,NBLSYM                                               LT  39
      CALL BLCKIN (A,IFL2,1,I2,1,122)                                    LT  40
      K2=NPSYM                                                           LT  41
      IF (IXBLK1.EQ.1) K2=NLSYM                                          LT  42
      DO 7 IC=1,NRH                                                      LT  43
      KP=K2+1                                                            LT  44
      J=JST                                                              LT  45
      DO 6 K=1,K2                                                        LT  46
      KP=KP-1                                                            LT  47
      JP1=J                                                              LT  48
      J=J-1                                                              LT  49
      SUM=(0.,0.)                                                        LT  50
      IF (NROW.LT.JP1) GO TO 6                                           LT  51
      DO 5 I=JP1,NROW                                                    LT  52
5     SUM=SUM+A(I,KP)*B(I,IC)                                            LT  53
      B(J,IC)=B(J,IC)-SUM                                                LT  54
6     CONTINUE                                                           LT  55
7     CONTINUE                                                           LT  56
8     JST=JST-K2                                                         LT  57
C                                                                       LT  58
C     UNSCRAMBLE SOLUTION                                                LT  59
C                                                                       LT  60
      DO 10 IC=1,NRH                                                     LT  61
      DO 9 I=1,NROW                                                      LT  62
      IXI=IX(I)                                                          LT  63
9     Y(IXI)=B(I,IC)                                                     LT  64
```

```
      DO 10 I=1,NROW
   10 B(I,IC)=Y(I)
      RETURN
      END
```

LUNSCR

PURPOSE

To unscramble the lower triangular matrix of the factored out-of-core matrix and to determine the appropriate ordering of the unknowns. The unscrambled factored matrix is written in blocks on file IU3 in ascending order and on file IU4 in descending order.

METHOD

During factorization by LFACTR, the elements in the lower triangular matrix L were not explicitly arranged in accordance with the row interchanges used in positioning for size during the calculations. Specifically, as the factorization proceeds by columns from left to right in the matrix, row rearrangements in the r^{th} column are not explicitly performed in the left $r - 1$ columns; rather, positioning information is stored in the IP array. For the in-core calculations, these rearrangements are included during the final solution (subroutine SOLVE). For the out-of-core case, rearrangement during the solution (subroutine LTSOLV) is inconvenient, since the transposed system $x^r A^t = B^r$ is being solved, where r signifies a row vector.

The procedure for unscrambling the L matrix is as follows. p_k is the positioning information contained in IP(K). Then for the r^{th} column, let t be a temporary variable:

$$t = \ell_{k,r}$$

$$\ell_{p_k,r} \text{ overwrites } \ell_{k,r}$$

$$t \text{ overwrites } \ell_{p_k,r} \qquad \text{for } k = r + 1, \ldots, n - 1$$

Since row interchanges were used on the transposed matrix, the positions of the unknowns in the equations have changed. The final arrangement is determined by performing interchanges on a vector of integers. Specifically, let

$$x_i = i \qquad i = 1, \ldots, n$$

then set

$$t = x_k$$

x_{p_k} overwrites x_k

t overwrites x_{p_k} for $k = 1, \ldots, n$

The integer now contained in x_i specifies the original placement of the i^{th} unknown.

SYMBOL DICTIONARY

A	=	array for matrix blocks
I1	=	first word of matrix block
I2	=	last word of matrix block
IP	=	array of pivot index data
IU2	=	input file
IU3	=	output file, blocks in normal order
IU4	=	output file, blocks in reversed order
IX	=	array x_i
IXBLK1	=	block number
KA	=	increment to locate the KK^{th} submatrix in case of symmetry
NOP	=	number of symmetric sections
NROW	=	row dimension of A

```
      SUBROUTINE LUNSCR (A,NROW,NOP,IX,IP,IU2,IU3,IU4)
C
C     S/R WHICH UNSCRAMBLES, SCRAMBLED FACTORED MATRIX
C
      COMPLEX A,TEMP
      COMMON /MATPAR/ ICASE,NBLOKS,NPBLK,NLAST,NBLSYM,NPSYM,NLSYM,IMAT,I
     1CASX,NBBX,NPBX,NLBX,NBBL,NPBL,NLBL
      DIMENSION A(NROW,1), IP(NROW), IX(NROW)
      I1=1
      I2=2*NPSYM*NROW
      NM1=NROW-1
      REWIND IU2
      REWIND IU3
      REWIND IU4
      DO 9 KK=1,NOP
      KA=(KK-1)*NROW
      DO 4 IXBLK1=1,NBLSYM
      CALL BLCKIN (A,IU2,I1,I2,1,121)
      K1=(IXBLK1-1)*NPSYM+2
      IF (NM1.LT.K1) GO TO 3
      J2=0
      DO 2 K=K1,NM1
      IF (J2.LT.NPSYM) J2=J2+1
      IPK=IP(K+KA)
      DO 1 J=1,J2
      TEMP=A(K,J)
      A(K,J)=A(IPK,J)
      A(IPK,J)=TEMP
1     CONTINUE
2     CONTINUE
3     CONTINUE
      CALL BLCKOT (A,IU3,I1,I2,1,122)
4     CONTINUE
      DO 5 IXBLK1=1,NBLSYM
      BACKSPACE IU3
      IF (IXBLK1.NE.1) BACKSPACE IU3
      CALL BLCKIN (A,IU3,I1,I2,1,123)
      CALL BLCKOT (A,IU4,I1,I2,1,124)
5     CONTINUE
      DO 6 I=1,NROW
      IX(I+KA)=I
6     CONTINUE
      DO 7 I=1,NROW
      IPI=IP(I+KA)
      IXT=IX(I+KA)
      IX(I+KA)=IX(IPI+KA)
      IX(IPI+KA)=IXT
7     CONTINUE
      IF (NOP.EQ.1) GO TO 9
      NB1=NBLSYM-1
C     SKIP NB1 LOGICAL RECORDS FORWARD
      DO 8 IXBLK1=1,NB1
      CALL BLCKIN (A,IU3,I1,I2,1,125)
8     CONTINUE
9     CONTINUE
      REWIND IU2
      REWIND IU3
      REWIND IU4
      RETURN
      END
```

MOVE

PURPOSE

To rotate and translate a previously defined structure, either moving original segments and patches or leaving the original fixed and producing new segments and patches.

METHOD

The formal parameters ROX, ROY, ROZ are the angles of rotation about the x, y, and z axes, respectively, and XS, YS, ZS are the translation distances in the x, y, and z directions. Angles are in radians, and a positive angle represents a right-hand rotation. The structure is first rotated about the x axis by ROX, then about the y axis by ROY, then about the z axis by ROZ, and finally translated by XS, YS, ZS. These operations transform a point with coordinates x, y, z to x', y', z', where

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{22} & T_{23} \\ T_{31} & T_{32} & T_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} x_s \\ y_s \\ z_s \end{pmatrix}$$

where

$$T_{11} = \cos\phi \cos\theta$$
$$T_{12} = \cos\phi \sin\theta \sin\psi - \sin\phi \cos\psi$$
$$T_{13} = \cos\phi \sin\theta \cos\psi + \sin\phi \sin\psi$$
$$T_{21} = \sin\phi \cos\theta$$
$$T_{22} = \sin\phi \sin\theta \sin\psi + \cos\phi \cos\psi$$
$$T_{23} = \sin\phi \sin\theta \cos\psi - \cos\phi \sin\psi$$
$$T_{31} = -\sin\theta$$
$$T_{32} = \cos\theta \sin\psi$$
$$T_{33} = \cos\theta \cos\psi$$

with

$$\psi = \text{ROX}$$
$$\theta = \text{ROY}$$
$$\phi = \text{ROZ}$$
$$X_s = \text{XS}$$
$$Y_s = \text{YS}$$
$$Z_s = \text{ZS}$$

This transformation is applied to those wire segments from segment number i_s to the last defined segment in COMMON/DATA/. Thus, if i_s is greater than 1, the segments from 1 to $i_s - 1$ are unaffected. All patches are transformed.

NRPT is the structure repetition factor. If NRPT is zero, the transformed segment and patch coordinates overwrite the original coordinates so that the structure is moved with nothing left in the original location. If NRPT is greater than zero, the transformed coordinates are written on the ends of the arrays in COMMON/DATA/ and the process repeated NRPT times so that NRPT new structures are formed, each shifted from the previous one by the specified transformation, while the original structure is unchanged.

CODING

MO18	Adjust symmetry flag if structure is rotated about the x or y axis. If the ground plane flag is also set on the GE card, symmetry will not be used in the solution.
MO19 - MO33	Compute transformation matrix.
MO37 - MO61	Transform segment coordinates.
MO63 - MO93	Transform patch coordinates.
MO94 - MO97	Set parameters to no-symmetry condition if NRPT > 0 or IX > 1.

SYMBOL DICTIONARY

ABS	= external routine (absolute value)
COS	= external routine (cosine)
CPH	= cos ϕ
CPS	= cos Ψ
CTH	= cos θ
IR	= DO loop index, array index for original patch
ISEGNO	= external routine (searches segment tag numbers)
ITGI	= increment applied to segment tag numbers as segments are transformed
ITS	= i_s is the first occurring segment in COMMON/DATA/ with tag ITS
IX	= i_s
I1	= lower DO loop limit for I (initially I1 = i_s)
K	= increment to segment number for transformed segment
KR	= array index for new patch

LDI = LD + 1

NRP = upper DO loop limit for IR

NRPT = repetition factor

ROX = Ψ (radians)

ROY = θ

ROZ = ϕ

SIN = external routine (sine)

SPH = sin ϕ

SPS = sin Ψ

STH = sin θ

T1X, T1Y, T1Z = arrays containing components of $\hat{t}_1$ for patches

T2X, T2Y, T2Z = arrays containing components of $\hat{t}_2$ for patches

XI = old x coordinate

XS = x_s

XX = T_{11}

XY = T_{12}

XZ = T_{13}

X2(I) = x coordinate of end 2 of segment I

YI = old y coordinate

YS = y_s

YX = T_{21}

YY = T_{22}

YZ = T_{23}

Y2(I) = y coordinate of end 2 of segment I

ZI = old Z coordinate

ZS = Z_s

ZX = T_{31}

ZY = T_{32}

ZZ = T_{33}

Z2(I) = Z coordinate of end 2 of segment I

```
      SUBROUTINE MOVE (ROX,ROY,ROZ,XS,YS,ZS,ITS,NRPT,ITGI)
C
C     SUBROUTINE MOVE MOVES THE STRUCTURE WITH RESPECT TO ITS
C     COORDINATE SYSTEM OR REPRODUCES STRUCTURE IN NEW POSITIONS.
C     STRUCTURE IS ROTATED ABOUT X,Y,Z AXES BY ROX,ROY,ROZ
C     RESPECTIVELY, THEN SHIFTED BY XS,YS,ZS
C
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /ANGL/ SALP(300)
      DIMENSION T1X(1), T1Y(1), T1Z(1), T2X(1), T2Y(1), T2Z(1), X2(1), Y
     12(1), Z2(1)
      EQUIVALENCE (X2(1),SI(1)), (Y2(1),ALP(1)), (Z2(1),BET(1))
      EQUIVALENCE (T1X,SI), (T1Y,ALP), (T1Z,BET), (T2X,ICON1), (T2Y,ICON
     12), (T2Z,ITAG)
      IF (ABS(ROX)+ABS(ROY).GT.1.E-10) IPSYM=IPSYM*3
      SPS=SIN(ROX)
      CPS=COS(ROX)
      STH=SIN(ROY)
      CTH=COS(ROY)
      SPH=SIN(ROZ)
      CPH=COS(ROZ)
      XX=CPH*CTH
      XY=CPH*STH*SPS-SPH*CPS
      XZ=CPH*STH*CPS+SPH*SPS
      YX=SPH*CTH
      YY=SPH*STH*SPS+CPH*CPS
      YZ=SPH*STH*CPS-CPH*SPS
      ZX=-STH
      ZY=CTH*SPS
      ZZ=CTH*CPS
      NRP=NRPT
      IF (NRPT.EQ.0) NRP=1
      IF (N.LT.N2) GO TO 3
      I1=ISEGNO(ITS,1)
      IF (I1.LT.N2) I1=N2
      IX=I1
      K=N
      IF (NRPT.EQ.0) K=I1-1
      DO 2 IR=1,NRP
      DO 1 I=I1,N
      K=K+1
      XI=X(I)
      YI=Y(I)
      ZI=Z(I)
      X(K)=XI*XX+YI*XY+ZI*XZ+XS
      Y(K)=XI*YX+YI*YY+ZI*YZ+YS
      Z(K)=XI*ZX+YI*ZY+ZI*ZZ+ZS
      XI=X2(I)
      YI=Y2(I)
      ZI=Z2(I)
      X2(K)=XI*XX+YI*XY+ZI*XZ+XS
      Y2(K)=XI*YX+YI*YY+ZI*YZ+YS
      Z2(K)=XI*ZX+YI*ZY+ZI*ZZ+ZS
      BI(K)=BI(I)
      ITAG(K)=ITAG(I)+ITGI
1     CONTINUE
      I1=N+1
      N=K
2     CONTINUE
3     IF (M.LT.M2) GO TO 6
      I1=M2
      K=M
```

```
      LDI=LD+1
      IF (NRPT.EQ.0) K=M1
      DO 5 II=1,NRP
      DO 4 I=I1,M
      K=K+1
      IR=LDI-I
      KR=LDI-K
      XI=X(IR)
      YI=Y(IR)
      ZI=Z(IR)
      X(KR)=XI*XX+YI*XY+ZI*XZ+XS
      Y(KR)=XI*YX+YI*YY+ZI*YZ+YS
      Z(KR)=XI*ZX+YI*ZY+ZI*ZZ+ZS
      XI=T1X(IR)
      YI=T1Y(IR)
      ZI=T1Z(IR)
      T1X(KR)=XI*XX+YI*XY+ZI*XZ
      T1Y(KR)=XI*YX+YI*YY+ZI*YZ
      T1Z(KR)=XI*ZX+YI*ZY+ZI*ZZ
      XI=T2X(IR)
      YI=T2Y(IR)
      ZI=T2Z(IR)
      T2X(KR)=XI*XX+YI*XY+ZI*XZ
      T2Y(KR)=XI*YX+YI*YY+ZI*YZ
      T2Z(KR)=XI*ZX+YI*ZY+ZI*ZZ
      SALP(KR)=SALP(IR)
4     BI(KR)=BI(IR)
      I1=M+1
5     M=K
6     IF ((NRPT.EQ.0).AND.(IX.EQ.1)) RETURN
      NP=N
      MP=M
      IPSYM=0
      RETURN
      END
```

NEFLD

PURPOSE

To compute the near electric field due to currents induced on a structure.

CODING

NE30 - NE93	Near E field due to currents on segments is computed.
NE30 - NE41	Each segment is checked to determine whether the field observation point (XOB, YOB, ZOB) falls within the segment volume. If it does, AX is set to the radius of that segment. AX is then sent to routine EFLD as the radius of the observation segment. If (XOB, YOB, ZOB) is on the axis of a segment at its center, the field calculation with AX set to the segment radius is the same as that used in filling the matrix.
NE42 - NE93	Loop computing the field contribution of each segment.
NE43 - NE50	Parameters of source segment are stored in COMMON/DATAJ/.
NE51 - NE85	When the extended thin wire approximation is used, IND1 is set to 0 if end 1 of segment I is connected to a single parallel segment of the same radius, 1 if it is a free end, and 2 if it connects to a multiple junction, a bend, or a segment of different radius. IND2 is the same for end 2. If IND1 or IND2 is 2, the extended thin wire approximation will not be used for that end.
NE87	EFLD stores the electric fields due to constant, sin ks, and cos ks currents in COMMON/DATAJ/.
NE88 - NE93	The field components are multiplied by the coefficients of the constant, sin ks, and cos ks components of the total segment current, and the field is summed.
NE95 - NE117	Near field due to patch currents is computed.

SYMBOL DICTIONARY

ACX = constant component of segment current at NE88; $\hat{t}_1$ component of patch current at NE110

AX = segment radius when the field evaluation point falls within a segment volume

B = source segment radius

BCX = sin ks component of segment current at NE89; $\hat{t}_2$ component of patch current at NE111

CCX = cos ks component of segment current at NE90

EX, EY, EZ = x, y, and z components of total electric field

EXC, EYC, EZC = E field due to a cos ks current on a segment

EXK, EYK, EZK = E field due to a constant current at NE87; E field due to the $\hat{t}_1$ component of patch current at NE114

EXS, EYX, EZS = E field due to a sin ks current at NE87; E field due to the $\hat{t}_2$ component of patch current at NE114

IP = loop index for direct and reflected field (1, 2, respectively)

T1X, T1Y, T1Z = arrays for $\hat{t}_1$

T1XJ, T1YJ, T1ZJ = $\hat{t}_1$ for source patch

T2X, T2Y, T2Z = arrays for $\hat{t}_2$

T2XJ, T2YJ, T2ZJ = $\hat{t}_2$ for source path

XI = cosine of the angle between segment I and the segment connected to its end

XOB, YOB, ZOB = field evaluation point

ZP = coordinates of the field evaluation point, z or ρ^2, in a cylindrical coordinate system centered on the source segment

CONSTANTS

0.5001 = fraction of segment length used to test whether the field evaluation point falls within a segment

0.9 = fraction of segment radius used to test whether the field evaluation point falls within a segment

0.999999 = minimum XI for extended thin wire kernel (maximum angle = 0.08 degree)

```
      SUBROUTINE NEFLD (XOB,YOB,ZOB,EX,EY,EZ)
C
C     NEFLD COMPUTES THE NEAR FIELD AT SPECIFIED POINTS IN SPACE AFTER
C     THE STRUCTURE CURRENTS HAVE BEEN COMPUTED.
C
      COMPLEX EX,EY,EZ,CUR,ACX,BCX,CCX,EXK,EYK,EZK,EXS,EYS,EZS,EXC,EYC,E
     1ZC,ZRATI,ZRATI2,T1,FRATI
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /ANGL/ SALP(300)
      COMMON /CRNT/ AIR(300),AII(300),BIR(300),BII(300),CIR(300),CII(300
     1),CUR(900)
      COMMON /DATAJ/ S,B,XJ,YJ,ZJ,CABJ,SABJ,SALPJ,EXK,EYK,EZK,EXS,EYS,EZ
     1S,EXC,EYC,EZC,RKH,IEXK,IND1,IND2,IPGND
      COMMON /GND/ZRATI,ZRATI2,FRATI,CL,CH,SCRWL,SCRWR,NRADL,KSYMP,IFAR,
     1IPERF,T1,T2
      DIMENSION CAB(1), SAB(1), T1X(1), T1Y(1), T1Z(1), T2X(1), T2Y(1),
     1T2Z(1)
      EQUIVALENCE (CAB,ALP), (SAB,BET)
      EQUIVALENCE (T1X,SI), (T1Y,ALP), (T1Z,BET), (T2X,ICON1), (T2Y,ICON
     12), (T2Z,ITAG)
      EQUIVALENCE (T1XJ,CABJ), (T1YJ,SABJ), (T1ZJ,SALPJ), (T2XJ,B), (T2Y
     1J,IND1), (T2ZJ,IND2)
      EX=(0.,0.)
      EY=(0.,0.)
      EZ=(0.,0.)
      AX=0.
      IF (N.EQ.0) GO TO 20
      DO 1 I=1,N
      XJ=XOB-X(I)
      YJ=YOB-Y(I)
      ZJ=ZOB-Z(I)
      ZP=CAB(I)*XJ+SAB(I)*YJ+SALP(I)*ZJ
      IF (ABS(ZP).GT.0.5001*SI(I)) GO TO 1
      ZP=XJ*XJ+YJ*YJ+ZJ*ZJ-ZP*ZP
      XJ=BI(I)
      IF (ZP.GT.0.9*XJ*XJ) GO TO 1
      AX=XJ
      GO TO 2
1     CONTINUE
2     DO 19 I=1,N
      S=SI(I)
      B=BI(I)
      XJ=X(I)
      YJ=Y(I)
      ZJ=Z(I)
      CABJ=CAB(I)
      SABJ=SAB(I)
      SALPJ=SALP(I)
      IF (IEXK.EQ.0) GO TO 18
      IPR=ICON1(I)
      IF (IPR) 3,8,4
3     IPR=-IPR
      IF (-ICON1(IPR).NE.I) GO TO 9
      GO TO 6
4     IF (IPR.NE.I) GO TO 5
      IF (CABJ*CABJ+SABJ*SABJ.GT.1.E-8) GO TO 9
      GO TO 7
5     IF (ICON2(IPR).NE.I) GO TO 9
6     XI=ABS(CABJ*CAB(IPR)+SABJ*SAB(IPR)+SALPJ*SALP(IPR))
      IF (XI.LT.0.999999) GO TO 9
      IF (ABS(BI(IPR)/B-1.).GT.1.E-6) GO TO 9
7     IND1=0
```

```
      GO TO 10
    8 IND1=1
      GO TO 10
    9 IND1=2
   10 IPR=ICON2(I)
      IF (IPR) 11,16,12
   11 IPR=-IPR
      IF (-ICON2(IPR).NE.I) GO TO 17
      GO TO 14
   12 IF (IPR.NE.I) GO TO 13
      IF (CABJ*CABJ+SABJ*SABJ.GT.1.E-8) GO TO 17
      GO TO 15
   13 IF (ICON1(IPR).NE.I) GO TO 17
   14 XI=ABS(CABJ*CAB(IPR)+SABJ*SAB(IPR)+SALPJ*SALP(IPR))
      IF (XI.LT.0.999999) GO TO 17
      IF (ABS(BI(IPR)/B-1.).GT.1.E-6) GO TO 17
   15 IND2=0
      GO TO 18
   16 IND2=1
      GO TO 18
   17 IND2=2
   18 CONTINUE
      CALL EFLD (XOB,YOB,ZOB,AX,1)
      ACX=CMPLX(AIR(I),AII(I))
      BCX=CMPLX(BIR(I),BII(I))
      CCX=CMPLX(CIR(I),CII(I))
      EX=EX+EXK*ACX+EXS*BCX+EXC*CCX
      EY=EY+EYK*ACX+EYS*BCX+EYC*CCX
   19 EZ=EZ+EZK*ACX+EZS*BCX+EZC*CCX
      IF (M.EQ.0) RETURN
   20 JC=N
      JL=LD+1
      DO 21 I=1,M
      JL=JL-1
      S=BI(JL)
      XJ=X(JL)
      YJ=Y(JL)
      ZJ=Z(JL)
      T1XJ=T1X(JL)
      T1YJ=T1Y(JL)
      T1ZJ=T1Z(JL)
      T2XJ=T2X(JL)
      T2YJ=T2Y(JL)
      T2ZJ=T2Z(JL)
      JC=JC+3
      ACX=T1XJ*CUR(JC-2)+T1YJ*CUR(JC-1)+T1ZJ*CUR(JC)
      BCX=T2XJ*CUR(JC-2)+T2YJ*CUR(JC-1)+T2ZJ*CUR(JC)
      DO 21 IP=1,KSYMP
      IPGND=IP
      CALL UNERE (XOB,YOB,ZOB)
      EX=EX+ACX*EXK+BCX*EXS
      EY=EY+ACX*EYK+BCX*EYS
   21 EZ=EZ+ACX*EZK+BCX*EZS
      RETURN
      END
```

NETWK

PURPOSE

To solve for the voltages and currents at the ports of non-radiating networks that are part of the antenna. This routine also is involved in the solution for current when there are no non-radiating networks, and computes the relative driving point matrix asymmetry when this option is requested.

METHOD

Driving Point Matrix Asymmetry (NT32 to NT84):

To satisfy physical reciprocity, the elements of the inverse of the interaction matrix should satisfy the condition

$$G^{-1}_{ij}/\Delta_j = G^{-1}_{ji}/\Delta_i \quad i, j = 1, \ldots, n ,$$

where Δ_i = length of segment i. This condition is not satisfied exactly, except on special structures, since the terms computed are not true reactions. The relative asymmetry of a matrix element is defined as

$$A = \left| \frac{\left(G^{-1}_{ij}/\Delta_j - G^{-1}_{ji}/\Delta_i\right)}{(G^{-1}_{ij}/\Delta_j)} \right| .$$

The code from NT32 to NT84 computes the relative asymmetries of matrix elements for i and j of all driving point segments: either voltage source driving points or network connection points. The maximum relative asymmetry is located, and the rms relative asymmetry of all elements used is computed.

LOCAL CODING STRUCTURE

NT32 - NT44 Determine numbers of segments that are network connection points.

NT46 - NT54 Determine numbers of segments that are voltage source driving points. Indices of segments with network connections or voltage sources are stored in array IPNT with no duplication of numbers.

NT59 - NT69 Compute $G^{-1}_{k\ell}/\Delta_\ell$ for k,ℓ = all segment numbers in IPNT.

NT70 - NT84 Compute relative asymmetries of elements computed above, search for maximum and compute rms asymmetry.

LOCAL SYMBOL DICTIONARY

ASA = sum of squares of relative asymmetries and rms value

ASM = Δ_{ISC1} before NT70; maximum relative asymmetry after NT69

CMN(J,I) = $G^{-1}_{k\ell}/\Delta_\ell$; k = IPNT(J), ℓ = IPNT(I)

CUR = temporary storage of $G^{-1}_{\ell k}/\Delta_k$

IPNT = array of driving point segment indices

IROW1 = number of entries in IPNT

ISC1 = temporary storage of segment index

MASYM = flag; if non-zero, matrix asymmetry is computed

NTEQ = row index of element having maximum asymmetry

NTSC = column index of element having maximum asymmetry

PWR = relative matrix asymmetry

RHS = vector for matrix solution used in obtaining $G^{-1}_{k\ell}$

Non-radiating Network Solution (NT89 to NT262):

The solution method when non-radiating networks are present is discussed in Part I.

Data for non-radiating networks is passed through the COMMON/NETCX/ where

ISEG1(I) = number of the segment to which end 1 of I^{th} two-port network is connected

ISEG2(I) = number of segment to which end 2 of I^{th} two-port network is connected

NONET = number of two-port networks for which data is given

Network parameters are contained in the arrays X11R, X11I, X12R, X12I, X22R, and X22I, and the type of network is determined by NTYP:

If NTYP is 1 -- the network parameters are the short-circuit admittance parameters of the network:

X11R, X11I = real and imaginary parts of Y_{11}

X12R, X12I = real and imaginary parts of $Y_{12} = Y_{21}$

X22R, X22I = real and imaginary parts of Y_{22}

If NTYP is 2 or 3 -- the network is a transmission line:

X11R = characteristic impedance of transmission line

X11I = length of transmission line in meters

X12R = real part of shunt admittance on end 1 of line

X12I = imaginary part of shunt admittance on end 1 of line

X22R = real part of shunt admittance on end 2 of line

X22I = imaginary part of shunt admittance on end 2 of line

If NTYP is 2 -- the transmission line runs straight between the segments with respect to the segment reference directions.

If NTYP is 3 -- the transmission line is twisted as shown in figure 8.

The short circuit admittance parameters of the transmission line, Y_{11}, Y_{12}, and Y_{22}, are computed from NT110 to NT120 in the code. When NTYP is 3, the sign of Y_{12} is reversed.

The code from NT99 to NT194 forms a loop that for each network: computes the network parameters Y_{11}, Y_{12} and Y_{22}; sorts the segment indices involved; and adds the parameters Y_{11}, Y_{12}, and Y_{22} to the appropriate network equations. The sorting procedure for the connection of end 1 of the network is described in figure 9. Decision 1 is made in the code from NT121 to NT126, decision 2 from NT128 to NT133, and decision 3 from NT138 to NT143. Segments having network connections only are assigned equation rows in the array CMN starting from the top in the order that the segments are encountered. Segments with both network and voltage source connections are assigned equation rows in CMN starting at the bottom and proceeding up. The former are eventually solved for the unknown gap voltages, while the latter are used to obtain source input admittances after the structure currents have been computed. The code from NT148 to NT174 assigns equation numbers for the connection of end 2 of the networks and sets IROW2 and ISC2.

The network short circuit parameters are added to the network equations from NT182 to NT193. The coefficient matrix is transposed in filling the CMN array, since the matrix solution routines operate on a transposed system. Hence, the first index should be considered the column number and the second index the row number. If a segment NSEG1 does not have a voltage source connected, the parameters Y_{11} and Y_{12} are added to column IROW1 at rows IROW1 and IROW2, respectively. IROW2 may be either (1) in the upper rows as part of the equations for the unknown gap voltages, or (2) if a voltage source is connected to segment NSEG2, in the lower rows for later determination of the source current. If a voltage source is connected to segment NSEG1, the

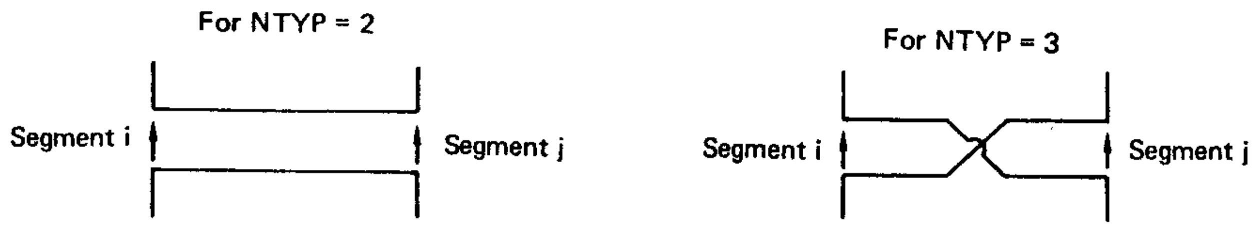


Figure 8. Options for Transmission Line Connection.

coefficients Y_{11} and Y_{12} are multiplied by the known source voltage and added to the right-hand side of the network equation in the rows IROW1 and IROW2. The parameters Y_{12} and Y_{22} are added to the equations in a similar manner.

The loop from NT199 to NT208 computes the elements of the inverse matrix G_{mn}^{-1} and adds them to the network equations. The network matrix is then factored at NT213. The code from NT218 to NT225 computes B_i = RHS(I), where

$$B_i = \sum_{j=1}^{N} G_{ij}^{-1} E'_j \qquad i = 1, \ldots N ,$$

with $(-E'_j)$ being the known applied field on segment j, not including unknown voltage drops at network ports. Those elements B_i for segments in the network equations are then added to the right-hand side of the network equations. At NT229 the network equations are solved for the excitation fields due to voltage drops at the network ports. The negatives of these fields are added to the excitation vector at NT234 to NT236, completing the definition of the excitation vector E_j. The structure equations are then solved for the induced currents.

$$I_j = \sum_{j=1}^{N} G_{ij}^{-1} E_j .$$

From NT241 to NT261, the voltage, current, admittance, and power seen looking into the structure at each network port are printed. This current does not include current through any voltage sources that are connected to the port.

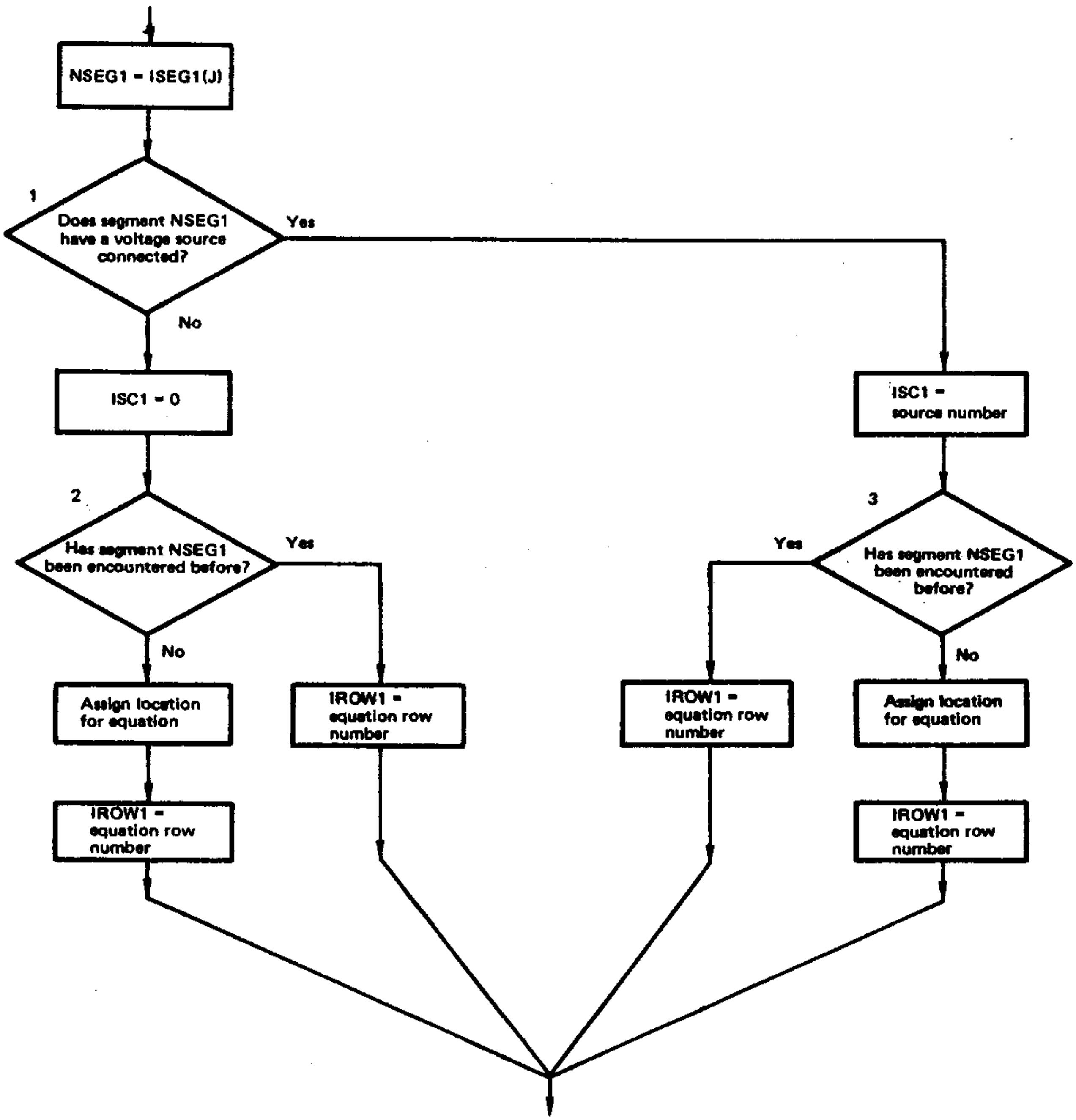


Figure 9. Sorting Procedure for Segments Having Network Connections

The code from NT269 to NT294 computes and prints the voltage, current, admittance, and power seen by each voltage source looking into the structure and parallel connected network port, if a network is present.

After the network equations have once been set up, they can be solved for various incident fields by entering the code at NT218. If the location of voltage sources is changed, however, the equations must be recomputed.

If a structure has no non-radiating networks, the currents are computed at NT266.

SYMBOL DICTIONARY

ASA	= sum of squares of relative matrix asymmetries and rms value
ASM	= segment length and maximum relative matrix asymmetry
CABS	= external routine (magnitude of complex number)
CM	= array of matrix elements G_{ij}
CMN	= array for network equation coefficients
CMPLX	= external routine (forms complex number)
CONJG	= external routine (conjugate)
COS	= external routine (cosine)
CUR	= current
EINC	= excitation vector
FACTR	= external routine (Gauss-Doolittle matrix factoring)
FLOAT	= external routine (integer to real conversion)
I	= DO loop index
IP	= array of positioning data from factoring of CM
IPNT	= array of positioning data from factoring of CMN
IROW1	= matrix element index
IROW2	= matrix element index
ISANT	= array of segment numbers for voltage source connection
ISC1	= segment location in array ISANT
ISC2	= segment location in array ISANT
ISEG1	= number of segment to which port 1 of network is connected
ISEG2	= number of segment to which port 2 is connected
IX	= array of positioning data from factoring of CM
J	= DO loop index
MASYM	= flag to request matrix asymmetry calculation
NCOL	= number of columns in CM
NDIMN	= array dimension of CMN

NDIMNP = NDIMN + 1
NONET = number of networks
NOP = N/NP
NPRINT = flag to control printing
NROW = number of rows in CM
NSANT = number of voltage sources
NSEG1 = array of segments to which port 1 of a network connects
NSEG2 = array of segments to which port 2 of a network connects
NTEQA(I) = segment number associated with I^{th} network equation
NTSC = number of network-voltage source equations
NTSCA(I) = segment number associated with I^{th} network-voltage source equation
NTSOL = flag to indicate network equations do not need to be recomputed
NTYP(I) = type of I^{th} network
PIN = total input power from sources
PNLS = power lost in networks
PWR = power
REAL = external routine (real part of complex number)
RHNT = vector for right-hand side of network equations
RHNX = component of RHNT due to Y_{11}, Y_{12}, Y_{22} terms
RHS = vector for right-hand side of structure interaction equation
SIN = external routine (sine)
SOLVE = external routine (Gauss-Doolittle solution)
SOLVES = external routine (Gauss-Doolittle solution of CM matrix)
SQRT = external routine (square root)
TP = 2π
VLT = voltage
VSANT(I) = voltage of source on segment NSANT(I)
VSRC(I) = voltage of source on I^{th} segment in network-voltage source equations
X11I, X11R, X12I, X12R, X22I, X22R = network or transmission line specification parameters

YMIT = admittance

Y11I = imaginary part of Y_{11}

Y11R = real part of Y_{11}

Y12I = imaginary part of Y_{12}

Y12R = real part of Y_{12}

Y22I = imaginary part of Y_{22}

Y22R = real part of Y_{22}

ZPED = impedance

CONSTANTS

6.283185308 = 2π

30 = row and column dimensions of CMN

31 = (row and column dimensions of CMN) + 1

```
      SUBROUTINE NETWK (CM,CMB,CMC,CMD,IP,EINC)                         NT   1
C                                                                       NT   2
C     SUBROUTINE NETWK SOLVES FOR STRUCTURE CURRENTS FOR A GIVEN        NT   3
C     EXCITATION INCLUDING THE EFFECT OF NON-RADIATING NETWORKS IF      NT   4
C     PRESENT.                                                          NT   5
C                                                                       NT   6
      COMPLEX CMN,RHNT,YMIT,RHS,ZPED,EINC,VSANT,VLT,CUR,VSRC,RHNX,VQD,VQ NT   7
     1DS,CUX,CM,CMB,CMC,CMD                                             NT   8
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300 NT   9
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX( NT  10
     2300),WLAM,IPSYM                                                   NT  11
      COMMON /CRNT/ AIR(300),AII(300),BIR(300),BII(300),CIR(300),CII(300 NT  12
     1),CUR(900)                                                        NT  13
      COMMON /VSORC/ VQD(30),VSANT(30),VQDS(30),IVQD(30),ISANT(30),IQDS( NT  14
     130),NVQD,NSANT,NQDS                                               NT  15
      COMMON /NETCX/ ZPED,PIN,PNLS,NEQ,NPEQ,NEQ2,NONET,NTSOL,NPRINT,MASY NT  16
     1M,ISEG1(30),ISEG2(30),X11R(30),X11I(30),X12R(30),X12I(30),X22R(30) NT  17
     2,X22I(30),NTYP(30)                                                NT  18
      DIMENSION EINC(1), IP(1)                                          NT  19
      DIMENSION CMN(30,30), RHNT(30), IPNT(30), NTEQA(30), NTSCA(30), RH NT  20
     1S(900), VSRC(10), RHNX(30)                                        NT  21
      DATA NDIMN,NDIMNP/30,31/,TP/6.283185308/                          NT  22
      PIN=0.                                                            NT  23
      PNLS=0.                                                           NT  24
      NEQT=NEQ+NEQ2                                                     NT  25
      IF (NTSOL.NE.0) GO TO 42                                          NT  26
      NOP=NEQ/NPEQ                                                      NT  27
      IF (MASYM.EQ.0) GO TO 14                                          NT  28
C                                                                       NT  29
C     COMPUTE RELATIVE MATRIX ASYMMETRY                                 NT  30
C                                                                       NT  31
      IROW1=0                                                           NT  32
      IF (NONET.EQ.0) GO TO 5                                           NT  33
      DO 4 I=1,NONET                                                    NT  34
      NSEG1=ISEG1(I)                                                    NT  35
      DO 3 ISC1=1,2                                                     NT  36
      IF (IROW1.EQ.0) GO TO 2                                           NT  37
      DO 1 J=1,IROW1                                                    NT  38
      IF (NSEG1.EQ.IPNT(J)) GO TO 3                                     NT  39
1     CONTINUE                                                          NT  40
2     IROW1=IROW1+1                                                     NT  41
      IPNT(IROW1)=NSEG1                                                 NT  42
3     NSEG1=ISEG2(I)                                                    NT  43
4     CONTINUE                                                          NT  44
5     IF (NSANT.EQ.0) GO TO 9                                           NT  45
      DO 8 I=1,NSANT                                                    NT  46
      NSEG1=ISANT(I)                                                    NT  47
      IF (IROW1.EQ.0) GO TO 7                                           NT  48
      DO 6 J=1,IROW1                                                    NT  49
      IF (NSEG1.EQ.IPNT(J)) GO TO 8                                     NT  50
6     CONTINUE                                                          NT  51
7     IROW1=IROW1+1                                                     NT  52
      IPNT(IROW1)=NSEG1                                                 NT  53
8     CONTINUE                                                          NT  54
9     IF (IROW1.LT.NDIMNP) GO TO 10                                     NT  55
      PRINT 59                                                          NT  56
      STOP                                                              NT  57
10    IF (IROW1.LT.2) GO TO 14                                          NT  58
      DO 12 I=1,IROW1                                                   NT  59
      ISC1=IPNT(I)                                                      NT  60
      ASM=SI(ISC1)                                                      NT  61
      DO 11 J=1,NEQT                                                    NT  62
11    RHS(J)=(0.,0.)                                                    NT  63
      RHS(ISC1)=(1.,0.)                                                 NT  64
```

```
      CALL SOLGF (CM,CMB,CMC,CMD,RHS,IP,NP,N1,N,MP,M1,M,NEQ,NEQ2)          NT  65
      CALL CABC (RHS)                                                      NT  66
      DO 12 J=1,IROW1                                                      NT  67
      ISC1=IPNT(J)                                                         NT  68
12    CMN(J,I)=RHS(ISC1)/ASM                                               NT  69
      ASM=0.                                                               NT  70
      ASA=0.                                                               NT  71
      DO 13 I=2,IROW1                                                      NT  72
      ISC1=I-1                                                             NT  73
      DO 13 J=1,ISC1                                                       NT  74
      CUX=CMN(I,J)                                                         NT  75
      PWR=CABS((CUX-CMN(J,I))/CUX)                                         NT  76
      ASA=ASA+PWR*PWR                                                      NT  77
      IF (PWR.LT.ASM) GO TO 13                                             NT  78
      ASM=PWR                                                              NT  79
      NTEQ=IPNT(I)                                                         NT  80
      NTSC=IPNT(J)                                                         NT  81
13    CONTINUE                                                             NT  82
      ASA=SQRT(ASA*2./FLOAT(IROW1*(IROW1-1)))                              NT  83
      PRINT 58, ASM,NTEQ,NTSC,ASA                                          NT  84
14    IF (NONET.EQ.0) GO TO 48                                             NT  85
C                                                                          NT  86
C     SOLUTION OF NETWORK EQUATIONS                                        NT  87
C                                                                          NT  88
      DO 15 I=1,NDIMN                                                      NT  89
      RHNX(I)=(0.,0.)                                                      NT  90
      DO 15 J=1,NDIMN                                                      NT  91
15    CMN(I,J)=(0.,0.)                                                     NT  92
      NTEQ=0                                                               NT  93
      NTSC=0                                                               NT  94
C                                                                          NT  95
C     SORT NETWORK AND SOURCE DATA AND ASSIGN EQUATION NUMBERS TO          NT  96
C     SEGMENTS.                                                            NT  97
C                                                                          NT  98
      DO 38 J=1,NONET                                                      NT  99
      NSEG1=ISEG1(J)                                                       NT 100
      NSEG2=ISEG2(J)                                                       NT 101
      IF (NTYP(J).GT.1) GO TO 16                                           NT 102
      Y11R=X11R(J)                                                         NT 103
      Y11I=X11I(J)                                                         NT 104
      Y12R=X12R(J)                                                         NT 105
      Y12I=X12I(J)                                                         NT 106
      Y22R=X22R(J)                                                         NT 107
      Y22I=X22I(J)                                                         NT 108
      GO TO 17                                                             NT 109
16    Y22R=TP*X11I(J)/WLAM                                                 NT 110
      Y12R=0.                                                              NT 111
      Y12I=1./(X11R(J)*SIN(Y22R))                                          NT 112
      Y11R=X12R(J)                                                         NT 113
      Y11I=-Y12I*COS(Y22R)                                                 NT 114
      Y22R=X22R(J)                                                         NT 115
      Y22I=Y11I+X22I(J)                                                    NT 116
      Y11I=Y11I+X12I(J)                                                    NT 117
      IF (NTYP(J).EQ.2) GO TO 17                                           NT 118
      Y12R=-Y12R                                                           NT 119
      Y12I=-Y12I                                                           NT 120
17    IF (NSANT.EQ.0) GO TO 19                                             NT 121
      DO 18 I=1,NSANT                                                      NT 122
      IF (NSEG1.NE.ISANT(I)) GO TO 18                                      NT 123
      ISC1=I                                                               NT 124
      GO TO 22                                                             NT 125
18    CONTINUE                                                             NT 126
19    ISC1=0                                                               NT 127
      IF (NTEQ.EQ.0) GO TO 21                                              NT 128
```

```
      DO 20 I=1,NTEQ
      IF (NSEG1.NE.NTEQA(I)) GO TO 20
      IROW1=I
      GO TO 25
20    CONTINUE
21    NTEQ=NTEQ+1
      IROW1=NTEQ
      NTEQA(NTEQ)=NSEG1
      GO TO 25
22    IF (NTSC.EQ.0) GO TO 24
      DO 23 I=1,NTSC
      IF (NSEG1.NE.NTSCA(I)) GO TO 23
      IROW1=NDIMNP-I
      GO TO 25
23    CONTINUE
24    NTSC=NTSC+1
      IROW1=NDIMNP-NTSC
      NTSCA(NTSC)=NSEG1
      VSRC(NTSC)=VSANT(ISC1)
25    IF (NSANT.EQ.0) GO TO 27
      DO 26 I=1,NSANT
      IF (NSEG2.NE.ISANT(I)) GO TO 26
      ISC2=I
      GO TO 30
26    CONTINUE
27    ISC2=0
      IF (NTEQ.EQ.0) GO TO 29
      DO 28 I=1,NTEQ
      IF (NSEG2.NE.NTEQA(I)) GO TO 28
      IROW2=I
      GO TO 33
28    CONTINUE
29    NTEQ=NTEQ+1
      IROW2=NTEQ
      NTEQA(NTEQ)=NSEG2
      GO TO 33
30    IF (NTSC.EQ.0) GO TO 32
      DO 31 I=1,NTSC
      IF (NSEG2.NE.NTSCA(I)) GO TO 31
      IROW2=NDIMNP-I
      GO TO 33
31    CONTINUE
32    NTSC=NTSC+1
      IROW2=NDIMNP-NTSC
      NTSCA(NTSC)=NSEG2
      VSRC(NTSC)=VSANT(ISC2)
33    IF (NTSC+NTEQ.LT.NDIMNP) GO TO 34
      PRINT 59
      STOP
C
C     FILL NETWORK EQUATION MATRIX AND RIGHT HAND SIDE VECTOR WITH
C     NETWORK SHORT-CIRCUIT ADMITTANCE MATRIX COEFFICIENTS.
C
34    IF (ISC1.NE.0) GO TO 35
      CMN(IROW1,IROW1)=CMN(IROW1,IROW1)-CMPLX(Y11R,Y11I)*SI(NSEG1)
      CMN(IROW1,IROW2)=CMN(IROW1,IROW2)-CMPLX(Y12R,Y12I)*SI(NSEG1)
      GO TO 36
35    RHNX(IROW1)=RHNX(IROW1)+CMPLX(Y11R,Y11I)*VSANT(ISC1)/WLAM
      RHNX(IROW2)=RHNX(IROW2)+CMPLX(Y12R,Y12I)*VSANT(ISC1)/WLAM
36    IF (ISC2.NE.0) GO TO 37
      CMN(IROW2,IROW2)=CMN(IROW2,IROW2)-CMPLX(Y22R,Y22I)*SI(NSEG2)
      CMN(IROW2,IROW1)=CMN(IROW2,IROW1)-CMPLX(Y12R,Y12I)*SI(NSEG2)
      GO TO 38
37    RHNX(IROW1)=RHNX(IROW1)+CMPLX(Y12R,Y12I)*VSANT(ISC2)/WLAM
```

```
      RHNX(IROW2)=RHNX(IROW2)+CMPLX(Y22R,Y22I)*VSANT(ISC2)/WLAM
   38 CONTINUE
C
C     ADD INTERACTION MATRIX ADMITTANCE ELEMENTS TO NETWORK EQUATION
C     MATRIX
C
      DO 41 I=1,NTEQ
      DO 39 J=1,NEQT
   39 RHS(J)=(0.,0.)
      IROW1=NTEQA(I)
      RHS(IROW1)=(1.,0.)
      CALL SOLGF (CM,CMB,CMC,CMD,RHS,IP,NP,N1,N,MP,M1,M,NEQ,NEQ2)
      CALL CABC (RHS)
      DO 40 J=1,NTEQ
      IROW1=NTEQA(J)
   40 CMN(I,J)=CMN(I,J)+RHS(IROW1)
   41 CONTINUE
C
C     FACTOR NETWORK EQUATION MATRIX
C
      CALL FACTR (NTEQ,CMN,IPNT,NDIMN)
C
C     ADD TO NETWORK EQUATION RIGHT HAND SIDE THE TERMS DUE TO ELEMENT
C     INTERACTIONS
C
   42 IF (NONET.EQ.0) GO TO 48
      DO 43 I=1,NEQT
   43 RHS(I)=EINC(I)
      CALL SOLGF (CM,CMB,CMC,CMD,RHS,IP,NP,N1,N,MP,M1,M,NEQ,NEQ2)
      CALL CABC (RHS)
      DO 44 I=1,NTEQ
      IROW1=NTEQA(I)
   44 RHNT(I)=RHNX(I)+RHS(IROW1)
C
C     SOLVE NETWORK EQUATIONS
C
      CALL SOLVE (NTEQ,CMN,IPNT,RHNT,NDIMN)
C
C     ADD FIELDS DUE TO NETWORK VOLTAGES TO ELECTRIC FIELDS APPLIED TO
C     STRUCTURE AND SOLVE FOR INDUCED CURRENT
C
      DO 45 I=1,NTEQ
      IROW1=NTEQA(I)
   45 EINC(IROW1)=EINC(IROW1)-RHNT(I)
      CALL SOLGF (CM,CMB,CMC,CMD,EINC,IP,NP,N1,N,MP,M1,M,NEQ,NEQ2)
      CALL CABC (EINC)
      IF (NPRINT.EQ.0) PRINT 61
      IF (NPRINT.EQ.0) PRINT 60
      DO 46 I=1,NTEQ
      IROW1=NTEQA(I)
      VLT=RHNT(I)*SI(IROW1)*WLAM
      CUX=EINC(IROW1)*WLAM
      YMIT=CUX/VLT
      ZPED=VLT/CUX
      IROW2=ITAG(IROW1)
      PWR=.5*REAL(VLT*CONJG(CUX))
      PNLS=PNLS-PWR
   46 IF (NPRINT.EQ.0) PRINT 62, IROW2,IROW1,VLT,CUX,ZPED,YMIT,PWR
      IF (NTSC.EQ.0) GO TO 49
      DO 47 I=1,NTSC
      IROW1=NTSCA(I)
      VLT=VSRC(I)
      CUX=EINC(IROW1)*WLAM
      YMIT=CUX/VLT
```

```
      ZPED=VLT/CUX
      IROW2=ITAG(IROW1)
      PWR=.5*REAL(VLT*CONJG(CUX))
      PNLS=PNLS-PWR
   47 IF (NPRINT.EQ.0) PRINT 62, IROW2,IROW1,VLT,CUX,ZPED,YMIT,PWR
      GO TO 49
C
C     SOLVE FOR CURRENTS WHEN NO NETWORKS ARE PRESENT
C
   48 CALL SOLGF (CM,CMB,CMC,CMD,EINC,IP,NP,N1,N,MP,M1,M,NEQ,NEQ2)
      CALL CABC (EINC)
      NTSC=0
   49 IF (NSANT+NVQD.EQ.0) RETURN
      PRINT 63
      PRINT 60
      IF (NSANT.EQ.0) GO TO 56
      DO 55 I=1,NSANT
      ISC1=ISANT(I)
      VLT=VSANT(I)
      IF (NTSC.EQ.0) GO TO 51
      DO 50 J=1,NTSC
      IF (NTSCA(J).EQ.ISC1) GO TO 52
   50 CONTINUE
   51 CUX=EINC(ISC1)*WLAM
      IROW1=0
      GO TO 54
   52 IROW1=NDIMNP-J
      CUX=RHNX(IROW1)
      DO 53 J=1,NTEQ
   53 CUX=CUX-CMN(J,IROW1)*RHNT(J)
      CUX=(EINC(ISC1)+CUX)*WLAM
   54 YMIT=CUX/VLT
      ZPED=VLT/CUX
      PWR=.5*REAL(VLT*CONJG(CUX))
      PIN=PIN+PWR
      IF (IROW1.NE.0) PNLS=PNLS+PWR
      IROW2=ITAG(ISC1)
   55 PRINT 62, IROW2,ISC1,VLT,CUX,ZPED,YMIT,PWR
   56 IF (NVQD.EQ.0) RETURN
      DO 57 I=1,NVQD
      ISC1=IVQD(I)
      VLT=VQD(I)
      CUX=CMPLX(AIR(ISC1),AII(ISC1))
      YMIT=CMPLX(BIR(ISC1),BII(ISC1))
      ZPED=CMPLX(CIR(ISC1),CII(ISC1))
      PWR=SI(ISC1)*TP*.5
      CUX=(CUX-YMIT*SIN(PWR)+ZPED*COS(PWR))*WLAM
      YMIT=CUX/VLT
      ZPED=VLT/CUX
      PWR=.5*REAL(VLT*CONJG(CUX))
      PIN=PIN+PWR
      IROW2=ITAG(ISC1)
   57 PRINT 64, IROW2,ISC1,VLT,CUX,ZPED,YMIT,PWR
      RETURN
C
   58 FORMAT (///,3X,47HMAXIMUM RELATIVE ASYMMETRY OF THE DRIVING POINT,
     121H ADMITTANCE MATRIX IS,E10.3,13H FOR SEGMENTS,I5,4H AND,I5,/,3X,
     225HRMS RELATIVE ASYMMETRY IS,E10.3)
   59 FORMAT (1X,44HERROR - - NETWORK ARRAY DIMENSIONS TOO SMALL)
   60 FORMAT (/,3X,3HTAG,3X,4HSEG.,4X,15HVOLTAGE (VOLTS),9X,14HCURRENT (
     1AMPS),9X,16HIMPEDANCE (OHMS),8X,17HADMITTANCE (MHOS),6X,5HPOWER,/,
     23X,3HNO.,3X,3HNO.,4X,4HREAL,8X,5HIMAG.,3(7X,4HREAL,8X,5HIMAG.),5X,
     37H(WATTS))
   61 FORMAT (///,27X,66H- - - STRUCTURE EXCITATION DATA AT NETWORK CONN
```

```
     1ECTION POINTS - - -)                                            NT 321
  62  FORMAT (2(1X,I5),9E12.5)                                        NT 322
  63  FORMAT (///,42X,36H- - - ANTENNA INPUT PARAMETERS - - -)        NT 323
  64  FORMAT (1X,I5,2H *,I4,9E12.5)                                   NT 324
      END                                                             NT 325-
```

NFPAT

PURPOSE

To compute and print the near E or H field over a range of points.

METHOD

The range of points in rectangular or spherical coordinates is obtained from parameters in COMMON/FPAT/. Subroutine NEFLD is called for near E field and NHFLD is called for near H field.

SYMBOL DICTIONARY

CPH	= cos ϕ
CTH	= cos θ
DXNR	= increment for x in rectangular coordinates or R in spherical coordinates
DYNR	= increment for y in rectangular coordinates or ϕ in spherical coordinates
DZNR	= increment for z in rectangular coordinates or θ in spherical coordinates
EX, EY, EZ	= x, y and z components of E or H
NEAR	= 0 for rectangular coordinates 1 for spherical coordinates
NFEH	= 0 for near E field 1 for near H field
NRX, NRY, NRZ	= number of values for x, y and z or R, ϕ, θ
SPH	= sin ϕ
STH	= sin θ
TA	= $\pi/180$
XNR	= initial x or R
XNRT	= x or R
XOB	= x
YNR	= initial y or ϕ
YNRT	= y or ϕ
YOB	= y
ZNR	= initial z or θ
ZNRT	= z or θ
ZOB	= z

```
      SUBROUTINE NFPAT                                                  NP   1
C     COMPUTE NEAR E OR H FIELDS OVER A RANGE OF POINTS                 NP   2
      COMPLEX EX,EY,EZ                                                  NP   3
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300 NP   4
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX( NP   5
     2300),WLAM,IPSYM                                                   NP   6
      COMMON /FPAT/ NTH,NPH,IPD,IAVP,INOR,IAX,THETS,PHIS,DTH,DPH,RFLD,GN NP   7
     1OR,CLT,CHT,EPSR2,SIG2,IXTYP,XPR6,PINR,PNLR,PLOSS,NEAR,NFEH,NRX,NRY NP   8
     2,NRZ,XNR,YNR,ZNR,DXNR,DYNR,DZNR                                   NP   9
      DATA TA/1.745329252E-02/                                          NP  10
      IF (NFEH.EQ.1) GO TO 1                                            NP  11
      PRINT 10                                                          NP  12
      GO TO 2                                                           NP  13
1     PRINT 12                                                          NP  14
2     ZNRT=ZNR-DZNR                                                     NP  15
      DO 9 I=1,NRZ                                                      NP  16
      ZNRT=ZNRT+DZNR                                                    NP  17
      IF (NEAR.EQ.0) GO TO 3                                            NP  18
      CTH=COS(TA*ZNRT)                                                  NP  19
      STH=SIN(TA*ZNRT)                                                  NP  20
3     YNRT=YNR-DYNR                                                     NP  21
      DO 9 J=1,NRY                                                      NP  22
      YNRT=YNRT+DYNR                                                    NP  23
      IF (NEAR.EQ.0) GO TO 4                                            NP  24
      CPH=COS(TA*YNRT)                                                  NP  25
      SPH=SIN(TA*YNRT)                                                  NP  26
4     XNRT=XNR-DXNR                                                     NP  27
      DO 9 KK=1,NRX                                                     NP  28
      XNRT=XNRT+DXNR                                                    NP  29
      IF (NEAR.EQ.0) GO TO 5                                            NP  30
      XOB=XNRT*STH*CPH                                                  NP  31
      YOB=XNRT*STH*SPH                                                  NP  32
      ZOB=XNRT*CTH                                                      NP  33
      GO TO 6                                                           NP  34
5     XOB=XNRT                                                          NP  35
      YOB=YNRT                                                          NP  36
      ZOB=ZNRT                                                          NP  37
6     TMP1=XOB/WLAM                                                     NP  38
      TMP2=YOB/WLAM                                                     NP  39
      TMP3=ZOB/WLAM                                                     NP  40
      IF (NFEH.EQ.1) GO TO 7                                            NP  41
      CALL NEFLD (TMP1,TMP2,TMP3,EX,EY,EZ)                              NP  42
      GO TO 8                                                           NP  43
7     CALL NHFLD (TMP1,TMP2,TMP3,EX,EY,EZ)                              NP  44
8     TMP1=CABS(EX)                                                     NP  45
      TMP2=CANG(EX)                                                     NP  46
      TMP3=CABS(EY)                                                     NP  47
      TMP4=CANG(EY)                                                     NP  48
      TMP5=CABS(EZ)                                                     NP  49
      TMP6=CANG(EZ)                                                     NP  50
      PRINT 11, XOB,YOB,ZOB,TMP1,TMP2,TMP3,TMP4,TMP5,TMP6               NP  51
9     CONTINUE                                                          NP  52
      RETURN                                                            NP  53
C                                                                       NP  54
10    FORMAT (///,35X,32H- - - NEAR ELECTRIC FIELDS - - -,//,12X,14H-  L NP  55
     1OCATION  -,21X,8H-  EX  -,15X,8H-  EY  -,15X,8H-  EZ  -,/,8X,1HX,1 NP  56
     20X,1HY,10X,1HZ,10X,9HMAGNITUDE,3X,5HPHASE,6X,9HMAGNITUDE,3X,5HPHAS NP  57
     3E,6X,9HMAGNITUDE,3X,5HPHASE,/,6X,6HMETERS,5X,6HMETERS,5X,6HMETERS, NP  58
     48X,7HVOLTS/M,3X,7HDEGREES,6X,7HVOLTS/M,3X,7HDEGREES,6X,7HVOLTS/M,3 NP  59
     5X,7HDEGREES)                                                      NP  60
11    FORMAT (2X,3(2X,F9.4),1X,3(3X,E11.4,2X,F7.2))                     NP  61
12    FORMAT (///,35X,32H- - - NEAR MAGNETIC FIELDS - - -,//,12X,14H-  L NP  62
     1OCATION  -,21X,8H-  HX  -,15X,8H-  HY  -,15X,8H-  HZ  -,/,8X,1HX,1 NP  63
     20X,1HY,10X,1HZ,10X,9HMAGNITUDE,3X,5HPHASE,6X,9HMAGNITUDE,3X,5HPHAS NP  64
```

```
     3E,6X,9HMAGNITUDE,3X,5HPHASE,/,6X,6HMETERS,5X,6HMETERS,5X,6HMETERS, NP  65
     49X,6HAMPS/M,3X,7HDEGREES,7X,6HAMPS/M,3X,7HDEGREES,7X,6HAMPS/M,3X,7 NP  66
     5HDEGREES)                                                          NP  67
      END                                                                NP  68-
```

NHFLD

PURPOSE

To compute the near magnetic field due to currents induced on a structure.

CODING

NH28 - NH56 Near H field due to currents on segments is computed.

NH29 - NH40 Each segment is checked to determine whether the field observation point (XOB, YOB, ZOB) falls within the segment volume. If it does, AX is set to the radius of that segment. AX is then sent to routine HSFLD as the radius of the observation segment to avoid a singularity in the field.

NH41 - NH56 Loop computing the field contribution of each segment.

NH42 - NH49 Parameters of source segment are stored in COMMON/DATAJ/.

NH50 HSFLD stores the magnetic field due to constant, sin ks, and cos ks currents in COMMON/DATAJ/.

NH54 - NH56 The field components are multiplied by the coefficients of the constant, sin ks, and cos ks components of the total segment current, and the field is summed.

NH58 - NH78 Near H fields due to patch currents are computed.

NH62 - NH71 Parameters of source patch are set in COMMON/DATAJ/.

NH72 H field is computed by HINTG.

NH76 - NH78 H fields due to $\hat{t}_1$ and $\hat{t}_2$ current components are multiplied by the current strengths and summed.

SYMBOL DICTIONARY

ACX = constant component of the segment current at NH51; $\hat{t}_1$ component of patch current at NH74

AX = segment radius when the field evaluation point falls within a segment volume

BCX = sin ks component of segment current at NH52; $\hat{t}_2$ component of patch current at NH75

CCX = cos ks component of segment current at NH53

HX, HY, HZ = total H field

T1X, T1Y, T1Z = arrays for $\hat{t}_1$

T1XJ, T1YJ, T1ZJ = $\hat{t}_1$ for patch I

T2X, T2Y, T2Z = arrays for $\hat{t}_2$

T2XJ, T2YJ, T2ZJ = $\hat{t}_2$ for patch I

XOB, YOB, ZOB = field evaluation point

ZP = coordinates of the field evaluation point, z or ρ^2, in a cylindrical coordinate system centered on the source segment.

CONSTANTS

0.5001 = fraction of segment length used to test whether the field evaluation point falls within a segment

0.9 = fraction of segment radius used to test whether the field evaluation point falls within a segment

```
      SUBROUTINE NHFLD (XOB,YOB,ZOB,HX,HY,HZ)
C
C     NHFLD COMPUTES THE NEAR FIELD AT SPECIFIED POINTS IN SPACE AFTER
C     THE STRUCTURE CURRENTS HAVE BEEN COMPUTED.
C
      COMPLEX HX,HY,HZ,CUR,ACX,BCX,CCX,EXK,EYK,EZK,EXS,EYS,EZS,EXC,EYC,E
     1ZC
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /ANGL/ SALP(300)
      COMMON /CRNT/ AIR(300),AII(300),BIR(300),BII(300),CIR(300),CII(300
     1),CUR(900)
      COMMON /DATAJ/ S,B,XJ,YJ,ZJ,CABJ,SABJ,SALPJ,EXK,EYK,EZK,EXS,EYS,EZ
     1S,EXC,EYC,EZC,RKH,IEXK,IND1,IND2,IPGND
      DIMENSION CAB(1), SAB(1)
      DIMENSION T1X(1), T1Y(1), T1Z(1), T2X(1), T2Y(1), T2Z(1), XS(1), Y
     1S(1), ZS(1)
      EQUIVALENCE (T1X,SI), (T1Y,ALP), (T1Z,BET), (T2X,ICON1), (T2Y,ICON
     12), (T2Z,ITAG), (XS,X), (YS,Y), (ZS,Z)
      EQUIVALENCE (T1XJ,CABJ), (T1YJ,SABJ), (T1ZJ,SALPJ), (T2XJ,B), (T2Y
     1J,IND1), (T2ZJ,IND2)
      EQUIVALENCE (CAB,ALP), (SAB,BET)
      HX=(0.,0.)
      HY=(0.,0.)
      HZ=(0.,0.)
      AX=0.
      IF (N.EQ.0) GO TO 4
      DO 1 I=1,N
      XJ=XOB-X(I)
      YJ=YOB-Y(I)
      ZJ=ZOB-Z(I)
      ZP=CAB(I)*XJ+SAB(I)*YJ+SALP(I)*ZJ
      IF (ABS(ZP).GT.0.5001*SI(I)) GO TO 1
      ZP=XJ*XJ+YJ*YJ+ZJ*ZJ-ZP*ZP
      XJ=BI(I)
      IF (ZP.GT.0.9*XJ*XJ) GO TO 1
      AX=XJ
      GO TO 2
1     CONTINUE
2     DO 3 I=1,N
      S=SI(I)
      B=BI(I)
      XJ=X(I)
      YJ=Y(I)
      ZJ=Z(I)
      CABJ=CAB(I)
      SABJ=SAB(I)
      SALPJ=SALP(I)
      CALL HSFLD (XOB,YOB,ZOB,AX)
      ACX=CMPLX(AIR(I),AII(I))
      BCX=CMPLX(BIR(I),BII(I))
      CCX=CMPLX(CIR(I),CII(I))
      HX=HX+EXK*ACX+EXS*BCX+EXC*CCX
      HY=HY+EYK*ACX+EYS*BCX+EYC*CCX
3     HZ=HZ+EZK*ACX+EZS*BCX+EZC*CCX
      IF (M.EQ.0) RETURN
4     JC=N
      JL=LD+1
      DO 5 I=1,M
      JL=JL-1
      S=BI(JL)
      XJ=X(JL)
      YJ=Y(JL)
```

```
      ZJ=Z(JL)                                                          NH  65
      T1XJ=T1X(JL)                                                      NH  66
      T1YJ=T1Y(JL)                                                      NH  67
      T1ZJ=T1Z(JL)                                                      NH  68
      T2XJ=T2X(JL)                                                      NH  69
      T2YJ=T2Y(JL)                                                      NH  70
      T2ZJ=T2Z(JL)                                                      NH  71
      CALL HINTG (XOB,YOB,ZOB)                                          NH  72
      JC=JC+3                                                           NH  73
      ACX=T1XJ*CUR(JC-2)+T1YJ*CUR(JC-1)+T1ZJ*CUR(JC)                    NH  74
      BCX=T2XJ*CUR(JC-2)+T2YJ*CUR(JC-1)+T2ZJ*CUR(JC)                    NH  75
      HX=HX+ACX*EXK+BCX*EXS                                             NH  76
      HY=HY+ACX*EYK+BCX*EYS                                             NH  77
5     HZ=HZ+ACX*EZK+BCX*EZS                                             NH  78
      RETURN                                                            NH  79
      END                                                               NH  80-
```

PATCH (entry SUBPH)

PURPOSE

To generate patch data for surfaces.

METHOD

The code from PA14 to PA129 generates data for a single new patch or multiple patches. There are four options for defining a single patch, as illustrated in Figure 5 of Part III. For a single patch, NX is zero and NY is NS + 1 where NS is the parameter from the SP input card and is shown on Figure 5. Rectangular, triangular or quadrilateral patches are defined by the coordinates of three or four corners in the parameters X1 though Z4. In the arbitrary shape option (Figure 5A in Part III) the center of the patch is X1, Y1, Z1; α is X2; β is Y2; and the area is Z2. The patch data is stored in COMMON/DATA/ from the top of the arrays downward (see Section III).

The code from PA131 to PA190 divides a patch into four patches and is used when a wire connects to a patch. If NY is equal to zero the patch NX is divided into four patches that become patches NX through NX + 3. Patches following NX are shifted in the arrays in COMMON/DATA/ to leave space for the three additional patches. If NY is greater than zero, patch NX is left in the arrays but four new patches to replace it are added to the end of the arrays. The z coordinate of patch NX is then changed to 10,000 at PA189.

SYMBOL DICTIONARY

MI	= array index for patch data
MIA	= array index for patch data
NTP	= patch type (NY for a single patch)
NX	= zero for a single patch. For multiple patches NX is defined in Figure 6 of Part III. After ENTRY SUBPH, NX is the number of the patch to be divided
S1X, S1Y, S1Z	= vector from corner 1 to corner 2
S2X, S2Y, S2Z	= vector from corner 2 to corner 3
SALN	= ±1 from array SALP
SALPN	= factor in computing center of mass of quadrilateral

PATCH

XA	=	$\lvert \bar{S}_1 \times \bar{S}_2 \rvert$ = area of rectangle or twice area of triangle (PA53)
XN2, YN2, ZN2	=	$\bar{S}_3 \times \bar{S}_4$ at PA79 to PA81. Line PA89 checks that the four corners are coplanar by the test $(\bar{S}_1 \times \bar{S}_2)\cdot(\bar{S}_3 \times \bar{S}_4)/\lvert \bar{S}_1 \times \bar{S}_2 \rvert \lvert \bar{S}_3 \times \bar{S}_4 \rvert > 0.9998$
XNV, YNV, ZNV	=	unit vector normal to the patch at PA54 to PA56
XS, YS, ZS	=	patch center at PA151 to PA153
XST	=	$\lvert \bar{S}_1 \times \bar{S}_2 \rvert$ at PA57

CONSTANTS

$0.9998 \approx \cos(1.°)$ in test for planar patch

```
      SUBROUTINE PATCH (NX,NY,X1,Y1,Z1,X2,Y2,Z2,X3,Y3,Z3,X4,Y4,Z4)
C     PATCH GENERATES AND MODIFIES PATCH GEOMETRY DATA
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /ANGL/ SALP(300)
      DIMENSION T1X(1), T1Y(1), T1Z(1), T2X(1), T2Y(1), T2Z(1)
      EQUIVALENCE (T1X,SI), (T1Y,ALP), (T1Z,BET), (T2X,ICON1), (T2Y,ICON
     12), (T2Z,ITAG)
C     NEW PATCHES.  FOR NX=0, NY=1,2,3,4 PATCH IS (RESPECTIVELY)
C     ARBITRARY, RECTAGULAR, TRIANGULAR, OR QUADRILATERAL.
C     FOR NX AND NY .GT. 0 A RECTANGULAR SURFACE IS PRODUCED WITH
C     NX BY NY RECTANGULAR PATCHES.
      M=M+1
      MI=LD+1-M
      NTP=NY
      IF (NX.GT.0) NTP=2
      IF (NTP.GT.1) GO TO 2
      X(MI)=X1
      Y(MI)=Y1
      Z(MI)=Z1
      BI(MI)=Z2
      ZNV=COS(X2)
      XNV=ZNV*COS(Y2)
      YNV=ZNV*SIN(Y2)
      ZNV=SIN(X2)
      XA=SQRT(XNV*XNV+YNV*YNV)
      IF (XA.LT.1.E-6) GO TO 1
      T1X(MI)=-YNV/XA
      T1Y(MI)=XNV/XA
      T1Z(MI)=0.
      GO TO 6
1     T1X(MI)=1.
      T1Y(MI)=0.
      T1Z(MI)=0.
      GO TO 6
2     S1X=X2-X1
      S1Y=Y2-Y1
      S1Z=Z2-Z1
      S2X=X3-X2
      S2Y=Y3-Y2
      S2Z=Z3-Z2
      IF (NX.EQ.0) GO TO 3
      S1X=S1X/NX
      S1Y=S1Y/NX
      S1Z=S1Z/NX
      S2X=S2X/NY
      S2Y=S2Y/NY
      S2Z=S2Z/NY
3     XNV=S1Y*S2Z-S1Z*S2Y
      YNV=S1Z*S2X-S1X*S2Z
      ZNV=S1X*S2Y-S1Y*S2X
      XA=SQRT(XNV*XNV+YNV*YNV+ZNV*ZNV)
      XNV=XNV/XA
      YNV=YNV/XA
      ZNV=ZNV/XA
      XST=SQRT(S1X*S1X+S1Y*S1Y+S1Z*S1Z)
      T1X(MI)=S1X/XST
      T1Y(MI)=S1Y/XST
      T1Z(MI)=S1Z/XST
      IF (NTP.GT.2) GO TO 4
      X(MI)=X1+.5*(S1X+S2X)
      Y(MI)=Y1+.5*(S1Y+S2Y)
      Z(MI)=Z1+.5*(S1Z+S2Z)
```

```
      BI(MI)=XA
      GO TO 6
    4 IF (NTP.EQ.4) GO TO 5
      X(MI)=(X1+X2+X3)/3.
      Y(MI)=(Y1+Y2+Y3)/3.
      Z(MI)=(Z1+Z2+Z3)/3.
      BI(MI)=.5*XA
      GO TO 6
    5 S1X=X3-X1
      S1Y=Y3-Y1
      S1Z=Z3-Z1
      S2X=X4-X1
      S2Y=Y4-Y1
      S2Z=Z4-Z1
      XN2=S1Y*S2Z-S1Z*S2Y
      YN2=S1Z*S2X-S1X*S2Z
      ZN2=S1X*S2Y-S1Y*S2X
      XST=SQRT(XN2*XN2+YN2*YN2+ZN2*ZN2)
      SALPN=1./(3.*(XA+XST))
      X(MI)=(XA*(X1+X2+X3)+XST*(X1+X3+X4))*SALPN
      Y(MI)=(XA*(Y1+Y2+Y3)+XST*(Y1+Y3+Y4))*SALPN
      Z(MI)=(XA*(Z1+Z2+Z3)+XST*(Z1+Z3+Z4))*SALPN
      BI(MI)=.5*(XA+XST)
      S1X=(XNV*XN2+YNV*YN2+ZNV*ZN2)/XST
      IF (S1X.GT.0.9998) GO TO 6
      PRINT 14
      STOP
    6 T2X(MI)=YNV*T1Z(MI)-ZNV*T1Y(MI)
      T2Y(MI)=ZNV*T1X(MI)-XNV*T1Z(MI)
      T2Z(MI)=XNV*T1Y(MI)-YNV*T1X(MI)
      SALP(MI)=1.
      IF (NX.EQ.0) GO TO 8
      M=M+NX*NY-1
      XN2=X(MI)-S1X-S2X
      YN2=Y(MI)-S1Y-S2Y
      ZN2=Z(MI)-S1Z-S2Z
      XS=T1X(MI)
      YS=T1Y(MI)
      ZS=T1Z(MI)
      XT=T2X(MI)
      YT=T2Y(MI)
      ZT=T2Z(MI)
      MI=MI+1
      DO 7 IY=1,NY
      XN2=XN2+S2X
      YN2=YN2+S2Y
      ZN2=ZN2+S2Z
      DO 7 IX=1,NX
      XST=IX
      MI=MI-1
      X(MI)=XN2+XST*S1X
      Y(MI)=YN2+XST*S1Y
      Z(MI)=ZN2+XST*S1Z
      BI(MI)=XA
      SALP(MI)=1
      T1X(MI)=XS
      T1Y(MI)=YS
      T1Z(MI)=ZS
      T2X(MI)=XT
      T2Y(MI)=YT
    7 T2Z(MI)=ZT
    8 IPSYM=0
      NP=N
      MP=M
```

```
      RETURN
C     DIVIDE PATCH FOR WIRE CONNECTION
      ENTRY SUBPH(NX,NY,X1,Y1,Z1,X2,Y2,Z2,X3,Y3,Z3,X4,Y4,Z4)
      IF (NY.GT.0) GO TO 10
      IF (NX.EQ.M) GO TO 10
      NXP=NX+1
      IX=LD-M
      DO 9 IY=NXP,M
      IX=IX+1
      NYP=IX-3
      X(NYP)=X(IX)
      Y(NYP)=Y(IX)
      Z(NYP)=Z(IX)
      BI(NYP)=BI(IX)
      SALP(NYP)=SALP(IX)
      T1X(NYP)=T1X(IX)
      T1Y(NYP)=T1Y(IX)
      T1Z(NYP)=T1Z(IX)
      T2X(NYP)=T2X(IX)
      T2Y(NYP)=T2Y(IX)
9     T2Z(NYP)=T2Z(IX)
10    MI=LD+1-NX
      XS=X(MI)
      YS=Y(MI)
      ZS=Z(MI)
      XA=BI(MI)*.25
      XST=SQRT(XA)*.5
      S1X=T1X(MI)
      S1Y=T1Y(MI)
      S1Z=T1Z(MI)
      S2X=T2X(MI)
      S2Y=T2Y(MI)
      S2Z=T2Z(MI)
      SALN=SALP(MI)
      XT=XST
      YT=XST
      IF (NY.GT.0) GO TO 11
      MIA=MI
      GO TO 12
11    M=M+1
      MP=MP+1
      MIA=LD+1-M
12    DO 13 IX=1,4
      X(MIA)=XS+XT*S1X+YT*S2X
      Y(MIA)=YS+XT*S1Y+YT*S2Y
      Z(MIA)=ZS+XT*S1Z+YT*S2Z
      BI(MIA)=XA
      T1X(MIA)=S1X
      T1Y(MIA)=S1Y
      T1Z(MIA)=S1Z
      T2X(MIA)=S2X
      T2Y(MIA)=S2Y
      T2Z(MIA)=S2Z
      SALP(MIA)=SALN
      IF (IX.EQ.2) YT=-YT
      IF (IX.EQ.1.OR.IX.EQ.3) XT=-XT
      MIA=MIA-1
13    CONTINUE
      M=M+3
      IF (NX.LE.MP) MP=MP+3
      IF (NY.GT.0) Z(MI)=10000.
      RETURN
C
14    FORMAT (62H ERROR -- CORNERS OF QUADRILATERAL PATCH DO NOT LIE IN
```

```
     1A PLANE)                                                          PA 193
      END                                                               PA 194-
```

PCINT

PURPOSE

To compute the interaction matrix elements representing the electric field, tangent to a segment connected to a surface, due to the current on the four patches around the connection point.

METHOD

The four patches at the base of a connected wire are located as shown in figure 10 with respect to the vectors $\hat{t}_1$ and $\hat{t}_2$, where patch numbers indicate the order of the patches in the data arrays. The position of a point on the surface is defined by $\bar{\rho}\,(S_1,S_2) = \bar{\rho}_0 + S_1\hat{t}_1 + S_2\hat{t}_2$, where $\bar{\rho}_0$ is the position of the center of the four patches where the wire connects, and S_1 and S_2 are coordinates measured from the center. The current over the surface is represented by $\bar{J}(S_1,S_2)$, the currents at the centers of the four patches are

$$\bar{J}_1 = \bar{J}(d,d)$$
$$\bar{J}_2 = \bar{J}(-d,d)$$
$$\bar{J}_3 = \bar{J}(-d,-d)$$
$$\bar{J}_4 = \bar{J}(d,-d)$$

and the current at the base of the segment, flowing onto the surface, is I_0. The current interpolation function is then

$$\bar{J}(S_1,S_2) = \left[\bar{f}(S_1,S_2) - \sum_{i=1}^{4} g_i(S_1,S_2)\,\bar{f}_i\right] I_0 + \sum_{i=1}^{4} g_i(S_1,S_2)\bar{J}_i \ ,$$

where

$$\bar{f}(S_1,S_2) = \frac{S_1\hat{t}_1 + S_2\hat{t}_2}{2\pi(S_1^{\,2} + S_2^{\,2})}$$

$$\bar{f}_1 = \bar{f}(d,d) \quad = (\hat{t}_1 + \hat{t}_2)/(4\pi d)$$
$$\bar{f}_2 = \bar{f}(-d,d) \quad = (-\hat{t}_1 + \hat{t}_2)/(4\pi d)$$
$$\bar{f}_3 = \bar{f}(-d,-d) = (-\hat{t}_1 - \hat{t}_2)/(4\pi d)$$
$$\bar{f}_4 = \bar{f}(d,-d) \quad = (\hat{t}_1 - \hat{t}_2)/(4\pi d)$$

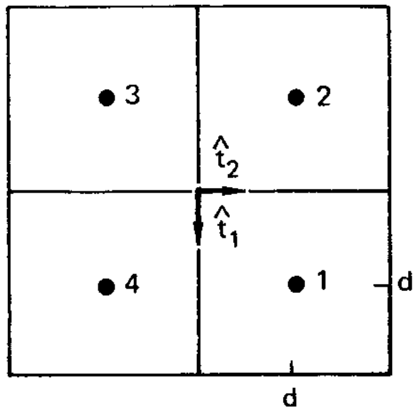


Figure 10. Patches at a Wire Connection Point.

$$g_1(S_1,S_2) = (d + S_1)(d + S_2)/(4d^2)$$

$$g_2(S_1,S_2) = (d - S_1)(d + S_2)/(4d^2)$$

$$g_3(S_1,S_2) = (d - S_1)(d - S_2)/(4d^2)$$

$$g_4(S_1,S_2) = (d + S_1)(d - S_2)/(4d^2)$$

If $\overline{\Gamma}_1(\overline{\rho})dA$ and $\overline{\Gamma}_2(\overline{\rho})dA$ are the electric fields at the center of the connected segment due to unit currents at $\overline{\rho}$ on the surface dA, flowing in the directions $\hat{t}_1$ and $\hat{t}_2$, respectively, the nine matrix elements to be computed are

$$E_1 = \int_S g_1(S_1,S_2)\ \hat{i} \cdot \overline{\Gamma}_1(\overline{\rho})dA$$

$$E_2 = \int_S g_2(S_1,S_2)\ \hat{i} \cdot \overline{\Gamma}_1(\overline{\rho})dA$$

$$E_3 = \int_S g_3(S_1,S_2)\ \hat{i} \cdot \overline{\Gamma}_1(\overline{\rho})dA$$

$$E_4 = \int_S g_4(S_1,S_2)\ \hat{i} \cdot \overline{\Gamma}_1(\overline{\rho})dA$$

$$E_5 = \int_S g_1(S_1,S_2)\ \hat{i} \cdot \overline{\Gamma}_2(\overline{\rho})dA$$

$$E_6 = \int_S g_2(S_1,S_2)\ \hat{i} \cdot \bar{\Gamma}_2(\bar{\rho})dA$$

$$E_7 = \int_S g_3(S_1,S_2)\ \hat{i} \cdot \bar{\Gamma}_2(\bar{\rho})dA$$

$$E_8 = \int_S g_4(S_1,S_2)\ \hat{i} \cdot \bar{\Gamma}_2(\bar{\rho})dA$$

$$E_9 = \int_S \left\{ \left[\bar{h}(S_1,S_2) \cdot \hat{t}_1\right] \left[\hat{i} \cdot \bar{\Gamma}_1(\bar{\rho})\right] + \left[\bar{h}(S_1,S_2) \cdot \hat{t}_2\right] \left[\hat{i} \cdot \bar{\Gamma}_2(\bar{\rho})\right] \right\} dA$$

where

$$\bar{h}(S_1,S_2) = \bar{f}(S_1,S_2) - \sum_{i=1}^{4} g_i(S_1,S_2)\bar{f}_i\ ,$$

and where $\hat{i}$ = the unit vector in the direction of the connected segment.

The integration is over the total area of the four patches and is performed by numerical quadrature. The number of increments in S_1 and S_2 used in integration is set by the variable NINT. When PCINT is called, the parameters in COMMON/DATAJ/ have the values for the first connected patch. During integration, these parameters are set for each integration patch. At the end of PCINT, they are reset to their original values.

SYMBOL DICTIONARY

CABI	=	x component of $\hat{i}$
D	=	d
DA	=	area of the surface element used in integration
DS	=	width of the surface element of area DA
E	=	array used to return the values E_1, E_2, ..., E_9
EXK, EYK, EZK	=	x, y, and z components of $\bar{\Gamma}_1(\bar{\rho})$DA at PC50; at PC51, EXK is set to $\hat{i} \cdot \bar{\Gamma}_1(\bar{\rho})$DA
EXS, EYS, EZS	=	x, y, and z components of $\bar{\Gamma}_2(\bar{\rho})$DA at PC50; at PC52, EXS is set to $\hat{i} \cdot \bar{\Gamma}_2(\bar{\rho})$DA

E1 = E_1
E2 = E_2
E3 = E_3
E4 = E_4
E5 = E_5
E6 = E_6
E7 = E_7
E8 = E_8
E9 = E_9
FCON = $1/(4\pi d)$ factor in $\overline{f}_1$, $\overline{f}_2$, ...
F1 = $\overline{h}(S_1,S_2) \cdot \hat{t}_1$
F2 = $\overline{h}(S_1,S_2) \cdot \hat{t}_2$
GCON = $1/(4d^2)$ factor in $g_1(S_1,S_2)$, ...
G1 = $g_1(S_1,S_2)$
G2 = $g_2(S_1,S_2)$
G3 = $g_3(S_1,S_2)$
G4 = $g_4(S_1,S_2)$
I1 = DO loop index
I2 = DO loop index
NINT = number of steps in S_1 and S_3 used in approximating the integrals for E_1, E_2, ..., E_9
S = area of each of the four patches at PC11; area of the surface element used in integration at PC20
SABI = y component of $\hat{i}$
SALPI = z component of $\hat{i}$
S1 = S_1
S2 = S_2
S2X = initial value of S_2
TPI = 2π
T1XJ, T1YJ, T1ZJ = x, y, and z components of $\hat{t}_1$
T2XJ, T2YJ, T2ZJ = x, y, and z components of $\hat{t}_2$
XI = x coordinate of the center of the connected segment

XJ, YJ, ZJ	= center of first patch above PC41; center of integration element below PC41
XS	= x component of $\overline{\rho}(S_1,S_2)$
XSS	= initial x coordinate of $\overline{\rho}(S_1,S_2)$
XXJ, XYJ, XZJ	= initial value of XJ, YJ, ZJ saved
X1	= x component of $\overline{\rho}(d,d)$ used as reference for computing $\overline{\rho}(S_1,S_2)$
YI	= y coordinate of the center of the connected segment
YS	= y component of $\overline{\rho}(S_1,S_2)$
YSS	= initial y component of $\overline{\rho}(S_1,S_2)$
Y1	= y component of $\overline{\rho}(d,d)$
ZI	= z coordinate of the center of the connected segment
ZS	= z component of $\overline{\rho}(S_1,S_2)$
ZSS	= initial z component of $\overline{\rho}(S_1,S_2)$
Z1	= z component of $\overline{\rho}(d,d)$

```
      SUBROUTINE PCINT (XI,YI,ZI,CABI,SABI,SALPI,E)
C     INTEGRATE OVER PATCHES AT WIRE CONNECTION POINT
      COMPLEX EXK,EYK,EZK,EXS,EYS,EZS,EXC,EYC,EZC,E,E1,E2,E3,E4,E5,E6,E7
     1,E8,E9
      COMMON /DATAJ/ S,B,XJ,YJ,ZJ,CABJ,SABJ,SALPJ,EXK,EYK,EZK,EXS,EYS,EZ
     1S,EXC,EYC,EZC,RKH,IEXK,IND1,IND2,IPGND
      DIMENSION E(9)
      EQUIVALENCE (T1XJ,CABJ), (T1YJ,SABJ), (T1ZJ,SALPJ), (T2XJ,B), (T2Y
     1J,IND1), (T2ZJ,IND2)
      DATA TPI/6.283185308/,NINT/10/
      D=SQRT(S)*.5
      DS=4.*D/FLOAT(NINT)
      DA=DS*DS
      GCON=1./S
      FCON=1./(2.*TPI*D)
      XXJ=XJ
      XYJ=YJ
      XZJ=ZJ
      XS=S
      S=DA
      S1=D+DS*.5
      XSS=XJ+S1*(T1XJ+T2XJ)
      YSS=YJ+S1*(T1YJ+T2YJ)
      ZSS=ZJ+S1*(T1ZJ+T2ZJ)
      S1=S1+D
      S2X=S1
      E1=(0.,0.)
      E2=(0.,0.)
      E3=(0.,0.)
      E4=(0.,0.)
      E5=(0.,0.)
      E6=(0.,0.)
      E7=(0.,0.)
      E8=(0.,0.)
      E9=(0.,0.)
      DO 1 I1=1,NINT
      S1=S1-DS
      S2=S2X
      XSS=XSS-DS*T1XJ
      YSS=YSS-DS*T1YJ
      ZSS=ZSS-DS*T1ZJ
      XJ=XSS
      YJ=YSS
      ZJ=ZSS
      DO 1 I2=1,NINT
      S2=S2-DS
      XJ=XJ-DS*T2XJ
      YJ=YJ-DS*T2YJ
      ZJ=ZJ-DS*T2ZJ
      CALL UNERE (XI,YI,ZI)
      EXK=EXK*CABI+EYK*SABI+EZK*SALPI
      EXS=EXS*CABI+EYS*SABI+EZS*SALPI
      G1=(D+S1)*(D+S2)*GCON
      G2=(D-S1)*(D+S2)*GCON
      G3=(D-S1)*(D-S2)*GCON
      G4=(D+S1)*(D-S2)*GCON
      F2=(S1*S1+S2*S2)*TPI
      F1=S1/F2-(G1-G2-G3+G4)*FCON
      F2=S2/F2-(G1+G2-G3-G4)*FCON
      E1=E1+EXK*G1
      E2=E2+EXK*G2
      E3=E3+EXK*G3
      E4=E4+EXK*G4
      E5=E5+EXS*G1
```

```
      E6=E6+EXS*G2                                                      PC  65
      E7=E7+EXS*G3                                                      PC  66
      E8=E8+EXS*G4                                                      PC  67
     1 E9=E9+EXK*F1+EXS*F2                                              PC  68
      E(1)=E1                                                           PC  69
      E(2)=E2                                                           PC  70
      E(3)=E3                                                           PC  71
      E(4)=E4                                                           PC  72
      E(5)=E5                                                           PC  73
      E(6)=E6                                                           PC  74
      E(7)=E7                                                           PC  75
      E(8)=E8                                                           PC  76
      E(9)=E9                                                           PC  77
      XJ=XXJ                                                            PC  78
      YJ=XYJ                                                            PC  79
      ZJ=XZJ                                                            PC  80
      S=XS                                                              PC  81
      RETURN                                                            PC  82
      END                                                               PC  83-
```

PRNT

PURPOSE

To set up the formats for printing a record of three integers, six floating point numbers, and a Hollerith string, where the variables equal to zero are replaced by blanks. This routine is used by LOAD in printing the impedance data table.

METHOD

A variable format is used to generate the record with arbitrary blank fill. Elements of the format are picked from the array IFORM in the DATA statement. Through IF statements operating on the subroutine input quantities, this routine chooses the desired format elements and builds the format in the array IVAR. The program is divided into two sections: the first builds the integer part of the format and the second the floating point part.

SYMBOL DICTIONARY

ABS	= external routine (absolute value)
FL	= elements of this array are set equal to the floating point input quantities FL1 - FL6
FLT	= array of non-zero floating point input quantities to be printed
FL1, FL2, FL3, FL4, FL5, FL6	= input floating point quantities
HALL	= 4H ALL (Hollerith ALL)
I	= DO loop index
IA	= input Hollerith string (array)
ICHAR	= number of characters in the input Hollerith string
IFORM	= array containing format elements
IN	= array set equal to input integer quantities (IN1 - IN3)
INT	= non-zero integer quantities to be printed
IN1, IN2, IN3	= input integer quantities
IVAR	= variable format array

IL	=	DO loop limit
J	=	implied DO loop index
K	=	index parameter
L	=	implied DO loop index
NCPW	=	number of Hollerith characters per computer word
NFLT	=	floating point print index, number of non-zero reals
NINT	=	integer print index; number of non-zero integers
NWORDS	=	number of computer words in the input Hollerith string

```
      SUBROUTINE PRNT (IN1,IN2,IN3,FL1,FL2,FL3,FL4,FL5,FL6,IA,ICHAR)
C
C     PRNT SETS UP THE PRINT FORMATS FOR IMPEDANCE LOADING
C
      DIMENSION IVAR(13), IA(1), IFORM(8), IN(3), INT(3), FL(6), FLT(6)
      INTEGER HALL
      DATA IFORM/5H(/3X,,3HI5,,3H5X,,3HA5,,6HE13.4,,4H13X,,3H3X,,5H2A10)
     1/
C
C     NUMBER OF CHARACTERS PER COMPUTER WORD IS NCPW
C
      DATA NCPW/10/,HALL/4H ALL/
      NWORDS=(ICHAR-1)/NCPW+1
      IN(1)=IN1
      IN(2)=IN2
      IN(3)=IN3
      FL(1)=FL1
      FL(2)=FL2
      FL(3)=FL3
      FL(4)=FL4
      FL(5)=FL5
      FL(6)=FL6
C
C     INTEGER FORMAT
C
      NINT=0
      IVAR(1)=IFORM(1)
      K=1
      I1=1
      IF (.NOT.(IN1.EQ.0.AND.IN2.EQ.0.AND.IN3.EQ.0)) GO TO 1
      INT(1)=HALL
      NINT=1
      I1=2
      K=K+1
      IVAR(K)=IFORM(4)
    1 DO 3 I=I1,3
      K=K+1
      IF (IN(I).EQ.0) GO TO 2
      NINT=NINT+1
      INT(NINT)=IN(I)
      IVAR(K)=IFORM(2)
      GO TO 3
    2 IVAR(K)=IFORM(3)
    3 CONTINUE
      K=K+1
      IVAR(K)=IFORM(7)
C
C     FLOATING POINT FORMAT
C
      NFLT=0
      DO 5 I=1,6
      K=K+1
      IF (ABS(FL(I)).LT.1.E-20) GO TO 4
      NFLT=NFLT+1
      FLT(NFLT)=FL(I)
      IVAR(K)=IFORM(5)
      GO TO 5
    4 IVAR(K)=IFORM(6)
    5 CONTINUE
      K=K+1
      IVAR(K)=IFORM(7)
      K=K+1
      IVAR(K)=IFORM(8)
      PRINT IVAR, (INT(I),I=1,NINT),(FLT(J),J=1,NFLT),(IA(L),L=1,NWORDS)
```

```
65        RETURN                                                     PR  65
66        END                                                        PR  66-
```

QDSRC

PURPOSE

To fill the excitation array for a current slope discontinuity voltage source.

METHOD

The current slope discontinuity voltage source is described in section IV-1 of Part I.

CODING

QD22 - QD25 The connection number for end 1 of segment IS is temporarily set to 0, and TBF is called to generate the function $f_\ell^*(s)$ for ℓ - IS. The zero in the second argument of TBF causes f_ℓ^* to go to zero at the first end of segment IS rather than the usual non-zero value that allows for current flowing onto the wire end cap.

QD26 - QD31 β_ℓ is computed and other quantities set.

QD32 - QD119 This loop computes the fields due to each segment on which f_ℓ^* is non-zero.

QD33 - QD77 Parameters of the source segment are stored in COMMON/DATAJ/. Flags for the extended thin wire approximation are set as in routine CMSET.

QD78 - QD91 This loop evaluates the electric field on each segment.

QD95 - QD116 This loop evaluates the magnetic field at each patch.

SYMBOL DICTIONARY

AI = radius of segment on which field is evaluated.

CABI = x component of unit vector in the direction of segment I

CCJ = CCJX = -j/60

CURD = β_ℓ

E = array of segment and patch excitation fields

ETC, ETK, ETS = E field tangent to a segment or H field components on a patch due to cosine, constant, and sine current components, respectively, on a segment

I1 = array index for patch excitation

IJ = flag which, if zero, indicates that the field is being evaluated on the source segment

IPR = temporary storage of connection number

IS = segment which has the source location on end 1

J = source segment number

SABI = y component of unit vector in the direction of segment I

T1X, T1Y, T1Z, T2X, T2Y, T2Z = arrays of components of $\hat{t}_1$ and $\hat{t}_2$ for patches

TP = 2π

TX, TY, TZ = components of $\hat{t}_1$ or $\hat{t}_2$ for patches

V = source voltage

XI, YI, ZI = coordinates of point where field is evaluated; XI is also used in the test for the extended thin wire approximation for the electric field

CONSTANTS

0.01666666667 = 1/60

0.999999 = minimum XI for the extended thin wire approximation (maximum angle = 0.08 degrees)

6.283185308 = 2π

```
      SUBROUTINE QDSRC (IS,V,E)
C     FILL INCIDENT FIELD ARRAY FOR CHARGE DISCONTINUITY VOLTAGE SOURCE
      COMPLEX VQDS,CURD,CCJ,V,EXK,EYK,EZK,EXS,EYS,EZS,EXC,EYC,EZC,ETK,ET
     1S,ETC,VSANT,VQD,E,ZARRAY
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /VSORC/ VQD(30),VSANT(30),VQDS(30),IVQD(30),ISANT(30),IQDS(
     130),NVQD,NSANT,NQDS
      COMMON /SEGJ/ AX(30),BX(30),CX(30),JCO(30),JSNO,ISCON(50),NSCON,IP
     1CON(10),NPCON
      COMMON /DATAJ/ S,B,XJ,YJ,ZJ,CABJ,SABJ,SALPJ,EXK,EYK,EZK,EXS,EYS,EZ
     1S,EXC,EYC,EZC,RKH,IEXK,IND1,IND2,IPGND
      COMMON /ANGL/ SALP(300)
      COMMON /ZLOAD/ ZARRAY(300),NLOAD,NLODF
      DIMENSION CCJX(2), E(1), CAB(1), SAB(1)
      DIMENSION T1X(1), T1Y(1), T1Z(1), T2X(1), T2Y(1), T2Z(1)
      EQUIVALENCE (CCJ,CCJX), (CAB,ALP), (SAB,BET)
      EQUIVALENCE (T1X,SI), (T1Y,ALP), (T1Z,BET), (T2X,ICON1), (T2Y,ICON
     12), (T2Z,ITAG)
      DATA TP/6.283185308/,CCJX/0.,-.01666666667/
      I=ICON1(IS)
      ICON1(IS)=0
      CALL TBF (IS,0)
      ICON1(IS)=I
      S=SI(IS)*.5
      CURD=CCJ*V/((ALOG(2.*S/BI(IS))-1.)*(BX(JSNO)*COS(TP*S)+CX(JSNO)*SI
     1N(TP*S))*WLAM)
      NQDS=NQDS+1
      VQDS(NQDS)=V
      IQDS(NQDS)=IS
      DO 20 JX=1,JSNO
      J=JCO(JX)
      S=SI(J)
      B=BI(J)
      XJ=X(J)
      YJ=Y(J)
      ZJ=Z(J)
      CABJ=CAB(J)
      SABJ=SAB(J)
      SALPJ=SALP(J)
      IF (IEXK.EQ.0) GO TO 16
      IPR=ICON1(J)
      IF (IPR) 1,6,2
1     IPR=-IPR
      IF (-ICON1(IPR).NE.J) GO TO 7
      GO TO 4
2     IF (IPR.NE.J) GO TO 3
      IF (CABJ*CABJ+SABJ*SABJ.GT.1.E-8) GO TO 7
      GO TO 5
3     IF (ICON2(IPR).NE.J) GO TO 7
4     XI=ABS(CABJ*CAB(IPR)+SABJ*SAB(IPR)+SALPJ*SALP(IPR))
      IF (XI.LT.0.999999) GO TO 7
      IF (ABS(BI(IPR)/B-1.).GT.1.E-6) GO TO 7
5     IND1=0
      GO TO 8
6     IND1=1
      GO TO 8
7     IND1=2
8     IPR=ICON2(J)
      IF (IPR) 9,14,10
9     IPR=-IPR
      IF (-ICON2(IPR).NE.J) GO TO 15
      GO TO 12
```

```
   10 IF (IPR.NE.J) GO TO 11
      IF (CABJ*CABJ+SABJ*SABJ.GT.1.E-8) GO TO 15
      GO TO 13
   11 IF (ICON1(IPR).NE.J) GO TO 15
   12 XI=ABS(CABJ*CAB(IPR)+SABJ*SAB(IPR)+SALPJ*SALP(IPR))
      IF (XI.LT.0.999999) GO TO 15
      IF (ABS(BI(IPR)/B-1.).GT.1.E-6) GO TO 15
   13 IND2=0
      GO TO 16
   14 IND2=1
      GO TO 16
   15 IND2=2
   16 CONTINUE
      DO 17 I=1,N
      IJ=I-J
      XI=X(I)
      YI=Y(I)
      ZI=Z(I)
      AI=BI(I)
      CALL EFLD (XI,YI,ZI,AI,IJ)
      CABI=CAB(I)
      SABI=SAB(I)
      SALPI=SALP(I)
      ETK=EXK*CABI+EYK*SABI+EZK*SALPI
      ETS=EXS*CABI+EYS*SABI+EZS*SALPI
      ETC=EXC*CABI+EYC*SABI+EZC*SALPI
   17 E(I)=E(I)-(ETK*AX(JX)+ETS*BX(JX)+ETC*CX(JX))*CURD
      IF (M.EQ.0) GO TO 19
      IJ=LD+1
      I1=N
      DO 18 I=1,M
      IJ=IJ-1
      XI=X(IJ)
      YI=Y(IJ)
      ZI=Z(IJ)
      CALL HSFLD (XI,YI,ZI,0.)
      I1=I1+1
      TX=T2X(IJ)
      TY=T2Y(IJ)
      TZ=T2Z(IJ)
      ETK=EXK*TX+EYK*TY+EZK*TZ
      ETS=EXS*TX+EYS*TY+EZS*TZ
      ETC=EXC*TX+EYC*TY+EZC*TZ
      E(I1)=E(I1)+(ETK*AX(JX)+ETS*BX(JX)+ETC*CX(JX))*CURD*SALP(IJ)
      I1=I1+1
      TX=T1X(IJ)
      TY=T1Y(IJ)
      TZ=T1Z(IJ)
      ETK=EXK*TX+EYK*TY+EZK*TZ
      ETS=EXS*TX+EYS*TY+EZS*TZ
      ETC=EXC*TX+EYC*TY+EZC*TZ
   18 E(I1)=E(I1)+(ETK*AX(JX)+ETS*BX(JX)+ETC*CX(JX))*CURD*SALP(IJ)
   19 IF (NLOAD.GT.0.OR.NLODF.GT.0) E(J)=E(J)+ZARRAY(J)*CURD*(AX(JX)+CX(
     1JX))
   20 CONTINUE
      RETURN
      END
```

RDPAT

PURPOSE

To compute and print radiated field quantities.

METHOD

The quantities computed and the output formats depend on the options selected by the first integer (IFAR) and fourth integer (IPD, IAVP, INOR, IAX) on the RP card (see Part III). These quantities are defined as follows:

(1) Power Gain

In the direction (θ,ϕ)

$$G_p(\theta,\phi) = 4\pi \frac{P_\Omega(\theta,\phi)}{P_{in}} ,$$

where $P_\Omega(\theta,\phi)$ is the power radiated per unit solid angle in the given direction, and P_{in} is the total power accepted by the antenna. Therefore, $P_{in} = (1/2)\mathrm{Re}(VI^*)$, where V is the applied source voltage, and

$$P_\Omega(\theta,\phi) = (1/2)\ R^2\mathrm{Re}(\bar{E}\ \mathrm{X}\ \bar{H}^*) = \frac{R^2}{2\eta}\ \bar{E}\ .\ \bar{E}^*,$$

where R is the observation sphere radius. Since the electric field calculated by FFLD (call it $\bar{E}'$) does not include $\exp(-jkR)/(R/\lambda)$,

$$\bar{E} = \frac{\exp(-jkR)}{R/\lambda}\ \bar{E}'$$

and

$$P_\Omega = \frac{\lambda^2}{2\eta}\ (\bar{E}' \cdot \bar{E}'^*) \quad .$$

Thus,

$$G_p(\theta,\varphi) = \frac{2\pi\lambda^2}{\eta P_{in}} (\bar{E}' \cdot \bar{E}'^*)$$

in terms of the program variables.

(2) Directive Gain

In the direction (θ,ϕ),

$$G_d(\theta,\phi) = 4\pi \frac{P_\Omega(\theta,\phi)}{P_{rad}}$$

where P_{rad} is the total power radiated by the antenna. The only difference from power gain is that P_{in} is replaced by P_{rad}, and $P_{rad} = P_{in} - P_{loss}$, where P_{loss} is calculated as the power lost in distributed and lumped loads on the structure and in the networks loads.

(3) Component Gain

The gains are also calculated for separate, orthogonal field components (u, v). In this case, $\bar{E}' \cdot \bar{E}'^*$ is replaced by $E_u' E_u'^*$ or $E_v' E_v'^*$, and the total gain is the sum of the two components.

(4) Average Gain

The user specifies a range and number of points in theta and phi that in turn specify the total solid angle covered, Ω, and the sampling density for the integral in the expression for average gain:

$$G_{av} = \frac{\int_\Omega G_p \, d\Omega}{\Omega}$$

The trapezoidal rule is used in evaluating the integral.

(5) Normalized Gain

Normalized gain is simply the gain divided by its maximum value or some value specified by the user.

The discussion of gains applies only to the case of a structure used as a radiating antenna. For the case of an incident plane wave, the program constants are defined such that the value of σ/λ^2 is printed under the heading "GAIN." The calculation is

$$\frac{\sigma}{\lambda^2} = \frac{4\pi R^2}{\lambda^2} \frac{W_{scat}}{W_{inc}} = \frac{4\pi}{\bar{E}_{inc} \cdot \bar{E}^*_{inc}} (\bar{E}'_{scat} \cdot \bar{E}'^*_{scat}) \quad ,$$

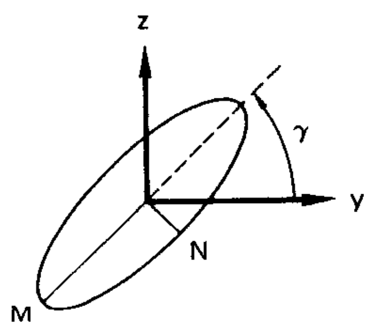
where W_{scat} is the scattered power per unit area at distance R in a given direction, W_{inc} is the power per unit area of the incident plane wave, and the primes on the electric fields specify the fields used in the program as defined above. For the case of a Hertzian dipole used as a source, the gain equations are used; however, P_{in} is equal to the total power radiated by the Hertzian source. That is

$$P_{in} = \frac{\pi\eta}{3} \left| \frac{I\ell}{\lambda} \right|^2 \quad ,$$

where the quantity $I\ell$ is an input quantity.

(6) Elliptic Polarization

Elliptic polarization parameters are calculated as follows:



$$M = [(E_{ym} \cos \gamma + E_{zm} \cos \xi \sin \gamma)^2 + E_{zm}^2 \sin^2 \xi \sin^2 \gamma]^{1/2} ,$$

$$N = [E_{ym} \sin \gamma - E_{zm} \cos \xi \cos \gamma)^2 + E_{zm}^2 \sin^2 \xi \cos^2 \gamma]^{1/2} ,$$

where

$$E_y = E_{ym} \exp[j(\omega t - kx)] ,$$

$$E_z = E_{zm} \exp[j(\omega t - kx + \xi)] ,$$

and γ is given by

$$\tan 2y = \frac{2E_{ym}E_{zm}\cos\xi}{E_{ym}^2 - E_{zm}^2}$$

In this routine, the coordinates y and z above are replaced by θ and ϕ, respectively.

The field is computed by FFLD at RD74 for space wave or by GFLD at RD76 for space and ground wave. Elliptic polarization parameters are computed from RD87 to RD118. RD127 to RD137 stores gain in the array GAIN for normalization. The integral of radiated power for the average gain calculation is summed at RD140 to RD147. Fields and gain are printed at RD162 for space wave or RD165 for ground wave. Average gain is computed and printed from RD168 to RD173. Normalized gain is printed from RD174 to RD208.

SYMBOL DICTIONARY

AXRAT	= N/M (elliptic axial ratio)
CHT	= height of cliff in meters
CLT	= distance in meters of cliff edge from origin
DA	= element of solid angle for average gain summation
DFAZ	= phase difference between E_θ and E_ϕ for elliptic polarization

DPH	= increment for ϕ
DTH	= increment for θ
EMAJR2	= M^2 (M = major axis)
EMINR2	= N^2
EPH	= E_ϕ (phi component of electric field, with or without the term exp(-jkR)/(R/λ) depending on return from GFLD or FFLD)
EPHA	= phase angle of EPH
EPHM	= |EPH|
EPHM2	= $\|EPH\|^2$
EPSR	= relative dielectric constant
EPSR2	= relative dielectric constant of second medium
ERD	= radial electric field for ground wave
ERDA	= phase of ERD
ERDM	= |ERD|
ETH	= E_θ
ETHA	= phase of E_θ
ETHM	= $\|E_\theta\|$
ETHM2	= $\|E_\theta\|^2$
EXRA	= phase of exp(-jkR)
EXRM	= 1/R
GCON	= factor multiplying $\|E\|^2$ to yield gain or σ/λ^2
GCOP	= GCON except when GCON yields directive gain; then GCOP remains power gain
GMAX	= value used for normalized gain
GNH	= horizontal gain in decibels, ϕ component
GNMJ	= major axis gain in decibels
GNMN	= minor axis gain in decibels
GNOR	= if non-zero, equals input gain quantity
GNV	= vertical gain (θ)
GTOT	= total gain
IAVP	= flag for average gain
IAX	= flag for gain type
IFAR	= first integer from RP card

INOR	=	integer to select normalized gain
IPD	=	flag to select power or directive gain
IXTYP	=	excitation type
NORMAX	=	dimension of FNORM (maximum number of gain values that will be stored for normalization)
NPH	=	number of ϕ values
NTH	=	number of θ values
PHA	=	ϕ in radians
PHI	=	ϕ in degrees
PHIS	=	initial ϕ
PI	=	π
PINR	=	input power for current element source
PINT	=	summation variable for average gain
PLOSS	=	power dissipated in structure loads
PNLR	=	power dissipated in networks and transmission lines
PRAD	=	power radiated by the antenna
RFLD	=	if non-zero, equal to the observation distance in meters
SIG	=	conductivity of ground (mhos/m)
SIG2	=	conductivity of second medium (mhos/m)
STILTA	=	sin γ; γ is tilt angle of the polarization ellipse
TA	=	$\pi/180$
TD	=	$180/\pi$
THA	=	θ in radians
THET	=	θ in degrees
THETS	=	initial θ
TILTA	=	γ (tilt angle of ellipse)
XPR6	=	minor axis of polarization ellipse or strength of current element source

CONSTANTS

1.745329252E-2	=	$\pi/180$
1.E-20	=	small value test

1.E-5	= small value test
-1.E10	= near minus infinity
3.141592654	= π
376.73	= $\eta_0 = \sqrt{\mu_0/\epsilon_0}$
394.51	= $\pi\eta_0/3$
57.2957795	= $180/\pi$
59.96	= $\eta_0/(2\pi)$
90.01	= test value for angle exceeding 90 degrees

```
      SUBROUTINE RDPAT
C     COMPUTE RADIATION PATTERN, GAIN, NORMALIZED GAIN
      INTEGER HPOL,HBLK,HCIR,HCLIF
      COMPLEX ETH,EPH,ERD,ZRATI,ZRATI2,T1,FRATI
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /SAVE/ IP(600),KCOM,COM(13,5),EPSR,SIG,SCRWLT,SCRWRT,FMHZ
      COMMON /GND/ZRATI,ZRATI2,FRATI,CL,CH,SCRWL,SCRWR,NRADL,KSYMP,IFAR,
     1IPERF,T1,T2
      COMMON /FPAT/ NTH,NPH,IPD,IAVP,INOR,IAX,THETS,PHIS,DTH,DPH,RFLD,GN
     1OR,CLT,CHT,EPSR2,SIG2,IXTYP,XPR6,PINR,PNLR,PLOSS,NEAR,NFEH,NRX,NRY
     2,NRZ,XNR,YNR,ZNR,DXNR,DYNR,DZNR
      COMMON /SCRATM/ GAIN(1200)
      DIMENSION IGTP(4), IGAX(4), IGNTP(10), HPOL(3)
      DATA HPOL/6HLINEAR,5HRIGHT,4HLEFT/,HBLK,HCIR/1H ,6HCIRCLE/
      DATA IGTP/6H    - ,6HPOWER ,6H- DIRE,6HCTIVE /
      DATA IGAX/6H MAJOR,6H MINOR,6H VERT.,6H HOR. /
      DATA IGNTP/6H MAJOR,6H AXIS ,6H MINOR,6H AXIS ,6H   VER,6HTICAL ,6
     1H HORIZ,6HONTAL ,6H      ,6HTOTAL /
      DATA PI,TA,TD/3.141592654,1.745329252E-02,57.29577951/
      DATA NORMAX/1200/
      IF (IFAR.LT.2) GO TO 2
      PRINT 35
      IF (IFAR.LE.3) GO TO 1
      PRINT 36, NRADL,SCRWLT,SCRWRT
      IF (IFAR.EQ.4) GO TO 2
1     IF (IFAR.EQ.2.OR.IFAR.EQ.5) HCLIF=HPOL(1)
      IF (IFAR.EQ.3.OR.IFAR.EQ.6) HCLIF=HCIR
      CL=CLT/WLAM
      CH=CHT/WLAM
      ZRATI2=CSQRT(1./CMPLX(EPSR2,-SIG2*WLAM*59.96))
      PRINT 37, HCLIF,CLT,CHT,EPSR2,SIG2
2     IF (IFAR.NE.1) GO TO 3
      PRINT 41
      GO TO 5
3     I=2*IPD+1
      J=I+1
      ITMP1=2*IAX+1
      ITMP2=ITMP1+1
      PRINT 38
      IF (RFLD.LT.1.E-20) GO TO 4
      EXRM=1./RFLD
      EXRA=RFLD/WLAM
      EXRA=-360.*(EXRA-AINT(EXRA))
      PRINT 39, RFLD,EXRM,EXRA
4     PRINT 40, IGTP(I),IGTP(J),IGAX(ITMP1),IGAX(ITMP2)
5     IF (IXTYP.EQ.0.OR.IXTYP.EQ.5) GO TO 7
      IF (IXTYP.EQ.4) GO TO 6
      PRAD=0.
      GCON=4.*PI/(1.+XPR6*XPR6)
      GCOP=GCON
      GO TO 8
6     PINR=394.51*XPR6*XPR6*WLAM*WLAM
7     GCOP=WLAM*WLAM*2.*PI/(376.73*PINR)
      PRAD=PINR-PLOSS-PNLR
      GCON=GCOP
      IF (IPD.NE.0) GCON=GCON*PINR/PRAD
8     I=0
      GMAX=-1.E10
      PINT=0.
      TMP1=DPH*TA
      TMP2=.5*DTH*TA
      PHI=PHIS-DPH
```

```
      DO 29 KPH=1,NPH
      PHI=PHI+DPH
      PHA=PHI*TA
      THET=THETS-DTH
      DO 29 KTH=1,NTH
      THET=THET+DTH
      IF (KSYMP.EQ.2.AND.THET.GT.90.01.AND.IFAR.NE.1) GO TO 29
      THA=THET*TA
      IF (IFAR.EQ.1) GO TO 9
      CALL FFLD (THA,PHA,ETH,EPH)
      GO TO 10
9     CALL GFLD (RFLD/WLAM,PHA,THET/WLAM,ETH,EPH,ERD,ZRATI,KSYMP)
      ERDM=CABS(ERD)
      ERDA=CANG(ERD)
10    ETHM2=REAL(ETH*CONJG(ETH))
      ETHM=SQRT(ETHM2)
      ETHA=CANG(ETH)
      EPHM2=REAL(EPH*CONJG(EPH))
      EPHM=SQRT(EPHM2)
      EPHA=CANG(EPH)
      IF (IFAR.EQ.1) GO TO 28
C     ELLIPTICAL POLARIZATION CALC.
      IF (ETHM2.GT.1.E-20.OR.EPHM2.GT.1.E-20) GO TO 11
      TILTA=0.
      EMAJR2=0.
      EMINR2=0.
      AXRAT=0.
      ISENS=HBLK
      GO TO 16
11    DFAZ=EPHA-ETHA
      IF (EPHA.LT.0.) GO TO 12
      DFAZ2=DFAZ-360.
      GO TO 13
12    DFAZ2=DFAZ+360.
13    IF (ABS(DFAZ).GT.ABS(DFAZ2)) DFAZ=DFAZ2
      CDFAZ=COS(DFAZ*TA)
      TSTOR1=ETHM2-EPHM2
      TSTOR2=2.*EPHM*ETHM*CDFAZ
      TILTA=.5*ATGN2(TSTOR2,TSTOR1)
      STILTA=SIN(TILTA)
      TSTOR1=TSTOR1*STILTA*STILTA
      TSTOR2=TSTOR2*STILTA*COS(TILTA)
      EMAJR2=-TSTOR1+TSTOR2+ETHM2
      EMINR2=TSTOR1-TSTOR2+EPHM2
      IF (EMINR2.LT.0.) EMINR2=0.
      AXRAT=SQRT(EMINR2/EMAJR2)
      TILTA=TILTA*TD
      IF (AXRAT.GT.1.E-5) GO TO 14
      ISENS=HPOL(1)
      GO TO 16
14    IF (DFAZ.GT.0.) GO TO 15
      ISENS=HPOL(2)
      GO TO 16
15    ISENS=HPOL(3)
16    GNMJ=DB10(GCON*EMAJR2)
      GNMN=DB10(GCON*EMINR2)
      GNV=DB10(GCON*ETHM2)
      GNH=DB10(GCON*EPHM2)
      GTOT=DB10(GCON*(ETHM2+EPHM2))
      IF (INOR.LT.1) GO TO 23
      I=I+1
      IF (I.GT.NORMAX) GO TO 23
      GO TO (17,18,19,20,21), INOR
17    TSTOR1=GNMJ
```

```
      GO TO 22
   18 TSTOR1=GNMN
      GO TO 22
   19 TSTOR1=GNV
      GO TO 22
   20 TSTOR1=GNH
      GO TO 22
   21 TSTOR1=GTOT
   22 GAIN(I)=TSTOR1
      IF (TSTOR1.GT.GMAX) GMAX=TSTOR1
   23 IF (IAVP.EQ.0) GO TO 24
      TSTOR1=GCOP*(ETHM2+EPHM2)
      TMP3=THA-TMP2
      TMP4=THA+TMP2
      IF (KTH.EQ.1) TMP3=THA
      IF (KTH.EQ.NTH) TMP4=THA
      DA=ABS(TMP1*(COS(TMP3)-COS(TMP4)))
      IF (KPH.EQ.1.OR.KPH.EQ.NPH) DA=.5*DA
      PINT=PINT+TSTOR1*DA
      IF (IAVP.EQ.2) GO TO 29
   24 IF (IAX.EQ.1) GO TO 25
      TMP5=GNMJ
      TMP6=GNMN
      GO TO 26
   25 TMP5=GNV
      TMP6=GNH
   26 ETHM=ETHM*WLAM
      EPHM=EPHM*WLAM
      IF (RFLD.LT.1.E-20) GO TO 27
      ETHM=ETHM*EXRM
      ETHA=ETHA+EXRA
      EPHM=EPHM*EXRM
      EPHA=EPHA+EXRA
   27 PRINT 42, THET,PHI,TMP5,TMP6,GTOT,AXRAT,TILTA,ISENS,ETHM,ETHA,EPHM
     1,EPHA
      GO TO 29
   28 PRINT 43, RFLD,PHI,THET,ETHM,ETHA,EPHM,EPHA,ERDM,ERDA
   29 CONTINUE
      IF (IAVP.EQ.0) GO TO 30
      TMP3=THETS*TA
      TMP4=TMP3+DTH*TA*FLOAT(NTH-1)
      TMP3=ABS(DPH*TA*FLOAT(NPH-1)*(COS(TMP3)-COS(TMP4)))
      PINT=PINT/TMP3
      TMP3=TMP3/PI
      PRINT 44, PINT,TMP3
   30 IF (INOR.EQ.0) GO TO 34
      IF (ABS(GNOR).GT.1.E-20) GMAX=GNOR
      ITMP1=(INOR-1)*2+1
      ITMP2=ITMP1+1
      PRINT 45, IGNTP(ITMP1),IGNTP(ITMP2),GMAX
      ITMP2=NPH*NTH
      IF (ITMP2.GT.NORMAX) ITMP2=NORMAX
      ITMP1=(ITMP2+2)/3
      ITMP2=ITMP1*3-ITMP2
      ITMP3=ITMP1
      ITMP4=2*ITMP1
      IF (ITMP2.EQ.2) ITMP4=ITMP4-1
      DO 31 I=1,ITMP1
      ITMP3=ITMP3+1
      ITMP4=ITMP4+1
      J=(I-1)/NTH
      TMP1=THETS+FLOAT(I-J*NTH-1)*DTH
      TMP2=PHIS+FLOAT(J)*DPH
      J=(ITMP3-1)/NTH
```

```
      TMP3=THETS+FLOAT(ITMP3-J*NTH-1)*DTH
      TMP4=PHIS+FLOAT(J)*DPH
      J=(ITMP4-1)/NTH
      TMP5=THETS+FLOAT(ITMP4-J*NTH-1)*DTH
      TMP6=PHIS+FLOAT(J)*DPH
      TSTOR1=GAIN(I)-GMAX
      IF (I.EQ.ITMP1.AND.ITMP2.NE.0) GO TO 32
      TSTOR2=GAIN(ITMP3)-GMAX
      PINT=GAIN(ITMP4)-GMAX
31    PRINT 46, TMP1,TMP2,TSTOR1,TMP3,TMP4,TSTOR2,TMP5,TMP6,PINT
      GO TO 34
32    IF (ITMP2.EQ.2) GO TO 33
      TSTOR2=GAIN(ITMP3)-GMAX
      PRINT 46, TMP1,TMP2,TSTOR1,TMP3,TMP4,TSTOR2
      GO TO 34
33    PRINT 46, TMP1,TMP2,TSTOR1
34    RETURN
C
35    FORMAT (///,31X,39H- - - FAR FIELD GROUND PARAMETERS - - -,//)
36    FORMAT (40X,25HRADIAL WIRE GROUND SCREEN,/,40X,I5,6H WIRES,/,40X,1
     12HWIRE LENGTH=,F8.2,7H METERS,/,40X,12HWIRE RADIUS=,E10.3,7H METER
     2S)
37    FORMAT (40X,A6,6H CLIFF,/,40X,14HEDGE DISTANCE=,F9.2,7H METERS,/,4
     10X,7HHEIGHT=,F8.2,7H METERS,/,40X,15HSECOND MEDIUM -,/,40X,27HRELA
     2TIVE DIELECTRIC CONST.=,F7.3,/,40X,13HCONDUCTIVITY=,E10.3,5H MHOS)
38    FORMAT (///,48X,30H- - - RADIATION PATTERNS - - -)
39    FORMAT (54X,6HRANGE=,E13.6,7H METERS,/,54X,12HEXP(-JKR)/R=,E12.5,9
     1H AT PHASE,F7.2,8H DEGREES,/)
40    FORMAT (/,2X,14H- - ANGLES - -,7X,2A6,7HGAINS -,7X,24H- - - POLARI
     1ZATION - - -,4X,20H- - - E(THETA) - - -,4X,16H- - - E(PHI) - -,2H
     2-,/,2X,5HTHETA,5X,3HPHI,7X,A6,2X,A6,3X,5HTOTAL,6X,5HAXIAL,5X,4HTIL
     3T,3X,5HSENSE,2(5X,9HMAGNITUDE,4X,6HPHASE ),/,2(1X,7HDEGREES,1X),3(
     46X,2HDB),8X,5HRATIO,5X,4HDEG.,8X,2(6X,7HVOLTS/M,4X,7HDEGREES))
41    FORMAT (///,28X,40H - - - RADIATED FIELDS NEAR GROUND - - -,//,8X,
     120H- - - LOCATION - - -,10X,16H- - E(THETA) - -,8X,14H- - E(PHI) -
     2 -,8X,17H- - E(RADIAL) - -,/,7X,3HRHO,6X,3HPHI,9X,1HZ,12X,3HMAG,6X
     3,5HPHASE,9X,3HMAG,6X,5HPHASE,9X,3HMAG,6X,5HPHASE,/,5X,6HMETERS,3X,
     47HDEGREES,4X,6HMETERS,8X,7HVOLTS/M,3X,7HDEGREES,6X,7HVOLTS/M,3X,7H
     5DEGREES,6X,7HVOLTS/M,3X,7HDEGREES,/)
42    FORMAT (1X,F7.2,F9.2,3X,3F8.2,F11.5,F9.2,2X,A6,2(E15.5,F9.2))
43    FORMAT (3X,F9.2,2X,F7.2,2X,F9.2,1X,3(3X,E11.4,2X,F7.2))
44    FORMAT (//,3X,19HAVERAGE POWER GAIN=,E12.5,7X, 31HSOLID ANGLE USED
     1 IN AVERAGING=(,F7.4,16H)*PI STERADIANS.,//)
45    FORMAT (//,37X,31H- - - - NORMALIZED GAIN - - - -,//,37X,2A6,4HGAI
     1N,/,38X,22HNORMALIZATION FACTOR =,F9.2,3H DB,//,3(4X,14H- - ANGLES
     2 - -,6X,4HGAIN,7X),/,3(4X,5HTHETA,5X,3HPHI,8X,2HDB,8X),/,3(3X,7HDE
     3GREES,2X,7HDEGREES,16X))
46    FORMAT (3(1X,2F9.2,1X,F9.2,6X))
      END
```

REBLK

PURPOSE

To read the matrix B by blocks of rows and write it by blocks of columns.

METHOD

When ICASX is 3 or 4 subroutine CMNGF writes B to file 14 by blocks of rows. Filling B by rows is convenient since the field of a single segment may contribute to several columns. However, blocks of columns are needed when $A^{-1}B$ is computed. Hence the format is converted.

NBBX is the number of block of B stored by rows and NBBL is the number of blocks stored by columns. The loop from RB16 to RB23 reads file 14 and stores the elements for block NPB of columns. This process is repeated for each of the NBBL blocks of columns.

SYMBOL DICTIONARY

- B = array for blocks of columns of B
- BX = array for blocks of rows of B
- N2C = number of columns in B
- NB = number of rows in B
- NBX = number of rows in blocks of rows of B (NPBX)
- NPB = number of columns in blocks of columns (NPBL or NLBL for last block)
- NPX = NPBX or NLBX for last block of rows

```
      SUBROUTINE REBLK (B,BX,NB,NBX,N2C)
C     REBLOCK ARRAY B IN N.G.F. SOLUTION FROM BLOCKS OF ROWS ON TAPE14
C     TO BLOCKS OF COLUMNS ON TAPE16
      COMPLEX B,BX
      COMMON /MATPAR/ ICASE,NBLOKS,NPBLK,NLAST,NBLSYM,NPSYM,NLSYM,IMAT,I
     1CASX,NBBX,NPBX,NLBX,NBBL,NPBL,NLBL
      DIMENSION B(NB,1), BX(NBX,1)
      REWIND 16
      NIB=0
      NPB=NPBL
      DO 3 IB=1,NBBL
      IF (IB.EQ.NBBL) NPB=NLBL
      REWIND 14
      NIX=0
      NPX=NPBX
      DO 2 IBX=1,NBBX
      IF (IBX.EQ.NBBX) NPX=NLBX
      READ (14) ((BX(I,J),I=1,NPX),J=1,N2C)
      DO 1 I=1,NPX
      IX=I+NIX
      DO 1 J=1,NPB
1     B(IX,J)=BX(I,J+NIB)
2     NIX=NIX+NPBX
      WRITE (16) ((B(I,J),I=1,NB),J=1,NPB)
3     NIB=NIB+NPBL
      REWIND 14
      REWIND 16
      RETURN
      END
```

REFLC

PURPOSE

To generate geometry data for structures having plane or cylindrical symmetry by forming symmetric images of a previously defined structure unit.

METHOD

The first part of the code, from statement RE20 to RE153, forms plane symmetric structures by reflecting segments and patches in the coordinate planes. The reflection planes are selected by the formal parameters IX, IY, and IZ. If IZ is greater than zero, an image of the existing segments and patches is formed by reflection in the x-y plane, which will be called reflection along the z axis. Next, if IY is greater than zero, an image of the existing segments and patches, including those generated in the previous step by reflection along the z axis, is formed by reflection along the y axis. Finally, if IX is greater than zero, an image of all segments and patches, including any previously formed by reflection along the z and y axes, is formed by reflection along the x axis. Any combination of zero and non-zero values of IX, IY, and IZ may be used to generate structures with one, two, or three planes of symmetry. Tag numbers of image segments are incremented by ITX from tags of the original segments, except that tags of zero are not incremented. After each reflection in a coordinate plane, ITX is doubled. Thus, if ITX is initially greater than the largest tag of the existing segments, no duplicate tags will be formed by reflection in one, two, or three planes.

The code from RE157 to RE204 forms cylindrically symmetric structures by forming images of previously defined segments and patches rotated about the z axis. The number of images, including the original structure, is selected by NOP in the formal parameters. The angle by which each image is rotated about the z axis from the previous image is computed as $2\pi/NOP$, so that the images are uniformly distributed about the z axis. Tag numbers of segments are incremented by ITX, except that tags of zero are not incremented.

When REFLC is used to form structures with either plane or cylindrical symmetry, the data in COMMON/DATA/ is set so that the program will take advantage of symmetry in filling and factoring the matrix. This is done by setting N equal to the total number of segments but leaving NP equal to the number of segments in the original structure unit that was reflected or rotated. The symmetry flag IPSYM is also set to indicate the type of symmetry: positive values indicating plane symmetry and negative values cylindrical symmetry. These symmetry conditions may later be changed if the structure is modified in such a way that symmetry is destroyed.

SYMBOL DICTIONARY

ABS = external routine (absolute value)
COS = external routine (cosine)
CS = cos (2π/NOP)
E1 = segment coordinate (temporary storage)
E2 = segment coordinate (temporary storage)
FNOP = NOP
I = DO loop index
ITAGI = segment tag (temporary storage)
ITI = segment tag increment
ITX = segment tag increment
IX = flag for reflection along x axis
IY = flag for reflection along y axis
IZ = flag for reflection along z axis
J = array location for new patch data
K = segment index and array location for old patch data
NOP = number of sections in cylindrically symmetric structure
NX = segment index and array location for new patch data
NNX = array location for old patch
SAM = 2π/NOP
SIN = external routine (sine)
SS = sin (2π/NOP)
T1X, T1Y, T1Z, T2X, T2Y, T2Z = x, y, z components of $\hat{t}_1$ and $\hat{t}_2$
XK = x coordinate of segment
X2(I) = x coordinate of end two of segment I
YK = y coordinate of segment

Y2(I) = y coordinate of end two of segment I

Z2(I) = z coordinate of end two of segment I

CONSTANTS

1.E-6 = tolerance in test for zero

1.E-5 = tolerance in test for zero

6.283185308 = 2π

```
      SUBROUTINE REFLC (IX,IY,IZ,ITX,NOP)
C
C     REFLC REFLECTS PARTIAL STRUCTURE ALONG X,Y, OR Z AXES OR ROTATES
C     STRUCTURE TO COMPLETE A SYMMETRIC STRUCTURE.
C
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /ANGL/ SALP(300)
      DIMENSION T1X(1), T1Y(1), T1Z(1), T2X(1), T2Y(1), T2Z(1), X2(1), Y
     12(1), Z2(1)
      EQUIVALENCE (T1X,SI), (T1Y,ALP), (T1Z,BET), (T2X,ICON1), (T2Y,ICON
     12), (T2Z,ITAG), (X2,SI), (Y2,ALP), (Z2,BET)
      NP=N
      MP=M
      IPSYM=0
      ITI=ITX
      IF (IX.LT.0) GO TO 19
      IF (NOP.EQ.0) RETURN
      IPSYM=1
      IF (IZ.EQ.0) GO TO 6
C
C     REFLECT ALONG Z AXIS
C
      IPSYM=2
      IF (N.LT.N2) GO TO 3
      DO 2 I=N2,N
      NX=I+N-N1
      E1=Z(I)
      E2=Z2(I)
      IF (ABS(E1)+ABS(E2).GT.1.E-5.AND.E1*E2.GE.-1.E-6) GO TO 1
      PRINT 24, I
      STOP
1     X(NX)=X(I)
      Y(NX)=Y(I)
      Z(NX)=-E1
      X2(NX)=X2(I)
      Y2(NX)=Y2(I)
      Z2(NX)=-E2
      ITAGI=ITAG(I)
      IF (ITAGI.EQ.0) ITAG(NX)=0
      IF (ITAGI.NE.0) ITAG(NX)=ITAGI+ITI
2     BI(NX)=BI(I)
      N=N*2-N1
      ITI=ITI*2
3     IF (M.LT.M2) GO TO 6
      NXX=LD+1-M1
      DO 5 I=M2,M
      NXX=NXX-1
      NX=NXX-M+M1
      IF (ABS(Z(NXX)).GT.1.E-10) GO TO 4
      PRINT 25, I
      STOP
4     X(NX)=X(NXX)
      Y(NX)=Y(NXX)
      Z(NX)=-Z(NXX)
      T1X(NX)=T1X(NXX)
      T1Y(NX)=T1Y(NXX)
      T1Z(NX)=-T1Z(NXX)
      T2X(NX)=T2X(NXX)
      T2Y(NX)=T2Y(NXX)
      T2Z(NX)=-T2Z(NXX)
      SALP(NX)=-SALP(NXX)
5     BI(NX)=BI(NXX)
```

```
      M=M*2-M1
    6 IF (IY.EQ.0) GO TO 12
C
C     REFLECT ALONG Y AXIS
C
      IF (N.LT.N2) GO TO 9
      DO 8 I=N2,N
      NX=I+N-N1
      E1=Y(I)
      E2=Y2(I)
      IF (ABS(E1)+ABS(E2).GT.1.E-5.AND.E1*E2.GE.-1.E-6) GO TO 7
      PRINT 24, I
      STOP
    7 X(NX)=X(I)
      Y(NX)=-E1
      Z(NX)=Z(I)
      X2(NX)=X2(I)
      Y2(NX)=-E2
      Z2(NX)=Z2(I)
      ITAGI=ITAG(I)
      IF (ITAGI.EQ.0) ITAG(NX)=0
      IF (ITAGI.NE.0) ITAG(NX)=ITAGI+ITI
    8 BI(NX)=BI(I)
      N=N*2-N1
      ITI=ITI*2
    9 IF (M.LT.M2) GO TO 12
      NXX=LD+1-M1
      DO 11 I=M2,M
      NXX=NXX-1
      NX=NXX-M+M1
      IF (ABS(Y(NXX)).GT.1.E-10) GO TO 10
      PRINT 25, I
      STOP
   10 X(NX)=X(NXX)
      Y(NX)=-Y(NXX)
      Z(NX)=Z(NXX)
      T1X(NX)=T1X(NXX)
      T1Y(NX)=-T1Y(NXX)
      T1Z(NX)=T1Z(NXX)
      T2X(NX)=T2X(NXX)
      T2Y(NX)=-T2Y(NXX)
      T2Z(NX)=T2Z(NXX)
      SALP(NX)=-SALP(NXX)
   11 BI(NX)=BI(NXX)
      M=M*2-M1
   12 IF (IX.EQ.0) GO TO 18
C
C     REFLECT ALONG X AXIS
C
      IF (N.LT.N2) GO TO 15
      DO 14 I=N2,N
      NX=I+N-N1
      E1=X(I)
      E2=X2(I)
      IF (ABS(E1)+ABS(E2).GT.1.E-5.AND.E1*E2.GE.-1.E-6) GO TO 13
      PRINT 24, I
      STOP
   13 X(NX)=-E1
      Y(NX)=Y(I)
      Z(NX)=Z(I)
      X2(NX)=-E2
      Y2(NX)=Y2(I)
      Z2(NX)=Z2(I)
      ITAGI=ITAG(I)
```

```
      IF (ITAGI.EQ.0) ITAG(NX)=0
      IF (ITAGI.NE.0) ITAG(NX)=ITAGI+ITI
   14 BI(NX)=BI(I)
      N=N*2-N1
   15 IF (M.LT.M2) GO TO 18
      NXX=LD+1-M1
      DO 17 I=M2,M
      NXX=NXX-1
      NX=NXX-M+M1
      IF (ABS(X(NXX)).GT.1.E-10) GO TO 16
      PRINT 25, I
      STOP
   16 X(NX)=-X(NXX)
      Y(NX)=Y(NXX)
      Z(NX)=Z(NXX)
      T1X(NX)=-T1X(NXX)
      T1Y(NX)=T1Y(NXX)
      T1Z(NX)=T1Z(NXX)
      T2X(NX)=-T2X(NXX)
      T2Y(NX)=T2Y(NXX)
      T2Z(NX)=T2Z(NXX)
      SALP(NX)=-SALP(NXX)
   17 BI(NX)=BI(NXX)
      M=M*2-M1
   18 RETURN
C
C     REPRODUCE STRUCTURE WITH ROTATION TO FORM CYLINDRICAL STRUCTURE
C
   19 FNOP=NOP
      IPSYM=-1
      SAM=6.283185308/FNOP
      CS=COS(SAM)
      SS=SIN(SAM)
      IF (N.LT.N2) GO TO 21
      N=N1+(N-N1)*NOP
      NX=NP+1
      DO 20 I=NX,N
      K=I-NP+N1
      XK=X(K)
      YK=Y(K)
      X(I)=XK*CS-YK*SS
      Y(I)=XK*SS+YK*CS
      Z(I)=Z(K)
      XK=X2(K)
      YK=Y2(K)
      X2(I)=XK*CS-YK*SS
      Y2(I)=XK*SS+YK*CS
      Z2(I)=Z2(K)
      ITAGI=ITAG(K)
      IF (ITAGI.EQ.0) ITAG(I)=0
      IF (ITAGI.NE.0) ITAG(I)=ITAGI+ITI
   20 BI(I)=BI(K)
   21 IF (M.LT.M2) GO TO 23
      M=M1+(M-M1)*NOP
      NX=MP+1
      K=LD+1-M1
      DO 22 I=NX,M
      K=K-1
      J=K-MP+M1
      XK=X(K)
      YK=Y(K)
      X(J)=XK*CS-YK*SS
      Y(J)=XK*SS+YK*CS
      Z(J)=Z(K)
```

```
      XK=T1X(K)                                                          RE 193
      YK=T1Y(K)                                                          RE 194
      T1X(J)=XK*CS-YK*SS                                                 RE 195
      T1Y(J)=XK*SS+YK*CS                                                 RE 196
      T1Z(J)=T1Z(K)                                                      RE 197
      XK=T2X(K)                                                          RE 198
      YK=T2Y(K)                                                          RE 199
      T2X(J)=XK*CS-YK*SS                                                 RE 200
      T2Y(J)=XK*SS+YK*CS                                                 RE 201
      T2Z(J)=T2Z(K)                                                      RE 202
      SALP(J)=SALP(K)                                                    RE 203
   22 BI(J)=BI(K)                                                        RE 204
   23 RETURN                                                             RE 205
C                                                                        RE 206
   24 FORMAT (29H GEOMETRY DATA ERROR--SEGMENT,I5,26H LIES IN PLANE OF S RE 207
     1YMMETRY)                                                           RE 208
   25 FORMAT (27H GEOMETRY DATA ERROR--PATCH,I4,26H LIES IN PLANE OF SYM RE 209
     1METRY)                                                             RE 210
      END                                                                RE 211-
```

ROM2

PURPOSE

To numerically integrate over the current distribution on a segment to obtain the field due to the Sommerfeld integral term.

METHOD

ROM2 integrates the product of $\bar{E}_s(\bar{r})$ (see discussion of EFLD) and the current over a segment. Separate integrals are evaluated for current distributions of constant, sin $k(s - s_0)$ and cos $k(s - s_0)$. With three vector components of the field, there are nine integrals evaluated simultaneously and stored in the array SUM. The integration method is the same as that described for subroutine INTX, but loops from one through nine are used at each step.

The parameter DMIN is set in EFLD to

$$\text{DMIN} = 0.01\ [|E_x'|^2 + |E_y'|^2 + |E_z'|^2]^{1/2}$$

$$\text{where } \bar{E}' = \int_{\text{segment}} [\bar{E}_D(\bar{r}) + \frac{k_1^2 - k_2^2}{k_1^2 + k_2^2} \bar{E}_I(\bar{r})]\ ds \quad .$$

DMIN is passed to TEST as the lower limit for the denominator in the relative error evaluation to avoid trying to maintain relative accuracy in integrating the Sommerfeld integral when it is much smaller than the other terms.

SYMBOL DICTIONARY

A	= lower limit of integral
B	= upper limit of integral
DMIN	= minimum for denominator in relative error test
DZ	= subinterval size
DZOT	= 0.5 DZ
EP	= tolerance for hitting upper limit

G1, G2, G3, G4, G5	= integrand values at points within the subinterval
N	= number of functions (9)
NM	= minimum subinterval size is (B - A)/NM
NS	= present subinterval size is (B - A)/NS
NT	= counter to control increasing subinterval size
NTS	= larger values retard increasing subinterval size
NX	= maximum subinterval size is (B - A)/NX
RX	= relative error limit
S	= B - A
SUM	= array for integral values
T00, T01, T02, T10, T11, T20	= (see subroutine INTX)
TMAG1, TMAG2	= sum of the magnitudes of the integral contributions for the constant current distribution
Z	= integration variable at left side of subinterval
ZE	= B
ZEND	= upper limit

CONSTANTS

1.E-4 = relative error criterion

65536 = limit for cutting subinterval size

```
      SUBROUTINE ROM2 (A,B,SUM,DMIN)
C
C     FOR THE SOMMERFELD GROUND OPTION, ROM2 INTEGRATES OVER THE SOURCE
C     SEGMENT TO OBTAIN THE TOTAL FIELD DUE TO GROUND.  THE METHOD OF
C     VARIABLE INTERVAL WIDTH ROMBERG INTEGRATION IS USED.  THERE ARE 9
C     FIELD COMPONENTS - THE X, Y, AND Z COMPONENTS DUE TO CONSTANT,
C     SINE, AND COSINE CURRENT DISTRIBUTIONS.
C
      COMPLEX SUM,G1,G2,G3,G4,G5,T00,T01,T10,T02,T11,T20
      DIMENSION SUM(9), G1(9), G2(9), G3(9), G4(9), G5(9), T01(9), T10(9
     1), T20(9)
      DATA NM,NTS,NX,N/65536,4,1,9/,RX/1.E-4/
      Z=A
      ZE=B
      S=B-A
      IF (S.GE.0.) GO TO 1
      PRINT 18
      STOP
1     EP=S/(1.E4*NM)
      ZEND=ZE-EP
      DO 2 I=1,N
2     SUM(I)=(0.,0.)
      NS=NX
      NT=0
      CALL SFLDS (Z,G1)
3     DZ=S/NS
      IF (Z+DZ.LE.ZE) GO TO 4
      DZ=ZE-Z
      IF (DZ.LE.EP) GO TO 17
4     DZOT=DZ*.5
      CALL SFLDS (Z+DZOT,G3)
      CALL SFLDS (Z+DZ,G5)
5     TMAG1=0.
      TMAG2=0.
C
C     EVALUATE 3 POINT ROMBERG RESULT AND TEST CONVERGENCE.
C
      DO 6 I=1,N
      T00=(G1(I)+G5(I))*DZOT
      T01(I)=(T00+DZ*G3(I))*.5
      T10(I)=(4.*T01(I)-T00)/3.
      IF (I.GT.3) GO TO 6
      TR=REAL(T01(I))
      TI=AIMAG(T01(I))
      TMAG1=TMAG1+TR*TR+TI*TI
      TR=REAL(T10(I))
      TI=AIMAG(T10(I))
      TMAG2=TMAG2+TR*TR+TI*TI
6     CONTINUE
      TMAG1=SQRT(TMAG1)
      TMAG2=SQRT(TMAG2)
      CALL TEST(TMAG1,TMAG2,TR,0.,0.,TI,DMIN)
      IF(TR.GT.RX)GO TO 8
      DO 7 I=1,N
7     SUM(I)=SUM(I)+T10(I)
      NT=NT+2
      GO TO 12
8     CALL SFLDS (Z+DZ*.25,G2)
      CALL SFLDS (Z+DZ*.75,G4)
      TMAG1=0.
      TMAG2=0.
C
C     EVALUATE 5 POINT ROMBERG RESULT AND TEST CONVERGENCE.
C
```

```
      DO 9 I=1,N
      T02=(T01(I)+DZOT*(G2(I)+G4(I)))*.5
      T11=(4.*T02-T01(I))/3.
      T20(I)=(16.*T11-T10(I))/15.
      IF (I.GT.3) GO TO 9
      TR=REAL(T11)
      TI=AIMAG(T11)
      TMAG1=TMAG1+TR*TR+TI*TI
      TR=REAL(T20(I))
      TI=AIMAG(T20(I))
      TMAG2=TMAG2+TR*TR+TI*TI
    9 CONTINUE
      TMAG1=SQRT(TMAG1)
      TMAG2=SQRT(TMAG2)
      CALL TEST(TMAG1,TMAG2,TR,0.,0.,TI,DMIN)
      IF(TR.GT.RX)GO TO 14
   10 DO 11 I=1,N
   11 SUM(I)=SUM(I)+T20(I)
      NT=NT+1
   12 Z=Z+DZ
      IF (Z.GT.ZEND) GO TO 17
      DO 13 I=1,N
   13 G1(I)=G5(I)
      IF (NT.LT.NTS.OR.NS.LE.NX) GO TO 3
      NS=NS/2
      NT=1
      GO TO 3
   14 NT=0
      IF (NS.LT.NM) GO TO 15
      PRINT 19, Z
      GO TO 10
   15 NS=NS*2
      DZ=S/NS
      DZOT=DZ*.5
      DO 16 I=1,N
      G5(I)=G3(I)
   16 G3(I)=G2(I)
      GO TO 5
   17 CONTINUE
      RETURN
C
   18 FORMAT (30H ERROR - B LESS THAN A IN ROM2)
   19 FORMAT (33H ROM2 -- STEP SIZE LIMITED AT Z =,E12.5)
      END
```

SBF

PURPOSE

To evaluate the current expansion function associated with a given segment, returning only that portion on a particular segment.

METHOD

SBF is very similar to routine TBF. Both routines evaluate the current expansion functions. However, while TBF stores the coefficients for each segment on which a given expansion function is non-zero, SBF returns the coefficients for only a single specified segment.

In the call to SBF, I is the segment on which the expansion function is centered. IS is the segment for which the function coefficients A_j, B_j and C_j are requested. These coefficients are returned in AA, BB, CC, respectively.

Refer to TBF for a discussion of the coding and variables. One additional variable in SBF -- JUNE -- is set to -1 or +1 if segment IS is found connected to end 1 or end 2, respectively, of segment I. If I = IS and segment I is not connected to a surface or ground plane, then JUNE is set to 0.

```
      SUBROUTINE SBF (I,IS,AA,BB,CC)                                    SB   1
C     COMPUTE COMPONENT OF BASIS FUNCTION I ON SEGMENT IS.              SB   2
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300 SB   3
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX( SB   4
     2300),WLAM,IPSYM                                                   SB   5
      DATA PI/3.141592654/,JMAX/30/                                     SB   6
      AA=0.                                                             SB   7
      BB=0.                                                             SB   8
      CC=0.                                                             SB   9
      JUNE=0                                                            SB  10
      JSNO=0                                                            SB  11
      PP=0.                                                             SB  12
      JCOX=ICON1(I)                                                     SB  13
      IF (JCOX.GT.10000) JCOX=I                                         SB  14
      JEND=-1                                                           SB  15
      IEND=-1                                                           SB  16
      SIG=-1.                                                           SB  17
      IF (JCOX) 1,11,2                                                  SB  18
1     JCOX=-JCOX                                                        SB  19
      GO TO 3                                                           SB  20
2     SIG=-SIG                                                          SB  21
      JEND=-JEND                                                        SB  22
3     JSNO=JSNO+1                                                       SB  23
      IF (JSNO.GE.JMAX) GO TO 24                                        SB  24
      D=PI*SI(JCOX)                                                     SB  25
      SDH=SIN(D)                                                        SB  26
      CDH=COS(D)                                                        SB  27
      SD=2.*SDH*CDH                                                     SB  28
      IF (D.GT.0.015) GO TO 4                                           SB  29
      OMC=4.*D*D                                                        SB  30
      OMC=((1.3888889E-3*OMC-4.1666666667E-2)*OMC+.5)*OMC               SB  31
      GO TO 5                                                           SB  32
4     OMC=1.-CDH*CDH+SDH*SDH                                            SB  33
5     AJ=1./(ALOG(1./(PI*BI(JCOX)))-.577215664)                         SB  34
      PP=PP-OMC/SD*AJ                                                   SB  35
      IF (JCOX.NE.IS) GO TO 6                                           SB  36
      AA=AJ/SD*SIG                                                      SB  37
      BB=AJ/(2.*CDH)                                                    SB  38
      CC=-AJ/(2.*SDH)*SIG                                               SB  39
      JUNE=IEND                                                         SB  40
6     IF (JCOX.EQ.I) GO TO 9                                            SB  41
      IF (JEND.EQ.1) GO TO 7                                            SB  42
      JCOX=ICON1(JCOX)                                                  SB  43
      GO TO 8                                                           SB  44
7     JCOX=ICON2(JCOX)                                                  SB  45
8     IF (IABS(JCOX).EQ.I) GO TO 10                                     SB  46
      IF (JCOX) 1,24,2                                                  SB  47
9     IF (JCOX.EQ.IS) BB=-BB                                            SB  48
10    IF (IEND.EQ.1) GO TO 12                                           SB  49
11    PM=-PP                                                            SB  50
      PP=0.                                                             SB  51
      NJUN1=JSNO                                                        SB  52
      JCOX=ICON2(I)                                                     SB  53
      IF (JCOX.GT.10000) JCOX=I                                         SB  54
      JEND=1                                                            SB  55
      IEND=1                                                            SB  56
      SIG=-1.                                                           SB  57
      IF (JCOX) 1,12,2                                                  SB  58
12    NJUN2=JSNO-NJUN1                                                  SB  59
      D=PI*SI(I)                                                        SB  60
      SDH=SIN(D)                                                        SB  61
      CDH=COS(D)                                                        SB  62
      SD=2.*SDH*CDH                                                     SB  63
      CD=CDH*CDH-SDH*SDH                                                SB  64
```

```
      IF (D.GT.0.015) GO TO 13
      OMC=4.*D*D
      OMC=((1.3888889E-3*OMC-4.1666666667E-2)*OMC+.5)*OMC
      GO TO 14
   13 OMC=1.-CD
   14 AP=1./(ALOG(1./(PI*BI(I)))-.577215664)
      AJ=AP
      IF (NJUN1.EQ.0) GO TO 19
      IF (NJUN2.EQ.0) GO TO 21
      QP=SD*(PM*PP+AJ*AP)+CD*(PM*AP-PP*AJ)
      QM=(AP*OMC-PP*SD)/QP
      QP=-(AJ*OMC+PM*SD)/QP
      IF (JUNE) 15,18,16
   15 AA=AA*QM
      BB=BB*QM
      CC=CC*QM
      GO TO 17
   16 AA=-AA*QP
      BB=BB*QP
      CC=-CC*QP
   17 IF (I.NE.IS) RETURN
   18 AA=AA-1.
      BB=BB+(AJ*QM+AP*QP)*SDH/SD
      CC=CC+(AJ*QM-AP*QP)*CDH/SD
      RETURN
   19 IF (NJUN2.EQ.0) GO TO 23
      QP=PI*BI(I)
      XXI=QP*QP
      XXI=QP*(1.-.5*XXI)/(1.-XXI)
      QP=-(OMC+XXI*SD)/(SD*(AP+XXI*PP)+CD*(XXI*AP-PP))
      IF (JUNE.NE.1) GO TO 20
      AA=-AA*QP
      BB=BB*QP
      CC=-CC*QP
      IF (I.NE.IS) RETURN
   20 AA=AA-1.
      D=CD-XXI*SD
      BB=BB+(SDH+AP*QP*(CDH-XXI*SDH))/D
      CC=CC+(CDH+AP*QP*(SDH+XXI*CDH))/D
      RETURN
   21 QM=PI*BI(I)
      XXI=QM*QM
      XXI=QM*(1.-.5*XXI)/(1.-XXI)
      QM=(OMC+XXI*SD)/(SD*(AJ-XXI*PM)+CD*(PM+XXI*AJ))
      IF (JUNE.NE.-1) GO TO 22
      AA=AA*QM
      BB=BB*QM
      CC=CC*QM
      IF (I.NE.IS) RETURN
   22 AA=AA-1.
      D=CD-XXI*SD
      BB=BB+(AJ*QM*(CDH-XXI*SDH)-SDH)/D
      CC=CC+(CDH-AJ*QM*(SDH+XXI*CDH))/D
      RETURN
   23 AA=-1.
      QP=PI*BI(I)
      XXI=QP*QP
      XXI=QP*(1.-.5*XXI)/(1.-XXI)
      CC=1./(CDH-XXI*SDH)
      RETURN
   24 PRINT 25, I
      STOP
C
   25 FORMAT (43H SBF - SEGMENT CONNECTION ERROR FOR SEGMENT,I5)
```

129 END

SECOND

PURPOSE

To obtain the time in seconds

METHOD

This subroutine acts as an interface of the computer system's time function and the NEC program. The system time function is called, the number is converted to seconds, and returned to the NEC program through the argument of subroutine SECOND. On CDC 6000 series computers, the system time function is SECOND and is called by the NEC program. This subroutine is, therefore, omitted on CDC 6000 computers.

CODE LISTING

```
      SUBROUTINE SECOND (T)                                              SE   1
```

Call system time function and set T equal to time in seconds.

```
      RETURN                                                             SE   9
      END                                                                SE  10-
```

SFLDS

PURPOSE

To evaluate the Sommerfeld-integral field components due to an infinitesimal current element on a segment.

METHOD

The coordinates of the segment are stored in COMMON/DATAJ/. The current element, at a distance T from the center of the segment is located at (XT, YT, ZT). From SL16 to SL42 the ρ, ϕ and z coordinates of the field evaluation point (XO, YO, ZO) are computed in a coordinate system with the z axis passing through the current element and ϕ = 0 in the direction of the segment reference direction projected on the x,y plane. R2 is as shown in Figure 6 (page 160) and is the same as R1 in Section IV of Part I.

The Sommerfeld-integral field is computed from SL85 to SL111 by giving R2 and θ', with

$$\theta' = \tan^{-1}\left(\frac{z + z'}{\rho}\right) ,$$

to subroutine INTRP. INTRP returns the quantities in equations 156 through 159 of Part I as

$$ERV = I_\rho^V$$

$$EZV = I_Z^V$$

$$ERH = I_\rho^H$$

$$EPH = I_\phi^H$$

These quantities are then multiplied by $\exp(-jkR_2)/R_2$. The components for a horizontal current element are multiplied by the appropriate factors of $\sin\phi$ or $\cos\phi$ and combined with the components for a vertical current element according to the elevation angle of the segment. Thus lines SL94 to SL96 are the ρ, z and ϕ components of the field of the current element. These are converted to x, y and z components and stored in E(1), E(2) and

E(3). They are also multiplied by sin(kT) and cos(kT) for the sine and cosine current distributions and stored in other elements of E.

When the separation of the source segment and observation point is large enough that the Norton approximation is used for the field, the code from SL49 to SL80 is executed. In this case SFLDS is called directly by EFLD, with T equal to zero, and returns an approximation to the field of the whole segment. The current is lumped at the center for a point source approximation.

GWAVE computes the total field including direct field and the asymptotic approximation of the field due to ground. Since EFLD has already computed

$$\bar{E}_D(\bar{r}) + \frac{k_1^2 - k_2^2}{k_1^2 + k_2^2}\,\bar{E}_I(\bar{r})$$

these terms must be removed from the field computed by GWAVE. The direct field $\bar{E}_D$ is set to zero by setting XX1 to zero before calling GWAVE. The second term is substracted from the field returned by GWAVE from SL59 to SL63. The field components of a vertical (V) and horizontal (H) current element in the direction $\phi = 0$ at the image point are

$$E_\rho^V = (E_R + E_T)\sin\theta\cos\theta$$

$$E_Z^V = E_R\cos^2\theta - E_T\sin^2\theta$$

$$E_\rho^H = (E_R\sin^2\theta - E_T\cos^2\theta)\cos\phi$$

$$E_Z^H = (E_R + E_T)\sin\theta\cos\theta\cos\phi$$

$$E_\phi^H = E_T\sin\phi$$

where

$$E_R = \frac{-j\eta}{4\pi^2}\,\frac{\exp(-jkR_2)}{(R_2/\lambda)^3}\,(1 + jkR_2)$$

$$E_T = \frac{-j\eta}{8\pi^2} \frac{\exp(-jkR_2)}{(R_2/\lambda)^3} (1 - k^2R_2^2 + jkR_2)$$

$$\cos \theta = (z + z')/R_2$$

$$\sin \theta = \rho/R_2$$

and current moment, $I\ell/\lambda^2 = 1$.

The sin ϕ and cos ϕ factors are omitted to match the quantities returned by GWAVE. Also, the fields of the horizontal current are reversed since the image of the source is in the direction ϕ = 180 degrees. These quantities are multiplied by FRATI and subtracted from the fields returned by GWAVE.

The total field, in x, y and z components, is stored from SL70 to SL72. S is the length of the segment in wavelengths. Hence it is $I\ell/\lambda^2$ when $I/\lambda = 1$. The current moment for a sine distribution is zero and for a cosine distribution is $\sin(\pi S)/\pi$.

SYMBOL DICTIONARY

CPH	= cos ϕ
E	= array for returning field components
EPH	= E_ϕ^H or I_ϕ^H
ER	= E_R
ERH	= E_ρ^H or I_ρ^H
ERV	= E_ρ^V or I_ρ^V
ET	= E_T
EZH	= E_Z^H or I_Z^H
EZV	= E_Z^V or I_Z^V
FRATI	= $(k_1^2 - k_2^2)/(k_1^2 + k_2^2)$
HRH	= E_ρ^H for image of source current element

SFLDS

HRV	=	E_ρ^V
HZV	=	H_Z^V
PHX	=	x component of $\hat{\phi}$
PHY	=	y component of $\hat{\phi}$
PI	=	π
POT	=	$\pi/2$
R1	=	direct distance to source (set to arbitrary value)
R2	=	distance to image
R2S	=	$(R2)^2$
RHØ	=	ρ
RHS	=	ρ^2
RHX	=	x component of ρ
RHY	=	y component of ρ
RK	=	kR_2
SFAC	=	value of current or current moment
SPH	=	$\sin \phi$
T	=	distance from center of segment to current element
THET	=	θ'
TP	=	2π
XT, YT, ZT	=	coordinates of current element
ZPHS	=	$(z + z')^2$

CONSTANTS

1.570796327	=	$\pi/2$
3.141592654	=	π
6.283185308	=	2π

```
      SUBROUTINE SFLDS (T,E)
C
C     SFLDX RETURNS THE FIELD DUE TO GROUND FOR A CURRENT ELEMENT ON
C     THE SOURCE SEGMENT AT T RELATIVE TO THE SEGMENT CENTER.
C
      COMPLEX E,ERV,EZV,ERH,EZH,EPH,T1,EXK,EYK,EZK,EXS,EYS,EZS,EXC,EYC,E
     1ZC,XX1,XX2,U,U2,ZRATI,ZRATI2,FRATI,ER,ET,HRV,HZV,HRH
      COMMON /DATAJ/ S,B,XJ,YJ,ZJ,CABJ,SABJ,SALPJ,EXK,EYK,EZK,EXS,EYS,EZ
     1S,EXC,EYC,EZC,RKH,IEXK,IND1,IND2,IPGND
      COMMON /INCOM/ XO,YO,ZO,SN,XSN,YSN,ISNOR
      COMMON /GWAV/ U,U2,XX1,XX2,R1,R2,ZMH,ZPH
      COMMON /GND/ZRATI,ZRATI2,FRATI,CL,CH,SCRWL,SCRWR,NRADL,KSYMP,IFAR,
     1IPERF,T1,T2
      DIMENSION E(9)
      DATA PI/3.141592654/,TP/6.283185308/,POT/1.570796327/
      XT=XJ+T*CABJ
      YT=YJ+T*SABJ
      ZT=ZJ+T*SALPJ
      RHX=XO-XT
      RHY=YO-YT
      RHS=RHX*RHX+RHY*RHY
      RHO=SQRT(RHS)
      IF (RHO.GT.0.) GO TO 1
      RHX=1.
      RHY=0.
      PHX=0.
      PHY=1.
      GO TO 2
1     RHX=RHX/RHO
      RHY=RHY/RHO
      PHX=-RHY
      PHY=RHX
2     CPH=RHX*XSN+RHY*YSN
      SPH=RHY*XSN-RHX*YSN
      IF (ABS(CPH).LT.1.E-10) CPH=0.
      IF (ABS(SPH).LT.1.E-10) SPH=0.
      ZPH=ZO+ZT
      ZPHS=ZPH*ZPH
      R2S=RHS+ZPHS
      R2=SQRT(R2S)
      RK=R2*TP
      XX2=CMPLX(COS(RK),-SIN(RK))
      IF (ISNOR.EQ.1) GO TO 3
C
C     USE NORTON APPROXIMATION FOR FIELD DUE TO GROUND.  CURRENT IS
C     LUMPED AT SEGMENT CENTER WITH CURRENT MOMENT FOR CONSTANT, SINE,
C     OR COSINE DISTRIBUTION.
C
      ZMH=1.
      R1=1.
      XX1=0.
      CALL GWAVE (ERV,EZV,ERH,EZH,EPH)
      ET=-(0.,4.77134)*FRATI*XX2/(R2S*R2)
      ER=2.*ET*CMPLX(1.,RK)
      ET=ET*CMPLX(1.-RK*RK,RK)
      HRV=(ER+ET)*RHO*ZPH/R2S
      HZV=(ZPHS*ER-RHS*ET)/R2S
      HRH=(RHS*ER-ZPHS*ET)/R2S
      ERV=ERV-HRV
      EZV=EZV-HZV
      ERH=ERH+HRH
      EZH=EZH+HRV
      EPH=EPH+ET
      ERV=ERV*SALPJ
```

```
      EZV=EZV*SALPJ                                                     SL  65
      ERH=ERH*SN*CPH                                                    SL  66
      EZH=EZH*SN*CPH                                                    SL  67
      EPH=EPH*SN*SPH                                                    SL  68
      ERH=ERV+ERH                                                       SL  69
      E(1)=(ERH*RHX+EPH*PHX)*S                                          SL  70
      E(2)=(ERH*RHY+EPH*PHY)*S                                          SL  71
      E(3)=(EZV+EZH)*S                                                  SL  72
      E(4)=0.                                                           SL  73
      E(5)=0.                                                           SL  74
      E(6)=0.                                                           SL  75
      SFAC=PI*S                                                         SL  76
      SFAC=SIN(SFAC)/SFAC                                               SL  77
      E(7)=E(1)*SFAC                                                    SL  78
      E(8)=E(2)*SFAC                                                    SL  79
      E(9)=E(3)*SFAC                                                    SL  80
      RETURN                                                            SL  81
C                                                                       SL  82
C     INTERPOLATE IN SOMMERFELD FIELD TABLES                            SL  83
C                                                                       SL  84
3     IF (RHO.LT.1.E-12) GO TO 4                                        SL  85
      THET=ATAN(ZPH/RHO)                                                SL  86
      GO TO 5                                                           SL  87
4     THET=POT                                                          SL  88
5     CALL INTRP (R2,THET,ERV,EZV,ERH,EPH)                              SL  89
C     COMBINE VERTICAL AND HORIZONTAL COMPONENTS AND CONVERT TO X,Y,Z   SL  90
C     COMPONENTS.  MULTIPLY BY EXP(-JKR)/R.                             SL  91
      XX2=XX2/R2                                                        SL  92
      SFAC=SN*CPH                                                       SL  93
      ERH=XX2*(SALPJ*ERV+SFAC*ERH)                                      SL  94
      EZH=XX2*(SALPJ*EZV-SFAC*ERV)                                      SL  95
      EPH=SN*SPH*XX2*EPH                                                SL  96
C     X,Y,Z FIELDS FOR CONSTANT CURRENT                                 SL  97
      E(1)=ERH*RHX+EPH*PHX                                              SL  98
      E(2)=ERH*RHY+EPH*PHY                                              SL  99
      E(3)=EZH                                                          SL 100
      RK=TP*T                                                           SL 101
C     X,Y,Z FIELDS FOR SINE CURRENT                                     SL 102
      SFAC=SIN(RK)                                                      SL 103
      E(4)=E(1)*SFAC                                                    SL 104
      E(5)=E(2)*SFAC                                                    SL 105
      E(6)=E(3)*SFAC                                                    SL 106
C     X,Y,Z FIELDS FOR COSINE CURRENT                                   SL 107
      SFAC=COS(RK)                                                      SL 108
      E(7)=E(1)*SFAC                                                    SL 109
      E(8)=E(2)*SFAC                                                    SL 110
      E(9)=E(3)*SFAC                                                    SL 111
      RETURN                                                            SL 112
      END                                                               SL 113-
```

SOLGF

PURPOSE

To solve for the basis function amplitudes in the NGF procedure.

METHOD

The operations performed here are described in the NGF overview in Section VI. SOLGF is called for either a NGF solution or a normal solution. For the normal solution, or for a NGF solution when no new segments or patches have been added, the solution is obtained by calling SOLVES at SF14. Otherwise, the rest of the code is executed.

The excitation vector XY is filled in the subroutine ETMNS in the order

1. E on NGF segments (N1 elements)
2. E on new segments (N - N1 elements)
3. H on NGF patches (2M1 elements)
4. H on new patches (2M - 2M1 elements)

rom SF18 to SF29 this vector is put in the order

$$\left.\begin{array}{l}\text{1. E on NGF segments}\\ \text{2. H on NGF patches}\end{array}\right\} E_1$$

$$\left.\begin{array}{l}\text{3. E on new segments}\\ \text{4. H on new patches}\end{array}\right\} E_2$$

o conform to the matrix structure. From SF30 to SF36, zeros are stored in XY n the locations opposite the rows of the C' matrix. Line SF37 then computes $^{-1}E_1$ storing it in place of E_1.

SF41 to SF52 computes $E_2 - C\,A^{-1}E_1$ and stores it in palce of $_2$. Matrix C is read from file 15 if necessary to form the product with $^{-1}E_1$. From SF55 to SF80

$$I_2 = [D - CA^{-1}B]^{-1}[E_2 - CA^{-1}E_1]$$

is computed in the original location of E_2. If ICASX is 4 the block parameters for the primary matrix are temporarily changed to those of $D - CA^{-1}B$ so that LTSOLV, which uses the primary block parameters, can perform the solution procedure. From SF84 to SF95

$$I_1 = A^{-1}E_1 - (A^{-1}B)I_2$$

is computed. The reordering step at the beginning of SOLGF is then reversed from SF98 to SF107 to put the solution vector in the order

1. amplitudes of NGF basis functions
2. amplitudes of new basis functions
3. NGF patch currents
4. new patch currents
5. amplitudes of modified basis functions for NGF segments that connect to new segments
6. meaningless values associated with B'_{ss}

Finally, from SF109 to SF113 the amplitudes of the modified basis functions are stored in place of the NGF basis functions that were set to zero.

SYMBOL DICTIONARY

A	=	array for matrix A_F
B	=	array starting just after A in CM (used for factoring $D - CA^{-1}B$ for ICASX = 2, 3 or 4)
C	=	array for matrix C
D	=	array used for factoring $D - CA^{-1}B$ when ICASX = 1
ICASS	=	saved value of ICASE
IFL	=	file in which blocks of A_F are stored in descending order (ascending order is always on 13)
IP	=	array of pivot element indices
M	=	number of patches

M1	= number of patches in NGF
MP	= number of patches in one symmetric section of the NGF structure
N	= number of segments
N1	= number of segments in NGF
N1C	= number of unknowns in NGF (N1 + 2M1)
N2	= N1 + 1
N2C	= number of new unknowns (order of D)
NBLSYS	= saved value of NBLSYM
NEQ	= total number of unknowns (NGF and new)
NEQS	= number of columns in B'_{sw} and B'_{ss}
NLSYS	= saved value of NLSYM
NP	= number of segments in a symmetric section of the NGF structure
NPSYS	= saved value of NPSYM
SUM	= summation variable for matrix products
XY	= excitation and solution vector

```
      SUBROUTINE SOLGF (A,B,C,D,XY,IP,NP,N1,N,MP,M1,M,N1C,N2C)          SF   1
C     SOLVE FOR CURRENT IN N.G.F. PROCEDURE                             SF   2
      COMPLEX A,B,C,D,SUM,XY,Y                                          SF   3
      COMMON /SCRATM/ Y(600)                                            SF   4
      COMMON /SEGJ/ AX(30),BX(30),CX(30),JCO(30),JSNO,ISCON(50),NSCON,IP SF   5
     1CON(10),NPCON                                                     SF   6
      COMMON /MATPAR/ ICASE,NBLOKS,NPBLK,NLAST,NBLSYM,NPSYM,NLSYM,IMAT,I SF   7
     1CASX,NBBX,NPBX,NLBX,NBBL,NPBL,NLBL                                SF   8
      DIMENSION B(N1C,1), C(N1C,1), D(N2C,1), IP(1), XY(1)              SF   9
      IFL=14                                                            SF  10
      IF (ICASX.GT.0) IFL=13                                            SF  11
      IF (N2C.GT.0) GO TO 1                                             SF  12
C     NORMAL SOLUTION.  NOT N.G.F.                                      SF  13
      CALL SOLVES (A,IP,XY,N1C,1,NP,N,MP,M,13,IFL)                      SF  14
      GO TO 22                                                          SF  15
1     IF (N1.EQ.N.OR.M1.EQ.0) GO TO 5                                   SF  16
C     REORDER EXCITATION ARRAY                                          SF  17
      N2=N1+1                                                           SF  18
      JJ=N+1                                                            SF  19
      NPM=N+2*M1                                                        SF  20
      DO 2 I=N2,NPM                                                     SF  21
2     Y(I)=XY(I)                                                        SF  22
      J=N1                                                              SF  23
      DO 3 I=JJ,NPM                                                     SF  24
      J=J+1                                                             SF  25
3     XY(J)=Y(I)                                                        SF  26
      DO 4 I=N2,N                                                       SF  27
      J=J+1                                                             SF  28
4     XY(J)=Y(I)                                                        SF  29
5     NEQS=NSCON+2*NPCON                                                SF  30
      IF (NEQS.EQ.0) GO TO 7                                            SF  31
      NEQ=N1C+N2C                                                       SF  32
      NEQS=NEQ-NEQS+1                                                   SF  33
C     COMPUTE INV(A)E1                                                  SF  34
      DO 6 I=NEQS,NEQ                                                   SF  35
6     XY(I)=(0.,0.)                                                     SF  36
7     CALL SOLVES (A,IP,XY,N1C,1,NP,N1,MP,M1,13,IFL)                    SF  37
      NI=0                                                              SF  38
      NPB=NPBL                                                          SF  39
C     COMPUTE E2-C(INV(A)E1)                                            SF  40
      DO 10 JJ=1,NBBL                                                   SF  41
      IF (JJ.EQ.NBBL) NPB=NLBL                                          SF  42
      IF (ICASX.GT.1) READ (15) ((C(I,J),I=1,N1C),J=1,NPB)              SF  43
      II=N1C+NI                                                         SF  44
      DO 9 I=1,NPB                                                      SF  45
      SUM=(0.,0.)                                                       SF  46
      DO 8 J=1,N1C                                                      SF  47
8     SUM=SUM+C(J,I)*XY(J)                                              SF  48
      J=II+I                                                            SF  49
9     XY(J)=XY(J)-SUM                                                   SF  50
10    NI=NI+NPBL                                                        SF  51
      REWIND 15                                                         SF  52
      JJ=N1C+1                                                          SF  53
C     COMPUTE INV(D)(E2-C(INV(A)E1)) = I2                               SF  54
      IF (ICASX.GT.1) GO TO 11                                          SF  55
      CALL SOLVE (N2C,D,IP(JJ),XY(JJ),N2C)                              SF  56
      GO TO 13                                                          SF  57
11    IF (ICASX.EQ.4) GO TO 12                                          SF  58
      NI=N2C*N2C                                                        SF  59
      READ (11) (B(J,1),J=1,NI)                                         SF  60
      REWIND 11                                                         SF  61
      CALL SOLVE (N2C,B,IP(JJ),XY(JJ),N2C)                              SF  62
      GO TO 13                                                          SF  63
12    NBLSYS=NBLSYM                                                     SF  64
```

```
      NPSYS=NPSYM
      NLSYS=NLSYM
      ICASS=ICASE
      NBLSYM=NBBL
      NPSYM=NPBL
      NLSYM=NLBL
      ICASE=3
      REWIND 11
      REWIND 16
      CALL LTSOLV (B,N2C,IP(JJ),XY(JJ),N2C,1,11,16)
      REWIND 11
      REWIND 16
      NBLSYM=NBLSYS
      NPSYM=NPSYS
      NLSYM=NLSYS
      ICASE=ICASS
 13   NI=0
      NPB=NPBL
C     COMPUTE INV(A)E1-(INV(A)B)I2 = I1
      DO 16 JJ=1,NBBL
      IF (JJ.EQ.NBBL) NPB=NLBL
      IF (ICASX.GT.1) READ (14) ((B(I,J),I=1,N1C),J=1,NPB)
      II=N1C+NI
      DO 15 I=1,N1C
      SUM=(0.,0.)
      DO 14 J=1,NPB
      JP=II+J
 14   SUM=SUM+B(I,J)*XY(JP)
 15   XY(I)=XY(I)-SUM
 16   NI=NI+NPBL
      REWIND 14
      IF (N1.EQ.N.OR.M1.EQ.0) GO TO 20
C     REORDER CURRENT ARRAY
      DO 17 I=N2,NPM
 17   Y(I)=XY(I)
      JJ=N1C+1
      J=N1
      DO 18 I=JJ,NPM
      J=J+1
 18   XY(J)=Y(I)
      DO 19 I=N2,N1C
      J=J+1
 19   XY(J)=Y(I)
 20   IF (NSCON.EQ.0) GO TO 22
      J=NEQS-1
      DO 21 I=1,NSCON
      J=J+1
      JJ=ISCON(I)
 21   XY(JJ)=XY(J)
 22   RETURN
      END
```

SOLVE

PURPOSE

To solve the system LUx = B, where L is a lower triangular matrix with ones on the diagonal, U is an upper triangular matrix, and B is the right-hand side vector (RHS).

METHOD

The algorithm used is described on pages 409-415 of ref. 1. The solution of the matrix equation LUx = B is found by first solving

$$Ly = B, \tag{3}$$

and then

$$Ux = y, \tag{4}$$

since

$$LUx = Ly = B \ .$$

The solution of equations (3) and (4) is straightforward since the matrices are both triangular. The solution of equation (3) can be written

$$y_i = \frac{1}{\ell_{ii}}\left(b_i - \sum_{j=1}^{i-1} \ell_{ij}\, y_j\right) \quad i = 1, \ldots, n \ .$$

Equation (4) can be written similarly.

The L and U matrices are both supplied by the subroutine FACTR and are stored in the matrix A; the 1's on the diagonal of L are suppressed. Care must be exercised in the solution, since rows were interchanged during factorization, and this necessitates rearranging the RHS vector; furthermore, the L matrix itself is not completely rearranged. The information pertinent to the row rearrangements has been stored by FACTR in an integer array (IP), and it is used in the computations. The final solution of the equations is overwritten on the input RHS vector B.

The only differences between the coding in SOLVE and the coding suggested in ref. 1 are: (1) double precision variables are not used for the accumulation of sums, since, for the size of matrices anticipated in core, the computer word length is sufficient, and (2) the transposes of the L and U matrices are supplied in A by FACTR. Thus, the row and column indices used in the routine are reversed to account for this transposition.

CODING

S015 - S025 The solution for y in equation (3).

S029 - S039 The solution for x in equation (4) and the storage of the solution in B.

SYMBOL DICTIONARY

A = array contains the input L and U matrices

B = array contains the input RHS and is overwritten with the solution

I = DO loop index

IP = array contains row positioning information

IP1 = I + 1

J = DO loop index

K = DO loop index

N = order of the matrix being solved

NDIM = dimension of the array where the matrix is stored NDIM $\geq$ N

PI = intermediate integer

SUM = intermediate variable

Y = scratch vector

```
      SUBROUTINE SOLVE (N,A,IP,B,NDIM)
C
C     SUBROUTINE TO SOLVE THE MATRIX EQUATION LU*X=B WHERE L IS A UNIT
C     LOWER TRIANGULAR MATRIX AND U IS AN UPPER TRIANGULAR MATRIX BOTH
C     OF WHICH ARE STORED IN A.  THE RHS VECTOR B IS INPUT AND THE
C     SOLUTION IS RETURNED THROUGH VECTOR B.    (MATRIX TRANSPOSED.
C
      COMPLEX A,B,Y,SUM
      INTEGER PI
      COMMON /SCRATM/ Y(600)
      DIMENSION A(NDIM,NDIM), IP(NDIM), B(NDIM)
C
C     FORWARD SUBSTITUTION
C
      DO 3 I=1,N
      PI=IP(I)
      Y(I)=B(PI)
      B(PI)=B(I)
      IP1=I+1
      IF (IP1.GT.N) GO TO 2
      DO 1 J=IP1,N
      B(J)=B(J)-A(I,J)*Y(I)
1     CONTINUE
2     CONTINUE
3     CONTINUE
C
C     BACKWARD SUBSTITUTION
C
      DO 6 K=1,N
      I=N-K+1
      SUM=(0.,0.)
      IP1=I+1
      IF (IP1.GT.N) GO TO 5
      DO 4 J=IP1,N
      SUM=SUM+A(J,I)*B(J)
4     CONTINUE
5     CONTINUE
      B(I)=(Y(I)-SUM)/A(I,I)
6     CONTINUE
      RETURN
      END
```

SOLVES

PURPOSE

To control solution of the matrix equation, including transforming and reordering the solution vector.

METHOD

When SOLVES is called, the array B contains the excitation computed by subroutines ETMNS or NETWK. The exciting electric field on all segments is stored first in B, followed by the magnetic fields on all patches. In the case of a symmetric structure, however, the matrix is filled with the coefficients of all segment and patch equations in the first symmetric sector occurring first. These are followed by the coefficients for successive sectors in the same order. This order is required for the solution procedure for symmetric structures described in section III-5 of Part I. For the case of a symmetric structure with both segments and patches, SOLVES first rearranges the excitation coefficients in array B to correspond to the order of the matrix coefficients.

For symmetric structures, SOLVES then computes the transforms of the subvectors in B according to equation (88) of Part I. Subroutine SOLVE or LTSOLV is then called to compute the solution or solution subvectors. The procedure is selected by the parameter ICASE as follows.

1 No symmetry, matrix in core. SOLVE is called for the solution.

2 Symmetry, matrix in core. SOLVE is called for each subvector.

3 No symmetry, matrix out of core. LTSOLV is called for the solution.

4 Symmetry, complete matrix does not fit in core but submatrices do. SOLVE is called for each subvector after first reading the appropriate submatrix from file IFL1.

5 Symmetry, submatrices do not fit in core. LTSOLV is called for each subvector.

SOLVES then computes the total current by inverse transforming the subvectors by equation (115) of Part I. For a symmetric structure with segments and patches, SOLVES then rearranges the solution in array B to put all segment currents first, followed by all patch currents, which is the order of the original excitation coefficients.

SOLVES

Multiple right-hand-side vectors (NRH) may be processed simultaneously at each step in SOLVES. This reduces the time spent reading files when LTSOLV is called, and is used in computing $A^{-1}B$ in the NGF procedure.

CODING

SS22 - SS39 Rearrange excitation coefficients.

SS43 - SS56 Transform subvectors.

SS63 - SS75 Solve for each subvector.

SS81 - SS94 Inverse transform subvectors.

SS96 - SS113 Rearrange solution coefficients.

SYMBOL DICTIONARY

A = array set aside for in-core matrix storage, i.e., factored matrices

B = right-hand side; the solution is overwritten on this array also

FNOP = decimal form of NOP

FNORM = 1/FNOP

IFL1 = file with matrix blocks in normal order

IFL2 = file with matrix blocks in reversed order

IP = array containing positioning data used in SOLVE

M = number of patches

MP = number of patches in a symmetric sector

N = number of segments

NCOL = number of columns in array A

NEQ = order of complete matrix

NOP = number of symmetric sectors

NP = number of segments in a symmetric sector

NPEQ = order of a submatrix

NRH = number of right-hand-side vectors in B

NROW = number of rows in A

SSX = array containing the coefficients S_{ik} in equation (89) of Part I

SUM = summation variable

Y = scratch vector

```
      SUBROUTINE SOLVES (A,IP,B,NEQ,NRH,NP,N,MP,M,IFL1,IFL2)
C
C     SUBROUTINE SOLVES, FOR SYMMETRIC STRUCTURES, HANDLES THE
C     TRANSFORMATION OF THE RIGHT HAND SIDE VECTOR AND SOLUTION OF THE
C     MATRIX EQ.
C
      COMPLEX A,B,Y,SUM,SSX
      COMMON /SMAT/ SSX(16,16)
      COMMON /SCRATM/ Y(600)
      COMMON /MATPAR/ ICASE,NBLOKS,NPBLK,NLAST,NBLSYM,NPSYM,NLSYM,IMAT,I
     1CASX,NBBX,NPBX,NLBX,NBBL,NPBL,NLBL
      DIMENSION A(1), IP(1), B(NEQ,NRH)
      NPEQ=NP+2*MP
      NOP=NEQ/NPEQ
      FNOP=NOP
      FNORM=1./FNOP
      NROW=NEQ
      IF (ICASE.GT.3) NROW=NPEQ
      IF (NOP.EQ.1) GO TO 11
      DO 10 IC=1,NRH
      IF (N.EQ.0.OR.M.EQ.0) GO TO 6
      DO 1 I=1,NEQ
    1 Y(I)=B(I,IC)
      KK=2*MP
      IA=NP
      IB=N
      J=NP
      DO 5 K=1,NOP
      IF (K.EQ.1) GO TO 3
      DO 2 I=1,NP
      IA=IA+1
      J=J+1
    2 B(J,IC)=Y(IA)
      IF (K.EQ.NOP) GO TO 5
    3 DO 4 I=1,KK
      IB=IB+1
      J=J+1
    4 B(J,IC)=Y(IB)
    5 CONTINUE
C
C     TRANSFORM MATRIX EQ. RHS VECTOR ACCORDING TO SYMMETRY MODES
C
    6 DO 10 I=1,NPEQ
      DO 7 K=1,NOP
      IA=I+(K-1)*NPEQ
    7 Y(K)=B(IA,IC)
      SUM=Y(1)
      DO 8 K=2,NOP
    8 SUM=SUM+Y(K)
      B(I,IC)=SUM*FNORM
      DO 10 K=2,NOP
      IA=I+(K-1)*NPEQ
      SUM=Y(1)
      DO 9 J=2,NOP
    9 SUM=SUM+Y(J)*CONJG(SSX(K,J))
   10 B(IA,IC)=SUM*FNORM
   11 IF (ICASE.LT.3) GO TO 12
      REWIND IFL1
      REWIND IFL2
C
C     SOLVE EACH MODE EQUATION
C
   12 DO 16 KK=1,NOP
      IA=(KK-1)*NPEQ+1
```

```
      IB=IA
      IF (ICASE.NE.4) GO TO 13
      I=NPEQ*NPEQ
      READ (IFL1) (A(J),J=1,I)
      IB=1
   13 IF (ICASE.EQ.3.OR.ICASE.EQ.5) GO TO 15
      DO 14 IC=1,NRH
   14 CALL SOLVE (NPEQ,A(IB),IP(IA),B(IA,IC),NROW)
      GO TO 16
   15 CALL LTSOLV (A,NPEQ,IP(IA),B(IA,1),NEQ,NRH,IFL1,IFL2)
   16 CONTINUE
      IF (NOP.EQ.1) RETURN
C
C     INVERSE TRANSFORM THE MODE SOLUTIONS
C
      DO 26 IC=1,NRH
      DO 20 I=1,NPEQ
      DO 17 K=1,NOP
      IA=I+(K-1)*NPEQ
   17 Y(K)=B(IA,IC)
      SUM=Y(1)
      DO 18 K=2,NOP
   18 SUM=SUM+Y(K)
      B(I,IC)=SUM
      DO 20 K=2,NOP
      IA=I+(K-1)*NPEQ
      SUM=Y(1)
      DO 19 J=2,NOP
   19 SUM=SUM+Y(J)*SSX(K,J)
   20 B(IA,IC)=SUM
      IF (N.EQ.0.OR.M.EQ.0) GO TO 26
      DO 21 I=1,NEQ
   21 Y(I)=B(I,IC)
      KK=2*MP
      IA=NP
      IB=N
      J=NP
      DO 25 K=1,NOP
      IF (K.EQ.1) GO TO 23
      DO 22 I=1,NP
      IA=IA+1
      J=J+1
   22 B(IA,IC)=Y(J)
      IF (K.EQ.NOP) GO TO 25
   23 DO 24 I=1,KK
      IB=IB+1
      J=J+1
   24 B(IB,IC)=Y(J)
   25 CONTINUE
   26 CONTINUE
      RETURN
      END
```

TBF

PURPOSE

To evaluate the current expansion function associated with a given segment.

METHOD

The current expansion function is described in section III-1 of Part I. The parameter I is the number of the segment on which the function is centered. On segment I and on all segments connected to either end of segment I, the function has the form

$$f_j(s) = A_j + B_j \sin[k(s - s_j)] + C_j \cos[k(s - s_j)] ,$$

where j is the segment number. TBF locates all connected segments and stores the segment numbers, j, in JCO in COMMON/SEGJ/. It computes A_j, B_j, and C_j and stores them in AX, BX, and CX, respectively, in the same location as was used in JCO. A_j, B_j, and C_j for j = I are stored last in the arrays.

If ICAP = 0, the function goes to zero at an end of segment I to which no other segment or surface is connected. If ICAP ≠ 0, the function has a non-zero value at a free end, allowing for the current onto the wire end cap.

CODING

Equations and symbols refer to Part I.

TB9 - TB55 This code forms a loop that locates all segments connected to the ends of segment I, first for end 1 (IEND = -1) and then for end 2 (IEND = 1).

TB9 - TB16 Parameters are initialized to start search for segments connected to end 1 of segment I.

TB34 PP = P_i^- for end 1 of segment I or P_i^+ for end 2 of segment I.

TB35 - TB37 Equations (43) to (48) of Part I evaluated except for $Q_i^{\pm}$:

$AX(JSNO) = A_j^{\pm}/Q_i^{\pm}$

$BX(JSNO) = B_j^{\pm}/Q_i^{\pm}$

$CX(JSNO) = C_j^{\pm}/Q_i^{\pm}$

JCO(JSNO) = j

TB38 Exit from loop if segment I is connected to a surface or ground plane. Segment I will occur in COMMON/SEGJ/ twice

	in this case, once for the center of the expansion function on segment I and once for the part of the function extending onto the image of segment I in the surface. Line TB45 changes the sign of $B_j^{\pm}$ for the image term. The sum of the two parts of the function on segment I then has zero derivative at the end connected to the surface.
TB39 - TB42	Check appropriate end of segment j to determine whether it shows a connection to segment I (end of search) or connection to another segment (multiple junction).
TB44	Continue search for connected segments (multiple junction).
TB46	Exit from loop after finishing search for both ends of segment I.
TB47 - TB55	Store values for end 1 of segment I and initialize for end 2. Then return to previous loop.
TB59 - TB70	Evaluate functions of segment length and radius for segment I. For $k\Delta < 0.03$, a series is used for $1 - \cos k\Delta$, where Δ = segment length.
TB73 - TB86	Final calculations if neither end of segment I is a free end.
TB89 - TB102	Final calculations for free end on end 1 of segment I.
TB104 - TB117	Final calculations for free end on end 2 of segment I.
TB119 - TB126	Final calculations for free ends on both ends of segment I.
TB128	$A_j = -1$ for $j = I$ in all cases.

SYMBOL DICTIONARY

AJ = a_j^-

AP = a_j^+

CD = $\cos k\Delta_j$

CDH = $\cos(k\Delta_j/2)$

D = $k\Delta_j/2$ or $\cos k\Delta_i - X_i \sin k\Delta_i$

ICAP = flag to determine whether the function goes to zero at a free end

IEND = -1 during calculations for end 1 of segment I and +1 for end 2.

JCOX = connection index

JEND = -1 if end 1 of a segment is connected to segment I, +1 if end 2 is connected to segment I.

JMAX = maximum number of segments allowed in the expansion function. This includes segment I and all segments connected to either end.

JSNOP = JSN + 1

NJUN1 = N^-

NJUN2 = N^+

OMC = $1 - \cos k\Delta_j$

PI = π

PM = P_i^-

PP = P_i^+

QM = Q_i^-

QP = Q_i^+

SD = $\sin k\Delta_j$

SDH = $\sin (k\Delta_j/2)$

SIG = sign for calculation of A_j and C_j

XX1 = $J_1(ka_i)/J_0(ka_i)$ (small argument series used for Bessel functions)

CONSTANTS

0.577215664 = Eulers constant

0.015 = 0.03/2

1.3888889E-3 = 1/720

3.141592654 = π

4.1666666667E-2 = 1/24

```
      SUBROUTINE TBF (I,ICAP)                                           TB   1
C     COMPUTE BASIS FUNCTION I                                          TB   2
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300 TB   3
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX( TB   4
     2300),WLAM,IPSYM                                                   TB   5
      COMMON /SEGJ/ AX(30),BX(30),CX(30),JCO(30),JSNO,ISCON(50),NSCON,IP TB   6
     1CON(10),NPCON                                                     TB   7
      DATA PI/3.141592654/,JMAX/30/                                     TB   8
      JSNO=0                                                            TB   9
      PP=0.                                                             TB  10
      JCOX=ICON1(I)                                                     TB  11
      IF (JCOX.GT.10000) JCOX=I                                         TB  12
      JEND=-1                                                           TB  13
      IEND=-1                                                           TB  14
      SIG=-1.                                                           TB  15
      IF (JCOX) 1,10,2                                                  TB  16
1     JCOX=-JCOX                                                        TB  17
      GO TO 3                                                           TB  18
2     SIG=-SIG                                                          TB  19
      JEND=-JEND                                                        TB  20
3     JSNO=JSNO+1                                                       TB  21
      IF (JSNO.GE.JMAX) GO TO 28                                        TB  22
      JCO(JSNO)=JCOX                                                    TB  23
      D=PI*SI(JCOX)                                                     TB  24
      SDH=SIN(D)                                                        TB  25
      CDH=COS(D)                                                        TB  26
      SD=2.*SDH*CDH                                                     TB  27
      IF (D.GT.0.015) GO TO 4                                           TB  28
      OMC=4.*D*D                                                        TB  29
      OMC=((1.3888889E-3*OMC-4.1666666667E-2)*OMC+.5)*OMC               TB  30
      GO TO 5                                                           TB  31
4     OMC=1.-CDH*CDH+SDH*SDH                                            TB  32
5     AJ=1./(ALOG(1./(PI*BI(JCOX)))-.577215664)                         TB  33
      PP=PP-OMC/SD*AJ                                                   TB  34
      AX(JSNO)=AJ/SD*SIG                                                TB  35
      BX(JSNO)=AJ/(2.*CDH)                                              TB  36
      CX(JSNO)=-AJ/(2.*SDH)*SIG                                         TB  37
      IF (JCOX.EQ.I) GO TO 8                                            TB  38
      IF (JEND.EQ.1) GO TO 6                                            TB  39
      JCOX=ICON1(JCOX)                                                  TB  40
      GO TO 7                                                           TB  41
6     JCOX=ICON2(JCOX)                                                  TB  42
7     IF (IABS(JCOX).EQ.I) GO TO 9                                      TB  43
      IF (JCOX) 1,28,2                                                  TB  44
8     BX(JSNO)=-BX(JSNO)                                                TB  45
9     IF (IEND.EQ.1) GO TO 11                                           TB  46
10    PM=-PP                                                            TB  47
      PP=0.                                                             TB  48
      NJUN1=JSNO                                                        TB  49
      JCOX=ICON2(I)                                                     TB  50
      IF (JCOX.GT.10000) JCOX=I                                         TB  51
      JEND=1                                                            TB  52
      IEND=1                                                            TB  53
      SIG=-1.                                                           TB  54
      IF (JCOX) 1,11,2                                                  TB  55
11    NJUN2=JSNO-NJUN1                                                  TB  56
      JSNOP=JSNO+1                                                      TB  57
      JCO(JSNOP)=I                                                      TB  58
      D=PI*SI(I)                                                        TB  59
      SDH=SIN(D)                                                        TB  60
      CDH=COS(D)                                                        TB  61
      SD=2.*SDH*CDH                                                     TB  62
      CD=CDH*CDH-SDH*SDH                                                TB  63
      IF (D.GT.0.015) GO TO 12                                          TB  64
```

```fortran
      OMC=4.*D*D
      OMC=((1.3888889E-3*OMC-4.1666666667E-2)*OMC+.5)*OMC
      GO TO 13
   12 OMC=1.-CD
   13 AP=1./(ALOG(1./(PI*BI(I)))-.577215664)
      AJ=AP
      IF (NJUN1.EQ.0) GO TO 16
      IF (NJUN2.EQ.0) GO TO 20
      QP=SD*(PM*PP+AJ*AP)+CD*(PM*AP-PP*AJ)
      QM=(AP*OMC-PP*SD)/QP
      QP=-(AJ*OMC+PM*SD)/QP
      BX(JSNOP)=(AJ*QM+AP*QP)*SDH/SD
      CX(JSNOP)=(AJ*QM-AP*QP)*CDH/SD
      DO 14 IEND=1,NJUN1
      AX(IEND)=AX(IEND)*QM
      BX(IEND)=BX(IEND)*QM
   14 CX(IEND)=CX(IEND)*QM
      JEND=NJUN1+1
      DO 15 IEND=JEND,JSNO
      AX(IEND)=-AX(IEND)*QP
      BX(IEND)=BX(IEND)*QP
   15 CX(IEND)=-CX(IEND)*QP
      GO TO 27
   16 IF (NJUN2.EQ.0) GO TO 24
      IF (ICAP.NE.0) GO TO 17
      XXI=0.
      GO TO 18
   17 QP=PI*BI(I)
      XXI=QP*QP
      XXI=QP*(1.-.5*XXI)/(1.-XXI)
   18 QP=-(OMC+XXI*SD)/(SD*(AP+XXI*PP)+CD*(XXI*AP-PP))
      D=CD-XXI*SD
      BX(JSNOP)=(SDH+AP*QP*(CDH-XXI*SDH))/D
      CX(JSNOP)=(CDH+AP*QP*(SDH+XXI*CDH))/D
      DO 19 IEND=1,NJUN2
      AX(IEND)=-AX(IEND)*QP
      BX(IEND)=BX(IEND)*QP
   19 CX(IEND)=-CX(IEND)*QP
      GO TO 27
   20 IF (ICAP.NE.0) GO TO 21
      XXI=0.
      GO TO 22
   21 QM=PI*BI(I)
      XXI=QM*QM
      XXI=QM*(1.-.5*XXI)/(1.-XXI)
   22 QM=(OMC+XXI*SD)/(SD*(AJ-XXI*PM)+CD*(PM+XXI*AJ))
      D=CD-XXI*SD
      BX(JSNOP)=(AJ*QM*(CDH-XXI*SDH)-SDH)/D
      CX(JSNOP)=(CDH-AJ*QM*(SDH+XXI*CDH))/D
      DO 23 IEND=1,NJUN1
      AX(IEND)=AX(IEND)*QM
      BX(IEND)=BX(IEND)*QM
   23 CX(IEND)=CX(IEND)*QM
      GO TO 27
   24 BX(JSNOP)=0.
      IF (ICAP.NE.0) GO TO 25
      XXI=0.
      GO TO 26
   25 QP=PI*BI(I)
      XXI=QP*QP
      XXI=QP*(1.-.5*XXI)/(1.-XXI)
   26 CX(JSNOP)=1./(CDH-XXI*SDH)
   27 JSNO=JSNOP
      AX(JSNO)=-1.
```

```
      RETURN                                                    TB 129
   28 PRINT 29, I                                               TB 130
      STOP                                                      TB 131
C                                                               TB 132
   29 FORMAT (43H TBF - SEGMENT CONNECTION ERROR FOR SEGMENT,I5)  TB 133
      END                                                       TB 134-
```

TEST

PURPOSE

To compute the relative difference of two numerical integration results for the Romberg variable-interval-width integration routines.

METHOD

The first numerical integration result is the complex number (F1R, F1I) and the second is (F2R, F2I). The real and imaginary parts of the two results are subtracted and the differences are divided by the largest of F2R, F2I, DMIN or 10^{-37}. The denominator is chosen to avoid trying to maintain a small relative error for a quantity that is insignificantly small.

SYMBOL DICTIONARY

ABS	=	external routine (absolute value)
DEN	=	largest of |F2R| and |F2I|
DMIN	=	minimum denominator
F1I	=	imaginary part of first integration result
F1R	=	real part of first integration result
F2I	=	imaginary part of second integration result
F2R	=	real part of second integration result
TI	=	relative difference of imaginary parts
TR	=	relative difference of real parts

CONSTANT

1.E-37 = tolerance in test for zero

TEST

```
      SUBROUTINE TEST (F1R,F2R,TR,F1I,F2I,TI,DMIN)
C
C     TEST FOR CONVERGENCE IN NUMERICAL INTEGRATION
C
      DEN=ABS(F2R)
      TR=ABS(F2I)
      IF (DEN.LT.TR) DEN=TR
      IF (DEN.LT.DMIN) DEN=DMIN
      IF (DEN.LT.1.E-37) GO TO 1
      TR=ABS((F1R-F2R)/DEN)
      TI=ABS((F1I-F2I)/DEN)
      RETURN
    1 TR=0.
      TI=0.
      RETURN
      END
```

TRIO

PURPOSE

To evaluate each of the parts of current expansion functions on a single segment due to each of the segments connected to the given segment.

METHOD

TRIO consists of a loop that uses the connection data in arrays ICON1 and ICON2 to locate all segments connected to segment J. Subroutine SBF is called to evaluate the current expansion function centered on each connected segment and on segment J. Only the function coefficients for that part of each expansion function on segment J are returned and are stored in arrays AX, BX, and CX. The number of the segment with which each expansion function part is associated is stored in array JCO and the total number of expansion functions involved is stored as JSNO.

SYMBOL DICTIONARY

IEND = -1 during calculations for end 1 of segment J, and +1 for end 2

JCOX = number of a segment connected to segment J

JEND = -1 if end 1 of segment JCOX is connected to segment J; +1 if end 2 of segment JCOX is connected to segment J

JMAX = dimension of the arrays in COMMON/SEGJ/

```
      SUBROUTINE TRIO (J)
C     COMPUTE THE COMPONENTS OF ALL BASIS FUNCTIONS ON SEGMENT J
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      COMMON /SEGJ/ AX(30),BX(30),CX(30),JCO(30),JSNO,ISCON(50),NSCON,IP
     1CON(10),NPCON
      DATA JMAX/30/
      JSNO=0
      JCOX=ICON1(J)
      IF (JCOX.GT.10000) GO TO 7
      JEND=-1
      IEND=-1
      IF (JCOX) 1,7,2
    1 JCOX=-JCOX
      GO TO 3
    2 JEND=-JEND
    3 IF (JCOX.EQ.J) GO TO 6
      JSNO=JSNO+1
      IF (JSNO.GE.JMAX) GO TO 9
      CALL SBF (JCOX,J,AX(JSNO),BX(JSNO),CX(JSNO))
      JCO(JSNO)=JCOX
      IF (JEND.EQ.1) GO TO 4
      JCOX=ICON1(JCOX)
      GO TO 5
    4 JCOX=ICON2(JCOX)
    5 IF (JCOX) 1,9,2
    6 IF (IEND.EQ.1) GO TO 8
    7 JCOX=ICON2(J)
      IF (JCOX.GT.10000) GO TO 8
      JEND=1
      IEND=1
      IF (JCOX) 1,8,2
    8 JSNO=JSNO+1
      CALL SBF (J,J,AX(JSNO),BX(JSNO),CX(JSNO))
      JCO(JSNO)=J
      RETURN
    9 PRINT 10, J
      STOP
C
   10 FORMAT (44H TRIO - SEGMENT CONNENTION ERROR FOR SEGMENT,I5)
      END
```

UNERE

PURPOSE

To calculate the electric field due to unit currents in the $\hat{t}_1$ and $\hat{t}_2$ directions on a surface patch.

METHOD

The electric field due to a patch j is calculated by the expression

$$\bar{E}(\bar{r}_0) = \frac{\eta_0}{i8\pi^2}\left[\left(\frac{-1 - i2\pi R/\lambda + 4\pi^2 (R/\lambda)^2}{(R/\lambda)^3}\right)\bar{J}_j + \left(\frac{3 + i6\pi R/\lambda - 4\pi^2 (R/\lambda)^2}{(R/\lambda)^5}\right)\bar{J}_j \cdot (\bar{R}/\lambda)(\bar{R}/\lambda)\right]\exp(-i2\pi R/\lambda)\frac{\Delta A_j}{\lambda^2},$$

where $i = \sqrt{-1}$, $\bar{J}_j = J_{1j}\hat{t}_{1j} + J_{2j}\hat{t}_{2j}$, $\bar{R}$ is the vector from the source to the observation point, and ΔA_j is the area of the patch. For UNERE, J_{1j} and J_{2j} are unity. The expression above for a single patch is obtained from the surface integral in equation (3) in Part I where constant current and one step integration are used for the patch.

CODING

UE14 - UE20 z components of patch parameters are adjusted for direct or reflected fields.

UE25 - UE32 For $R < 10^{-10}$, the fields are set to zero.

UE34 - UE47 Expression for $\bar{E}$ is evaluated for $\bar{J}_j$ equal to $\hat{t}_1$ and $\hat{t}_2$.

UE50 - UE55 For reflection in a perfect ground, $\bar{E}$ is reversed in sign.

UE57 - UE79 For reflection in an imperfect ground, $\bar{E}$ is multiplied by the reflection coefficients.

SYMBOL DICTIONARY

CONST = $\frac{\eta_0}{8\pi^2}$

CTH = $\cos\theta$; θ is the angle between the reflected ray and the normal to the surface

EDP = $(\bar{E} \cdot \hat{p})(R_H - R_V)$

ER = $\frac{\eta_0}{i8\pi^2}$ exp(−i 2π R/λ) $\Delta A_j/\lambda^2$ at UE37

= Q2 ($\hat{t}_{1j} \cdot \bar{R}/\lambda$) at UE40

= Q2 ($\hat{t}_{2j} \cdot \bar{R}/\lambda$) at UE44

EXK, EYK, EZK = $\bar{E}$ due to current $\hat{t}_{1j}$

EXS, EYS, EZS = $\bar{E}$ due to current $\hat{t}_{2j}$

IPGND = flag to cause computation of reflected field when equal to 2

PX, PY = $\hat{p}$; unit vector normal to the plane of incidence of the reflected ray

Q1 = $\left[\frac{-1 - i2\pi R/\lambda + 4\pi^2(R/\lambda)^2}{(R/\lambda)^3}\right]$(ER)

Q2 = $\left[\frac{3 + i6\pi R/\lambda - 4\pi^2(R/\lambda)^2}{(R/\lambda)^5}\right]$(ER)

R = R/λ

RRH = R_H

RRV = R_V

RT = $(R/\lambda)^3$

RX, RY, RZ = $\bar{R}/\lambda$

R2 = $(R/\lambda)^2$

S = $\Delta A_j/\lambda^2$

T1XJ, T1YJ, T1ZJ = $\hat{t}_{1j}$

T2XJ, T2YJ, T2ZJ = $\hat{t}_{2j}$

TPI = 2π

TT1 = −2πR/λ

TT2 = $4\pi 2(R/\lambda)^2$

XOB, YOB, ZOB = field evaluation point

XYMAG = magnitude of the projection of $\overline{R}/\lambda$ onto the x-y plane

ZR = z component of $\overline{R}/\lambda$ after reflection

CONSTANTS

$4.771341188 = \frac{\eta_0}{8\pi^2}$

$6.283185308 = 2\pi$

```
      SUBROUTINE UNERE (XOB,YOB,ZOB)
C     CALCULATES THE ELECTRIC FIELD DUE TO UNIT CURRENT IN THE T1 AND T2
C     DIRECTIONS ON A PATCH
      COMPLEX EXK,EYK,EZK,EXS,EYS,EZS,EXC,EYC,EZC,ZRATI,ZRATI2,T1,ER,Q1,
     1Q2,RRV,RRH,EDP,FRATI
      COMMON /DATAJ/ S,B,XJ,YJ,ZJ,CABJ,SABJ,SALPJ,EXK,EYK,EZK,EXS,EYS,EZ
     1S,EXC,EYC,EZC,RKH,IEXK,IND1,IND2,IPGND
      COMMON /GND/ZRATI,ZRATI2,FRATI,CL,CH,SCRWL,SCRWR,NRADL,KSYMP,IFAR,
     1IPERF,T1,T2
      EQUIVALENCE (T1XJ,CABJ), (T1YJ,SABJ), (T1ZJ,SALPJ), (T2XJ,B), (T2Y
     1J,IND1), (T2ZJ,IND2)
      DATA TPI,CONST/6.283185308,4.771341188/
C     CONST=ETA/(8.*PI**2)
      ZR=ZJ
      T1ZR=T1ZJ
      T2ZR=T2ZJ
      IF (IPGND.NE.2) GO TO 1
      ZR=-ZR
      T1ZR=-T1ZR
      T2ZR=-T2ZR
1     RX=XOB-XJ
      RY=YOB-YJ
      RZ=ZOB-ZR
      R2=RX*RX+RY*RY+RZ*RZ
      IF (R2.GT.1.E-20) GO TO 2
      EXK=(0.,0.)
      EYK=(0.,0.)
      EZK=(0.,0.)
      EXS=(0.,0.)
      EYS=(0.,0.)
      EZS=(0.,0.)
      RETURN
2     R=SQRT(R2)
      TT1=-TPI*R
      TT2=TT1*TT1
      RT=R2*R
      ER=CMPLX(SIN(TT1),-COS(TT1))*(CONST*S)
      Q1=CMPLX(TT2-1.,TT1)*ER/RT
      Q2=CMPLX(3.-TT2,-3.*TT1)*ER/(RT*R2)
      ER=Q2*(T1XJ*RX+T1YJ*RY+T1ZR*RZ)
      EXK=Q1*T1XJ+ER*RX
      EYK=Q1*T1YJ+ER*RY
      EZK=Q1*T1ZR+ER*RZ
      ER=Q2*(T2XJ*RX+T2YJ*RY+T2ZR*RZ)
      EXS=Q1*T2XJ+ER*RX
      EYS=Q1*T2YJ+ER*RY
      EZS=Q1*T2ZR+ER*RZ
      IF (IPGND.EQ.1) GO TO 6
      IF (IPERF.NE.1) GO TO 3
      EXK=-EXK
      EYK=-EYK
      EZK=-EZK
      EXS=-EXS
      EYS=-EYS
      EZS=-EZS
      GO TO 6
3     XYMAG=SQRT(RX*RX+RY*RY)
      IF (XYMAG.GT.1.E-6) GO TO 4
      PX=0.
      PY=0.
      CTH=1.
      RRV=(1.,0.)
      GO TO 5
4     PX=-RY/XYMAG
```

```
      PY=RX/XYMAG
      CTH=RZ/SQRT(XYMAG*XYMAG+RZ*RZ)
      RRV=CSQRT(1.-ZRATI*ZRATI*(1.-CTH*CTH))
5     RRH=ZRATI*CTH
      RRH=(RRH-RRV)/(RRH+RRV)
      RRV=ZRATI*RRV
      RRV=-(CTH-RRV)/(CTH+RRV)
      EDP=(EXK*PX+EYK*PY)*(RRH-RRV)
      EXK=EXK*RRV+EDP*PX
      EYK=EYK*RRV+EDP*PY
      EZK=EZK*RRV
      EDP=(EXS*PX+EYS*PY)*(RRH-RRV)
      EXS=EXS*RRV+EDP*PX
      EYS=EYS*RRV+EDP*PY
      EZS=EZS*RRV
6     RETURN
      END
```

WIRE

PURPOSE

To compute segment coordinates to fill COMMON/DATA/ for a straight line of segments.

METHOD

The formal parameters specify the beginning and ending points of the line and the number of segments into which it is to be divided. The code computes the coordinates of the end points of each segment. The lengths of successive segments are scaled by the factor RDEL if this factor is not one. For NS segments, the length of the first segment is

$$S_1 = \frac{L(1 - RDEL)}{1 - (RDEL)^{NS}}$$

or $$S_1 = L/NS \text{ if } RDEL = 1$$

where L is the total length of wire.

The radius is RAD for the first segment and is scaled by RRAD.

SYMBOL DICTIONARY

DELZ	=	segment length
FNS	=	real number equivalent of NS
IST	=	initial segment number
ITG	=	tag number assigned to all segments of the line
NS	=	number of segments into which line is divided
RAD	=	radius of first segment
RADZ	=	segment radius
RD, RDEL	=	scaling factor for segment length
RRAD	=	scaling factor for segment radius
XD	=	increment to x coordinates
XS1	=	x coordinate of first end of segment
XS2	=	x coordinate of second end of segment
XW1	=	x coordinate of first end of line
XW2	=	x coordinate of second end of line

X2(I)	= x coordinate of end 2 of segment I
YD	= increment to y coordinates
YS1	= y coordinate of first end of segment
YS2	= y coordinate of second end of segment
YW1	= y coordinate of first end of wire
YW2	= y coordinate of second end of wire
Y2(I)	= y coordinate of end 2 of segment I
ZD	= increment to z coordinates
ZS1	= z coordinate of first end of segment
ZS2	= z coordinate of second end of segment
ZW1	= z coordinate of first end of line
ZW2	= z coordinate of second end of line
Z2(I)	= z coordinate of second end of segment I

```
      SUBROUTINE WIRE (XW1,YW1,ZW1,XW2,YW2,ZW2,RAD,RDEL,RRAD,NS,ITG)
C
C     SUBROUTINE WIRE GENERATES SEGMENT GEOMETRY DATA FOR A STRAIGHT
C     WIRE OF NS SEGMENTS.
C
      COMMON /DATA/ LD,N1,N2,N,NP,M1,M2,M,MP,X(300),Y(300),Z(300),SI(300
     1),BI(300),ALP(300),BET(300),ICON1(300),ICON2(300),ITAG(300),ICONX(
     2300),WLAM,IPSYM
      DIMENSION X2(1), Y2(1), Z2(1)
      EQUIVALENCE (X2(1),SI(1)), (Y2(1),ALP(1)), (Z2(1),BET(1))
      IST=N+1
      N=N+NS
      NP=N
      MP=M
      IPSYM=0
      IF (NS.LT.1) RETURN
      XD=XW2-XW1
      YD=YW2-YW1
      ZD=ZW2-ZW1
      IF (ABS(RDEL-1.).LT.1.E-6) GO TO 1
      DELZ=SQRT(XD*XD+YD*YD+ZD*ZD)
      XD=XD/DELZ
      YD=YD/DELZ
      ZD=ZD/DELZ
      DELZ=DELZ*(1.-RDEL)/(1.-RDEL**NS)
      RD=RDEL
      GO TO 2
1     FNS=NS
      XD=XD/FNS
      YD=YD/FNS
      ZD=ZD/FNS
      DELZ=1.
      RD=1.
2     RADZ=RAD
      XS1=XW1
      YS1=YW1
      ZS1=ZW1
      DO 3 I=IST,N
      ITAG(I)=ITG
      XS2=XS1+XD*DELZ
      YS2=YS1+YD*DELZ
      ZS2=ZS1+ZD*DELZ
      X(I)=XS1
      Y(I)=YS1
      Z(I)=ZS1
      X2(I)=XS2
      Y2(I)=YS2
      Z2(I)=ZS2
      BI(I)=RADZ
      DELZ=DELZ*RD
      RADZ=RADZ*RRAD
      XS1=XS2
      YS1=YS2
3     ZS1=ZS2
      X2(N)=XW2
      Y2(N)=YW2
      Z2(N)=ZW2
      RETURN
      END
```

ZINT

PURPOSE

To compute the internal impedance of a circular wire with finite conductivity.

METHOD

The internal impedance per unit length of a circular wire is given by

$$Z = \frac{j}{b}\sqrt{\frac{f\mu}{2\pi\sigma}}\left[\frac{\mathrm{Ber}(q) + j\mathrm{Bei}(q)}{\mathrm{Ber}'(q) + j\mathrm{Bei}'(q)}\right],$$

where

$q = b\sqrt{2\pi f\mu\sigma}$

σ = wire conductivity

μ = permeability of free space

b = wire radius

f = frequency

Ber, Bei = Kelvin functions

The term that modifies the diagonal matrix element G_{ii} in the interaction matrix is the total impedance of segment i divided by Δ_i/λ, where Δ_i = segment length. Thus, if G_{ii} is the diagonal matrix element without loading, the new element is

$$G_{ii} - Z\Delta_i/(\Delta_i/\lambda) = G_{ii} - Z\lambda \; .$$

Normalized to wavelength, this term is

$$Z_i = Z\lambda = \frac{j}{(b/\lambda)}\sqrt{\frac{c\mu}{2\pi(\sigma\lambda)}}\left[\frac{\mathrm{Ber}(q) + j\mathrm{Bei}(q)}{\mathrm{Ber}'(q) + j\mathrm{Bei}'(q)}\right],$$

where

$q = (b/\lambda)\sqrt{2\pi c\mu(\sigma\lambda)}$

c = velocity of light

The Kelvin functions and derivatives of Kelvin functions are computed from their polynomial approximations.

CODING

ZI8 - ZI15	Functions θ, φ, f, and g for large argument polynomial approximations (see ref. 5).
ZI19 - ZI26	Compute Ber(q) + jBei(q) for $q \leq 8$.
ZI27 - ZI31	Compute Ber'(q) + jBei'(q) for $q \leq 8$.
ZI32	[Ber(q) + jBei(q)]/[Ber'(q) + jBei'(q)].
ZI34	Ber(q) + jBei(q) for $8 < q \leq 110$.
ZI35	Ber'(q) + jBei'(q) for $8 < q \leq 110$.
ZI36	[Ber(q) + jBei(q)]/[Ber'(q) + jBei'(q)].
ZI38	[Ber(q) + jBei(q)]/[Ber'(q) + jBei'(q)] for $110 < q < \infty$.
ZI39	Computation of Z_i.

SYMBOL DICTIONARY

BEI	= Bei(q) or Bei'(q)
BER	= Ber(q) or Ber'(q)
BR1	= Ber(q) + jBei(q) or [Ber(q) + jBei(q)]/[Ber'(q) + Bei'(q)]
BR2	= Ber'(q) + jBei'(q)
CEXP	= external routine [exp(complex argument)]
CMOTP	= $c\mu/(2\pi)$
CMPLX	= external routine (forms complex number)
CN	= $(1 + j)/\sqrt{2}$
D	= function argument
F(D)	= f(D) (see ref. 5)
FJ	= j
G(D)	= g(D) (see ref. 5)
PH(D)	= φ(X), D = 8/X (see ref. 5)
PI	= π
POT	= π/2
ROLAM	= b/λ
S	= $(X/8)^4$
SIGL	= σλ
SQRT	= external routine (square root)
TH(D)	= θ(X), D = 8/X (see ref. 5)
TP	= 2π

TPCMU = $2\pi c\mu$; c = velocity of light

X = q

Y = $(X/8)^2$

ZINT = Z_i

CONSTANTS

1.5707963	= $\pi/2$
3.141592654	= π
6.283185308	= 2π
60.	= $c\mu/2\pi$
2.368705E+3	= $2\pi c\mu$
(0., 1.)	= j
(0.70710678, 0.70710678)	= $(1 + j)/\sqrt{2}$
(0.70710678, -0.70710678)	= limit for $q \to \infty$ of [Ber(q) + jBei(q)]/ [Ber'(q) + jBei'(q)]

Other constants are factors in the polynomial approximations.

```
      COMPLEX FUNCTION ZINT(SIGL,ROLAM)
C
C     ZINT COMPUTES THE INTERNAL IMPEDANCE OF A CIRCULAR WIRE
C
C
      COMPLEX TH,PH,F,G,FJ,CN,BR1,BR2
      COMPLEX CC1,CC2,CC3,CC4,CC5,CC6,CC7,CC8,CC9,CC10,CC11,CC12,CC13,CC
     114
      DIMENSION FJX(2), CNX(2), CCN(28)
      EQUIVALENCE (FJ,FJX), (CN,CNX), (CC1,CCN(1)), (CC2,CCN(3)), (CC3,C
     1CN(5)), (CC4,CCN(7)), (CC5,CCN(9)), (CC6,CCN(11)), (CC7,CCN(13)),
     2(CC8,CCN(15)), (CC9,CCN(17)), (CC10,CCN(19)), (CC11,CCN(21)), (CC1
     32,CCN(23)), (CC13,CCN(25)), (CC14,CCN(27))
      DATA PI,POT,TP,TPCMU/3.1415926,1.5707963,6.2831853,2.368705E+3/
      DATA CMOTP/60.00/,FJX/0.,1./,CNX/.70710678,.70710678/
      DATA CCN/6.E-7,1.9E-6,-3.4E-6,5.1E-6,-2.52E-5,0.,-9.06E-5,-9.01E-5
     1,0.,-9.765E-4,.0110486,-.0110485,0.,-.3926991,1.6E-6,-3.2E-6,1.17E
     2-5,-2.4E-6,3.46E-5,3.38E-5,5.E-7,2.452E-4,-1.3813E-3,1.3811E-3,-6.
     3250Q1E-2,-1.E-7,.7071068,.7071068/
      TH(D)=(((((CC1*D+CC2)*D+CC3)*D+CC4)*D+CC5)*D+CC6)*D+CC7
      PH(D)=(((((CC8*D+CC9)*D+CC10)*D+CC11)*D+CC12)*D+CC13)*D+CC14
      F(D)=SQRT(POT/D)*CEXP(-CN*D+TH(-8./X))
      G(D)=CEXP(CN*D+TH(8./X))/SQRT(TP*D)
      X=SQRT(TPCMU*SIGL)*ROLAM
      IF (X.GT.110.) GO TO 2
      IF (X.GT.8.) GO TO 1
      Y=X/8.
      Y=Y*Y
      S=Y*Y
      BER=((((((-9.01E-6*S+1.22552E-3)*S-.08349609)*S+2.6419140)*S-32.36
     13456)*S+113.77778)*S-64.)*S+1.
      BEI=((((((1.1346E-4*S-.01103667)*S+.52185615)*S-10.567658)*S+72.81
     17777)*S-113.77778)*S+16.)*Y
      BR1=CMPLX(BER,BEI)
      BER=(((((((-3.94E-6*S+4.5957E-4)*S-.02609253)*S+.66047849)*S-6.068
     11481)*S+14.222222)*S-4.)*Y)*X
      BEI=((((((4.609E-5*S-3.79386E-3)*S+.14677204)*S-2.3116751)*S+11.37
     17778)*S-10.666667)*S+.5)*X
      BR2=CMPLX(BER,BEI)
      BR1=BR1/BR2
      GO TO 3
    1 BR2=FJ*F(X)/PI
      BR1=G(X)+BR2
      BR2=G(X)*PH(8./X)-BR2*PH(-8./X)
      BR1=BR1/BR2
      GO TO 3
    2 BR1=CMPLX(.70710678,-.70710678)
    3 ZINT=FJ*SQRT(CMOTP/SIGL)*BR1/ROLAM
      RETURN
      END
```

Section III
Common Blocks

This section discusses each labeled common block which is used in the NEC-2 code. For each common block, a list of the routines in which it is used is given along with a definition of the variables used in conjunction with the common block. The common blocks are presented in alphabetical order.

COMMON/ANGL/ SALP(300)

Routines Using /ANGL/

CABC, CMSS, CMSW, CMWS, CMWW, DATAGN, ETMNS, FFLD, GFIL, GFLD, GFOUT, MOVE, NEFLD, NHFLD, PATCH, QDSRC, REFLC

/ANGL/ Parameters for Wire Segments

SALP(I) = sin (α), where α = elevation angle of segment I (see figure 11)

/ANGL/ Parameters for Surface Patches

$$\text{SALP}(\text{LD}-\text{I}+1) = \begin{cases} +1 \text{ if } \hat{t}_1 \times \hat{t}_2 = \hat{n} \text{ for patch I, or} \\ -1 \text{ if } \hat{t}_1 \times \hat{t}_2 = -\hat{n} \text{ for patch I} \end{cases}$$

The second case occurs when the patch has been produced by reflection of a patch originally input.

COMMON/CMB/ CM(4000)

Routines Using /CMB/

MAIN, GFIL, GFOUT

The interaction matrix is stored in array CM. If the matrix is too large to fit in CM, then pairs of blocks of the matrix are stored in CM as they are needed.

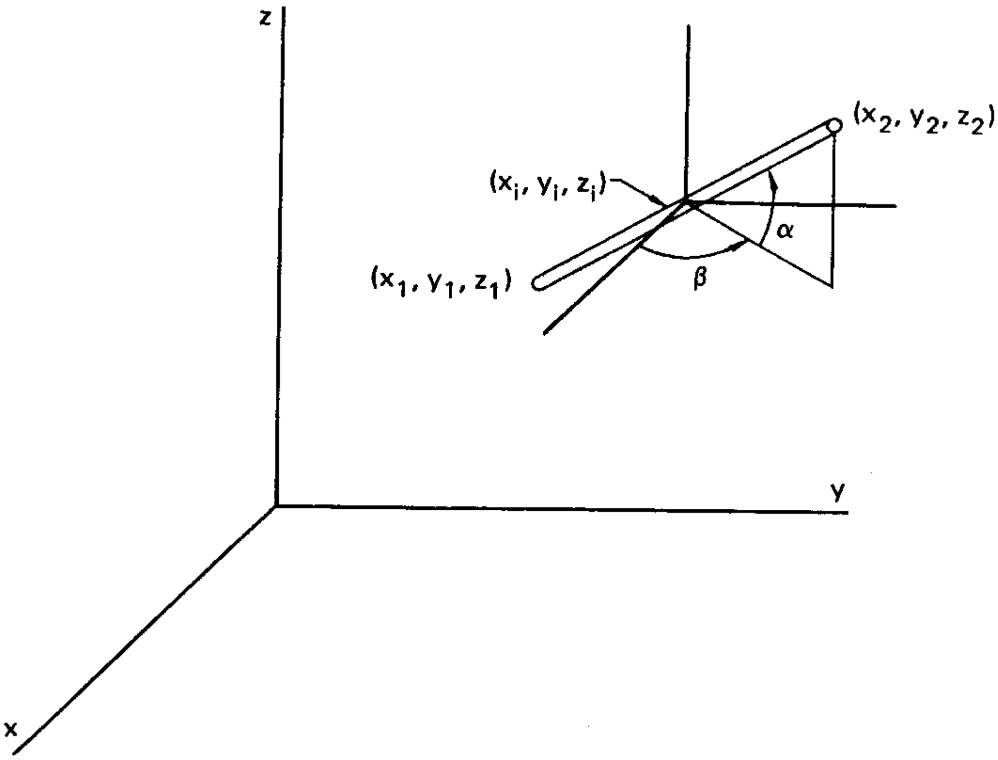


Figure 11. Coordinates of Segment i.

COMMON/CRNT/AIR(300), AII(300), BIR(300), BII(300), CIR(300), CII(300)
CUR(900)

Routines Using /CRNT/

MAIN, CABC, FFLD, GFLD, NEFLD, NETWK, NHFLD

/CRNT/ Parameters for Wire Segments

Subroutine CABC fills the first six arrays in /CRNT/ with the real and imaginary parts of the constants in the current expansion of each segment,

$$I_i(s) = A_i + B_i \sin [k(s - s_i)] + C_i \cos [k(s - s_i)] ,$$

where $s = s_i$ at the center of segment i. Except during intermediate calculations for non-radiating networks, the current basis-function amplitudes are computed and stored in array CUR. CABC replaces the basis function amplitudes in CUR by the current at the center of each segment, $(A_i + C_i)$. For i = I, AIR(I), AII(I) = A_i/λ (real, imaginary)

BIR(I), BII(I) = B_i/λ

CIR(I), CII(I) = C_i/λ

CUR(I) = amplitude of i^{th} basis function going into CABC or $(A_i + C_i)/\lambda$ at end of CABC

/CRNT/ Parameters for Surface Patches

Surface current components are stored in CUR. Before CABC is called, the surface current strengths in directions $\hat{t}_1$ and $\hat{t}_2$ on patch i are stored in CUR(N + 2I - 1) and CUR(N + 2I), respectively where N is the number of segments. After CABC, the x, y and z components of surface current are stored in CUR(N + 3I - 2), CUR(N + 3I - 1) and CUR(N + 3I), respectively.

COMMON/DATA/ LD, N1, N2, N, NP, M1, M2, M, MP, X(300), Y(300), Z(300), SI(300), BI(300), ALP(300), BET(300), ICON1(300), ICON2(300), ITAG(300), ICONX(300), WLAM, IPSYM

Routines Using /DATA/

MAIN, ARC, CABC, CMNGF, CMSET, CMSS, CMSW, CMWS, CMWW, CONECT, DATAGN, ETMNS, FFLD, FFLDS, GFIL, GFLD, GFOUT, ISEGNO, LOAD, MOVE, NEFLD, NETWK, NFPAT, NHFLD, PATCH, QDSRC, RDPAT, REFLC, SBF, TBF, TRIO, WIRE

/DATA/ Parameters for Wire Segments

The arrays in /DATA/ are used to store the parameters defining the segments. Two forms of the segment parameters are used.

During geometry input in routines ARC, CONECT, DATAGN, MOVE, REFLEC and WIRE, the coordinates of the segment ends are stored. The symbol meanings in the geometry routines are:

X(I) = X_1

Y(I) = Y_1

Z(I) = Z_1

SI(I) = X_2 [equivalenced to X2(I)]

ALP(I) = Y_2 [equivalenced to Y2(I)]

BET(I) = Z_2 [equivalenced to Z2(I)]

where X_1, Y_1, Z_1 are the coordinates of the first end of the segment, and X_2, Y_2, Z_2 are the coordinates of the second end, as illustrated in figure 11. Coordinates may have any units but must be scaled to meters before data input is ended, since the main program requires meters.

In the main program, the segment data is converted to: the coordinates of the segment center, components of the unit vector in the direction of the segment, and the segment length. The symbol meanings after the geometry section are:

X(I), Y(I), Z(I) = X_i, Y_i, Z_i (see figure 11.)

SI(I) = segment length

ALP(I) = cos α cos β [equivalenced to CAB(I)]

BET(I) = cos α sin β [equivalenced to SAB(I)]

The z component of the unit vector in the direction of the segment, sin α, is stored in /ANGL/.

The other symbol meanings in /DATA/ for segments are:

BI(I) = radius of segment I

ICON1(I) = connection number for end 1 of segment I. If k is a positive integer less than 10,000, the meaning of ICON1 is as follows.

ICON1(I) = 0: no connection.

ICON1(I) = ±k: end 1 connects to segment k. If more than one segment connects to end 1 of segment I, then k is the number of the next connected segment encountered by starting at I and going through the list of segments in cyclic order. ICON1(I) = +k: parallel reference directions with end 2 of the other segment connecting to end 1 of segment I. ICON1(I) = −k: opposed reference directions.

ICON1(I) = I: end 1 of segment I connects to a ground plane.

ICON1(I) = 10,000 +k: end 1 of segment I connects to a surface with the 4 patches around the connection point numbered k, k + 1, k + 2 and k + 3.

ICON2(I) = connection number for end 2 of segment I.

ITAG(I) = tag number of segment I. This number is assigned during structure input to permit later reference to the segment without knowing the segment index I in the data arrays.

ICONX(I) = equation number for the new basis function when segment I is in a numerical Green's function file and a new segment connects to segment I modifying the old basis function.

/DATA/ Parameters for Surface Patches

Patch parameters are set in subroutine PATCH. The input parameters for a patch are the coordinates of the patch center, patch area, and orientation of the outward, normal unit vector, $\hat{n}$. The parameters stored in /DATA/ are the center point coordinates, area, and the components of the two surface unit vectors, $\hat{t}_1$ and $\hat{t}_2$. The vector $\hat{t}_1$ is parallel to a side of the triangular, rectangular, or quadrilateral patch. For a patch of arbitrary shape, it is chosen by the following rules:

For a horizontal patch, $\hat{t}_1 = \hat{x}$;

For a nonhorizontal patch, $\hat{t}_1 = (\hat{z} \times \hat{n})/|\hat{z} \times \hat{n}|$;

$\hat{t}_2$ is then chosen as $\hat{t}_2 = \hat{n} \times \hat{t}_1$

With J = LD + 1 - I, the parameters for patch I are stored as follows.

X(J), Y(J), Z(J) = x, y, and z coordinates of the patch center

SI(J), ALP(J), BET(J) = x, y, z components of $\hat{t}_1$ (equivalenced to T1X, T1Y, T1Z)

ICON1(J), ICON2(J), ITAG(J) = x, y, and z components of $\hat{t}_2$ (equivalenced to T2X, T2Y, T2Z)

BI(J) = patch area

Scalar variables in /DATA/ are:

IPSYM = symmetry flag. The meanings of IPSYM are:

- IPSYM = 0: no symmetry
- IPSYM > 0: plane symmetry
- IPSYM < 0: cylindrical symmetry
- IPSYM = 2: plane symmetry about Z = 0
- |IPSYM| > 2: structure has been rotated about x or y axis. If ground plane is indicated by IGND ≠ 0 in the call to subroutine CONECT and IPSYM = 2, symmetry about a horizontal plane is removed by multiplying NP by 2. If |IPSYM| > 2 and IGND ≠ 0, all symmetry is removed by setting NP = N and IPSYM = 0 in CONECT.

LD	= length of arrays in /DATA/
N1	= number of segments in NGF. If NGF is not used N1 = 0
N2	= N1 + 1
N	= total number of segments
NP	= number of segments in a symmetric cell
M1	= number of patches in NGF. If NGF is not used M1 = 0
M2	= M1 + 1
M	= total number of patches
MP	= number of patches in a symmetric cell
WLAM	= wavelength in meters

<u>COMMON/DATAJ/</u> S, B, XJ, YJ, ZJ, CABJ, SABJ, SALPJ, EXK, EYK, EZK, EXS, EYS, EZS, EXC, EYC, EZC, RKH, IEXK, IND1, IND2, IPGND

<u>Routines Using /DATAJ/</u>

CMNGF, CMSET, CMSS, CMSW, CMWS, CMWW, EFLD, HINTG, HSFLD, NEFLD, NHFLD, PCINT, QDSRC, SFLDS, UNERE

/DATAJ/ is used to pass the parameters of the source segment or patch to the routines that compute the E or H field and to return the field components.

<u>/DATAJ/ Parameters for Wire Segments</u>

S	= segment length
B	= segment radius

XJ, YJ, ZJ = coordinates of segment center

CABJ, SABJ, SALPJ = x, y, and z, respectively, of the unit vector in the direction of the segment

EXK, EYK, EZK = x, y, and z components of the E or H field due to a constant current

EXS, EYS, EZS = x, y, and z components of the E or H field due to a sin ks current

EXC, EYC, EZC = x, y, and z components of the E or H field due to cos ks current

RKH = minimum distance for use of the Hertzian dipole approximation for computing the E field of a segment

IEXK = flag to select thin wire approximation or extended thin wire approximation for E field (IEXK = 1 for extended thin wire approximation)

IND1 = flag to inhibit use of the extended thin wire approximation on end 1 of the source segment. This is used when there is a bend or change in radius at end 1. IND1 = 2 inhibits the extended thin wire approximation.

IND2 = flag to inhibit use of the extended thin wire approximation on end 2 of the source segment

IPGND = not used

/DATAJ/ Parameters for Surface Patches

S = patch area in units of wavelength squared

B = x component of $\hat{t}_2$ for the patch

XJ, YJ, ZJ = x, y, and z components of the position of the patch center

CABJ, SABJ, SALPJ = x, y, and z components of $\hat{t}_1$

EXK, EYK, EZK = x, y, and z components of $\bar{E}$ or $\bar{H}$ due to a current with unit magnitude in the direction $\hat{t}_1$ on the patch

EXS, EYS, EZS = $\bar{E}$ or $\bar{H}$ due to a current $\hat{t}_2$ on the patch

EXC, EYC, EZC = not used; may serve as intermediate variables in some routines

IND1 = y component of $\hat{t}_2$

IND2 = z component of $\hat{t}_2$

IPGND = flag to request calculation of the direct field or field reflected from the ground (two for ground)

COMMON/FPAT/ NTH, NPH, IPD, IAVP, INOR, IAX, THETS, PHIS, DTH, DPH, RFLD, GNOR, CLT, CHT, EPSR2, SIG2, IXTYP, XPR6, PINR, PNLR, PLOSS NEAR, NFEH, NRX, NRY, NRZ, XNR, YNR, ZNR, DXNR, DYNR, DZNR

Routines Using /FPAT/

MAIN, NFPAT, RDPAT

Variables are defined in subroutine descriptions.

COMMON/GGRID/ AR1(11, 10, 4), AR2(17, 5, 4), AR3(9, 8, 4), EPSCF, DXA(3), DYA(3), XSA(3), YSA(3), NXA(3), NYA(3)

Routines Using /GGRID/

MAIN, GFIL, GFOUT, INTRP

Variables are defined under subroutine INTRP.

COMMON/GND/ ZRATI, ZRATI2, FRATI, CL, CH, SCRWL, SCRWR, NRADL, KSYMP, IFAR, IPERF, T1, T2

Routines Using /GND/

MAIN, CMSW, EFLD, ETMNS, FFLD, GFIL, GFOUT, HINTG, HSFLD, NEFLD, RDPAT, SFLDS, UNERE

/GND/ contains parameters of the ground including the two-medium ground and radial-wire ground-screen cases. The symbol definitions are as follows.

ZRATI = $[\epsilon_r - j\sigma/\omega\epsilon_0]^{-1/2}$

where σ is ground conductivity (mhos/meter), ϵ_r is the relative dielectric constant, ϵ_0 is the permittivity of free space (farads/meter), and $\omega = 2\pi f$.

ZRATI2 = same as ZRATI, but for a second ground medium

FRATI = $(k_1^2 - k_2^2)/(k_1^2 + k_2^2)$ where $k_2 = \omega \sqrt{\mu_0 \epsilon_0}$ and $k_1 = k_2/\text{ZRATI}$

CL = distance in wavelengths of cliff edge from origin

CH = cliff height in wavelengths

SCRAWL = length of wires in radial-wire ground screen (normalized to wavelength)

SCRWR = radius of wires in screen in wavelengths

NRADL = number of radials in ground screen; zero implies no screen (input quantity, GN card)

KSYMP = ground flag (=1, no ground; =2, ground present)

IFAR = input integer flag on RP card; specifies type of field computation or type of ground system for far fields

IPERF = flag to select type of ground (see GN card)

T1, T2 = constants for the radial-wire ground-screen impedance

COMMON/GWAV/ U, U2, XX1, XX2, R1, R2, ZMH, ZPH

Routines Using /GWAV/

MAIN, GFLD, GWAVE, SFLDS

Symbol Definitions:

U = $(\epsilon_r - j\sigma/\omega\epsilon_0)^{-1/2}$

ϵ_r = relative dielectric constant; σ = conductivity of ground

U2 = U^2

XX1, XX2 : defined in GFLD and SFLDS

R1 = distance from current element to point at which field is evaluated

R2 = distance from image of current element to point at which field is evaluated

ZMH = Z - Z'

ZPH = Z + Z' where Z is height of the field evaluation point and Z' is the height of the current element

COMMON/INCOM/ XO, YO, ZO, SN XSN, YSN, ISNOR

Routines Using /INCOM/

EFLD, SFLDS

Symbol Definitions:

XO, YO, ZO = point at which field due to ground will be evaluated

SN = cos α (see Figure 11)

XSN = cos ß

YSN = sin ß

ISNOR = 1 to evaluate field due to ground by interpolation
0 to use Norton's approximation

COMMON/MATPAR/ ICASE, NBLOKS, NPBLK, NLAST, NBLSYM, NPSYM, NLSYM, IMAT, ICASX, NBBX, NPBX, NLBX, NBBL NPBL, NLBL

Routines Using /MATPAR/

MAIN, CMNGF, CMSET, FACGF, FACIO, FACTRS, FBLOCK, FBNGF, GFIL, GFOUT, LFACTR, LTSOLV, LUNSCR, REBLK, SOLGF, SOLVES

/MATPAR/ contains matrix blocking parameters for cases requiring file storage of the matrix. Symbol definitions in /MATPAR/ are as follows.

ICASE = storage mode for primary matrix, defined as follows.

1 unsymmetric matrix fits in core

2 symmetric matrix fits in core

3 unsymmetric matrix out of core

4 symmetric matrix out of core, but submatrices fit in core

5 symmetric matrix out of core, submatrices also out of core

NBLOKS = number of blocks of columns of the computed matrix (in core matrix, NBLOKS = 1)

NPBLK = number of columns in the first (NBLOKS - 1) blocks

NLAST = number of columns in the last block

NBLSYM, NPSYM, NLSYM = same function as the preceding three variables; however, in this case the parameters refer to the submatrix in the symmetry case

IMAT = storage reserved in CM for the primary NGF matrix A or a block of A (number of complex numbers)

ICASX = storage mode for NGF solution (see Section VII)

NBBX = number of blocks in matrix B stored by blocks of rows

NPBX = number of rows in a block of B stored by rows

NLBX = number of rows in the last block of B

NBBL = number of blocks in matrix C stored by rows (and number of blocks in B stored by columns)

NPBL = number of rows (columns) in a block of C (B)

NLBL = number of rows (columns) in the last block of C (B)

COMMON/NETCX/ ZPED, PIN, PNLS, NEQ, NPEQ, NEQ2, NONET, NTSOL, NPRINT, MASYM, ISEG1(30), ISEG2(30), X11R(30), X11I(30), X12R(30), X12I(30), X22R(30), X22I(30), NTYP(30)

Routines Using /NETCX/

MAIN, NETWK

Variables are defined under subroutine NETWK.

COMMON/SAVE/ IP(600), KCOM, COM(13,5), EPSR, SIG, SCRWLT, SCRWRT, FMHZ

Routines Using /SAVE/

MAIN, GFIL, GFOUT, RDPAT

Symbol Definitions:

IP = vector of indices of pivot elements used in factoring the matrix

KCOM = number of CM or CE data cards (maximum 5)

COM = array storing the contents of CM or CE cards

EPSR = relative dielectric constant of the ground

SIG = conductivity of the ground

SCRWLT = length of radials in radial wire ground screen approximation (meters)

SCRWRT = radius of wires in radial wire ground screen approximation (meters)

FMHZ = frequency in MHz

COMMON/SCRATM/D(600)

in routines CMSET, FACTR, LFACTR

COMMON/SCRATM/Y(600)

in routines LTSOLV, SOLGF, SOLVE, SOLVES

COMMON/SCRATM/GAIN(1200)

in routine RDPAT

Symbol Definitions:

D and Y =
complex vectors used in matrix decomposition and solution

GAIN = array to store antenna gain for subsequent normalization

COMMON/SEGJ/ AX(30), BX(30), CX(30), JCO(30), JSNO, ISCON(50), NSCON, IPCON(10), NPCON

Routines Using /SEGJ/

MAIN, CABC, CMNGF, CMSET, CMSW, CMWS, CMWW, CONECT, QDSRC, SFLDS, TBF, TRIO

/SEGJ/ is used to store the parameters defining current expansion functions. The equations for the current expansion functions are given in Section III-1 of Part I. The i^{th} current expansion function consists of a center section on segment i and branches on each segment connected to segment i. On segment j, where j is i or the number of a segment connected to segment i, the i^{th} expansion function is

$$f_j^i(s) = A_j^i + B_j^i \sin [k(s - s_j)] + C_j^i \cos [k(s - s_j)]$$

with the constants defined in Part I to match conditions on the current. A superscript i has been added to indicate the number of the current expansion function.

When subroutine TBF is called for expansion function i, it locates each segment connected to segment i and stores the segment number, j, in array JCO. TBF also computes the constants A_j^i, B_j^i, and C_j^i for segment j and stores them in AX, BX, and CX, respectively.

After all connected segments have been found, i is stored in the next location in JCO, and A_i^i, B_i^i, and C_i^i are stored in the corresponding locations in AX, BX, and CX.

/SEGJ/ is also used by subroutine TRIO. When TRIO is called for segment j, it locates each segment i connected to segment j and stores i in array JCO. TRIO calls SBF to compute the constants A_j^i, B_j^i, and C_j^i for the branch of expansion function i that extends onto segment j and stores these in AX, BX, and CX. The total number of entries, including i = j, is stored in JSNO. The remaining parameters are used with the NGF solution.

ISCON(I) = number of the segment in the NGF file having equation number I in the set of equations for modified basis functions. This is used when a new segment or patch connects to the NGF segment

NSCON = number of entries in ISCON

IPCON(I) = number of the patch in the NGF file having equation number I in the set of equations for modified patch basis functions. This is used when a new segment connects to the NGF patch

NPCON = number of entries in IPCON

COMMON/SMAT/ SSX(16,16)

Routines Using /SMAT/

CMSET, FBLOCK GFIL, GFOUT, SOLVES

The array SSX is described under subroutine FBLOCK. In some copies of NEC-2 the variable name S is used in FBLOCK rather than SSX.

COMMON/TMH/ ZPK, RHKS

Routines Using /TMH/

GH, HFK

/TMH/ is used to pass values from HFK to GH. The variables ZPK and RHKS are defined in the discussion of subroutine HFK.

COMMON/TMI/ ZPK, RKB2, IJX

Routines Using /TMI/

EKSC, EKSCX, GF

/TMI/ is used to pass values from EKSC or EKSCX to GF. The meanings of the variables are listed in subroutines EKSC and EKSCX.

COMMON/VSORC/ VQD(10), VSANT(10), VQDS(10), IVQD(10), ISANT(10), IQDS(10), NVQD, NSANT, NQDS

Routines Using /VSORC/

MAIN, CABC, COUPLE, ETMNS, NETWK, QDSRC

The arrays in /VSORC/ contain the strengths and locations of voltage sources on wires. Separate arrays are used for applied-field voltage sources and current-derivative discontinuity voltage sources. The variables are defined as follows.

ISANT(I) = number of the segment on which the I^{th} applied-field source is located

IVQD(I) = IQDS(I) = number of the segment on end 1 of which the I^{th} current-slope discontinuity voltage source is located

VQD(I) = VQDS(I) = voltage of the I^{th} current-slope discontinuity source

VSANT(I) = voltage of the I^{th} applied-field voltage source

NSANT = number of applied-field voltage sources

NVQD = NQDS = number of current-slope discontinuity voltage sources

NVQD, IVQD, and VQD are set in MAIN from the input data. NQDS, IQDS, and VQDS are set in subroutine QDSRC. The latter were included to allow for current-slope discontinuities other than voltage sources, such as lumped loads. Loading by this means has not been implemented in NEC-2 however.

COMMON/YPARM/ NCOUP, ICOUP, NCTAG(5), NCSEG(5), Y11A(5), Y12A(20)

Routines Using /YPARM/

MAIN, COUPLE

Symbol Definitions:

NCOUP = number of segments between which coupling will be computed

ICOUP = number of segments in the coupling array that have been excited. When ICOUP = NCOUP subroutine COUPLE completes the coupling calculation

NCTAG(I) = tag number of segment I

NCSEG(I) = number of segment in set of segments having tag NCTAG(I)

Y11A(I) = self admittance of I^{th} segment specified

Y12A(I) = mutual admittances stored in order (1,2), (1,3), ... (2,3), (2,4), ... etc.

COMMON/ZLOAD/ ZARRAY(300)

Routines Using /ZLOAD/

MAIN, CMNGF, CMSET, GFIL, GFOUT, LOAD, QDSRC

ZARRAY(I) = $Z_I/(\Delta_I/\lambda)$, where Z_I is the total impedance on segment I, Δ_I is the length of segment I, and λ is the wavelength.

Section IV
System Library Functions Used by NEC

ABS(X)	= absolute value of X
AIMAG(Z)	= imaginary part of the complex number Z; result is real
AINT(X)	= integer truncation; result is real
ALOG(X)	= natural log of X
ALOG10(X)	= log to the base ten of X
ASIN(X)	= arcsine of X; result in radians
ATAN(X)	= arctangent of X; result in radians
ATAN2(X_1,X_2)	= arctangent of X_1/X_2; result in radians covering all four quadrants
CABS(Z)	= magnitude of the complex number, Z
CEXP(Z)	= complex exponential (e^Z)
CMPLX(X_1,X_2)	= formation of a complex number, $Z = X_1 + jX_2$
CONJG(Z)	= conjugate of the complex number Z
COS(X)	= cosine of X
CSQRT(Z)	= square root of a complex number, $\sqrt{Z}$
FLOAT(K)	= real number equivalent of integer K
IABS(K)	= absolute value of integer K
INT(X)	= X truncated to an integer
REAL(Z)	= real part of the complex number Z
SIN(X)	= sine of X
SQRT(X)	= square root of X
TAN(X)	= tangent of X

Section V
Array Dimension Limitations

Array dimensions in the program limit the structure model in various ways. Any of these limits may be increased if necessary at the expense of core storage capacity, which may require reducing other array dimensions. The limits imposed by array dimensions are described below.

In-Core Matrix Storage, I_r = 4000.

Arrays:

COMMON/CMB/ CM(I_r)

Limit constant:

IRESRV = I_r at MA68 of MAIN

I_r is the number of words of core available for storage of the interaction matrix. The complete matrix will fit in core storage if (N + 2M) × (NP + 2MP) is not greater than I_r. For out-of-core solution, I_r must be at least 2(N + 2M) and should be as large as possible to minimize file manipulation.

Maximum Segments and Patches

Minimum Dimensions for N segments and M patches:

COMMON/DATA/ X(N + M), Y(N + M), Z(N + M), SI(N + M), BI(N + M), ALP(N + M), BET(N + M), ICON1(N + M), ICON2(N + M), ITAG(N + M), ICONX(N + M)

COMMON/CRNT/AIR (N), AII(N), BIR(N), BII(N), CIR(N), CII(N), CUR(N + 3M)

COMMON/ANGL/ SALP(N + M)

COMMON/SAVE/ IP(N + 2M)

COMMON/ZLOAD/ ZARRAY(N)

COMMON/SCRATM/ D(N + 2M) or Y(N + 2M)

MAIN: IX(N + 2M)

SUBROUTINE NETWK: RHS(N + 3M)

Limit Constant:

LD = N + M at MA66 of MAIN

All segments and patches resulting from reflection or rotation of a symmetric structure must be included in determining the limiting structure size.

<u>Maximum Number of Non-radiating Networks, N_n = 30.</u>

Arrays:

COMMON/NETCX/: ISEG1(N_n), ISEG2(N_n), X11R(N_n), X11I(N_n), X12R(N_n), X12I(N_n), X22R(N_n), X22I(N_n), NTYP(N_n)

SUBROUTINE NETWK: RHNT(N_n), IPNT(N_n), NTEQA(N_n), NTSCA(N_n), RHNX(N_n), CMN(N_n,N_n)

Limit Constants:

NETMX = N_n at MA63 of MAIN

NDIMN = N_n at NT22 of NETWK

NDIMNP = N_n + 1 at NT22 of NETWK

N_n is the limit for either the number of networks (including transmission lines) or the number of segments having one or more network ports connected, whichever is greater. When relative driving point matrix asymmetry is computed, N_n must also be greater than or equal to the sum of the number of segments with network ports connected plus the number of segments with voltage sources.

Maximum Number of Degrees of Symmetry, N_p = 16.

Arrays:

COMMON/SMAT/ S(N_p, N_p)

N_p limits the number of symmetric cells in a structure. The number of symmetric cells is equal to the ratio of N to NP in COMMON/DATA/.

Maximum Number of Segments Joined at Junctions, N_j = 30

If N^- and N^+ are the numbers of segments connected to end 1 and end 2 of a segment, respectively, then the dimensions in COMMON/SEGJ/, N_j, must be at least $N^- + N^+ + 1$.

Array:

COMMON/SEGJ/ AX(N_j), BX(N_j), CX(N_j), JCO(N_j), JSNO

Limit Constants:

JMAX = N_j at SB6 in SBF

JMAX = N_j at TB8 in TBF

JMAX = N_j at TR8 in TRIO

Maximum Number of Voltage Sources, N_v = 30.

Arrays:

COMMON/VSORC/ VQD(N_v), VSANT(N_v), VQDS(N_v), IVQD(N_v), ISANT(N_v), IQDS(N_v)

Limit Constant:

NSMAX = N_v at MA63 of MAIN

A model may use up to N_v applied field voltage sources and up to N_v current slope discontinuity voltage sources.

Maximum Number of Loading Cards, $N_1 = 30$

Arrays:

MAIN: LDTYP(N_1), LDTAG(N_1), LDTAGF(N_1), LDTAGT(N_1), ZLR(N_1), ZLI(N_1), ZLC(N_1)

Limit Constants:

LOADMX = N_1 at MA63 of MAIN

When the NGF option is used only new loading cards are counted, not those used in generating the NGF file.

Number of Comment Cards Saved, $N_c = 5$

Arrays:

COMMON/SAVE/: COM(13,N_c)

Limit Constant:

Constants at MA71 of MAIN

Any number of comment cards may be placed at the beginning of a data deck and will be printed in the output. Only N_c of the cards will be saved in array COM for later use in labeling plots, however. The first $N_c - 1$ comment cards and the last comment card will be saved.

Maximum Field Points for Normalized Gain, $N_g = 1200$.

Arrays:

COMMON/SCRATM/ GAIN(N_g)

Limit Constant:

NORMAX = N_g at RD22 of SUBROUTINE RDPAT

N_g is the maximum number of field points from a single RP data card that can be stored for output in normalized form or for plotting if plotting is implemented. If an RP card requesting more than N_g points calls for normalized gain, the gain will be computed and printed at all requested angles, but only the first N_g gains will be stored and normalized.

COMMON/SCRATM/ GAIN occurs in SUBROUTINE RDPAT. COMMON/SCRATM/ D and COMMON/SCRATM/ Y occur in certain other routines where D and Y are complex (see "Maximum Segments and Patches"). GAIN, D, and Y should be dimensioned so that each common statement contains the same number of words.

Maximum Number of Frequencies for Normalized Impedance or Maximum Number of Angles for Which Received Signal Strength Is Stored, N_f = 200

Array:

MAIN: FNORM(N_f)

Limit Constant:

NORMF = N_f at MA63 of MAIN

The maximum number of frequencies for which input impedance may be stored and normalized is $N_f/4$, since the real and imaginary impedance and magnitude and phase are each stored. The receiving current can be stored for up to N_f angles.

Maximum Number of Points in Coupling Calculation, $N_c = 5$.

The maximum number of segments among which coupling can be computed (CP cards) is N_c.

COMMON/YPARM/: NCTAG(N_c), NCSEG(N_c), Y11A(N_c), Y12A($N_c^2 - N_c$)

Limit Constants:

Constants at MA207 and MA212 of MAIN should equal N_c

Maximum Number of NGF Segments to Which New Segments or Patches Connect, $N_s = 50$

COMMON/SEGJ/: ISCON(N_s)

Limit Constant:

NSMAX = N_s at CN13 of CONECT

Maximum Number of NGF Patches to Which New Segments Connect, $N_p = 10$.

COMMON/SEGJ/: IPCON(N_p)

Limit Constant:

NPMAX = N_p at CN13 of CONECT

Section VI
Overview of Numerical Green's Function Operation

NEC includes a provision to generate and factor an interaction matrix and save the result on a file. A later run, using the file, may add to the structure and solve the complete model without unnecessary repetition of calculations. This procedure is called the Numerical Green's Function (NGF) option since the effect is as if the free space Green's function in NEC were replaced by the Green's function for the structure on the file. The NGF is particularly useful for a large structure, such as a ship, on which various antennas will be added or modified. It also permits taking advantage of partial symmetry since a NGF file may be written for the symmetric part of a structure, taking advantage of the symmetry to reduce computation time. Unsymmetric parts can then be added in a later run.

For the NGF solution the matrix is partitioned as

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \end{bmatrix} = \begin{bmatrix} E_1 \\ E_2 \end{bmatrix} ,$$

where A is the interaction matrix for the initial structure, D is the matrix for the added structure, and B and C represent mutual interactions. The current is computed as

$$I_2 = \left[D - CA^{-1}B\right]^{-1} \left[E_2 - CA^{-1}E_1\right] ,$$

$$I_1 = A^{-1}E_1 - A^{-1}BI_2 ,$$

after the factored matrix A has been read from the NGF file along with other necessary data. Since the LU decomposition is obtained in NEC rather than the inverse, the multiplication by A^{-1} is accomplished by using the solution procedure in subroutine SOLVE on each column in the matrix to the right of A^{-1}.

To use the NGF option the parameters of the fixed, or NGF, part of the model are defined in the first run. A WG data card causes the matrix A to be computed (CMSET), factored (FACTRS), and written to file TAPE20 by subroutine

GFOUT. Other necessary data, such as segment and patch coordinates, frequency, loading, and ground parameters, are also written to TAPE20.

When the NGF file, TAPE20, is used the data are read into the usual arrays by subroutine GFIL and new segments and patches are added to the arrays in COMMON/DATA/. Subroutine CMNGF is then called to compute the matrix elements in B, C, and D. FACGF computes $A^{-1}B$, storing it in place of B, and computes $(D - CA^{-1}B)$, factors it into L and U parts, and stores the result in place of D. For each excitation E_1 and E_2, SOLGF completes the procedure of solving for I_1 and I_2.

The procedure is complicated by the connection of new segments or patches to NGF segments or patches. A connection to a segment modifies the current basis function (see Section III.1 of Part I). Since the elements in A cannot be changed, a modified basis function must be treated as a new basis function with a new column added to B and D and the new basis function amplitude added to the end of I_2. The amplitude of the original basis function is set to zero by adding a row containing all zeros except for a one in the column of C corresponding to the modified basis function. Since the segment is not modified the boundary condition equation is not altered in this case.

When a new segment connects to a NGF patch the patch must be divided into four new patches, after the user defined patches, requiring eight new rows and columns in B, C, and D. Two additional rows are added to set the two current components on the old patch to zero. Since the old patch is replaced by the four new patches, the condition on the field at the center of the patch should be removed. This is done by adding two new columns each containing all zeros except for a one in the row of the equation to be removed.

The matrix structure is further complicated by the division of each submatrix into sections for segment-to-segment, patch-to-patch, segment-to-patch and patch-to-segment interactions. The matrix structure is shown in Figure 12, where the subscript w denotes wire segments and s denotes surface patches. The elements of B'_{ww} and B'_{sw} are the E fields and H fields due to modified basis functions in the NGF section. Each column of B'_{ss} and row of C'_{ww} and C'_{ss} contains 0's and a single 1.

The subroutine ETMNS fills the excitation array with the E fields illuminating all segments, followed by the H fields on patches. These elements are reordered in SOLGF to correspond to the matrix structure. After

A_{ww}	A_{ws}	B_{ww}	B_{ws}	B'_{ww}	0
A_{sw}	A_{ss}	B_{sw}	B_{ss}	B'_{sw}	B'_{ss}
C_{ww}	C_{ws}	D_{ww}	D_{ws}	D'_{ww}	0
C_{sw}	C_{ss}	D_{sw}	D_{ss}	D'_{sw}	0
C'_{ww}	0	0	0	0	0
0	C'_{ss}	0	0	0	0

Figure 12. Matrix Structure for the NGF Solution

the solution this reordering is reversed in SOLGF to put basis function amplitudes for segments first, followed by those for patches. If symmetry is used in the NGF section the matrix A is structured as submatrices for the symmetric sections. Each submatrix contains elements for segments and patches in that section, with the order as shown for A in Figure 12. In this case the excitation and solution vectors are ordered in SOLVES to correspond to the submatrix structure.

Section VII
Overview of Matrix Operations Using File Storage

File storage is used when the matrix size exceeds the length of the array CM in COMMON/CMB/. For the basic solution (not NGF) there are five matrix storage modes associated with the integer ICASE as follows:

ICASE	Matrix Storage
1	Matrix fits in CM; no structure symmetry
2	Matrix fits in CM; structure symmetry used
3	Matrix stored on file; no symmetry
4	Matrix stored on file; symmetry; each submatrix fits into CM for LU decomposition
5	Matrix stored on file; symmetry; submatrices do not fit into CM.

For case 3 the matrix is initially written on file 11 by blocks of rows. The block size is chosen in subroutine FBLOCK so that two blocks will fit into CM for the Gauss elimination procedure. The block size and number of blocks is set by the parameters NBLOKS, NPBLK, and NLAST in COMMON/MATPAR/.

Subroutine FACIO reads file 11 and writes file 12 using 13 and 14 for scratch storage. LUNSCR then reads 12 and writes the blocks of the factored matrix on file 13 in forward order and on file 14 in reversed order. File 13 is then used for forward substitution in the solution and file 14 is used for backward substitution.

For case 4, FACTRS reads the matrix from file 11, where it was written by blocks of rows (columns of the transposed matrix), and writes it to file 12 by submatrices. The submatrices are then read from 12, factored, and written to 13.

In case 5, FACTRS reads the matrix from file 11 and writes it to file 12 by blocks of rows (columns of the transposed matrix) for each submatrix. File 12 is then copied back to file 11, and the procedure of case 3 is repeated for each submatrix.

When a NGF file is to be written, half of CM is reserved for matrix storage and manipulations of the matrices B, C, and D. Hence for cases 1, 2 or 4 the primary matrix A (or submatrix for case 4) must fit into half of CM.

There is no restriction for cases 3 or 5 since, with two matrix blocks fitting into CM for the LU decomposition, half of CM is available during the solution when blocks are used one at a time.

There are four modes for storing B, C, and D in the NGF solution. These are associated with the integer ICASX as follows:

A_F = matrix A factored into L and U

$$A_R = \begin{cases} A_F \text{ for ICASE} = 1 \text{ or } 2 \\ \text{one block of } A_F \text{ for ICASE} = 3 \\ \text{one submatrix for ICASE} = 4 \\ \text{one block of submatrix for ICASE} = 5 \end{cases}$$

A_X = A_F for ICASE = 1 or 2
nothing otherwise

ICASX	NGF Matrix Storage
1	A_R, B, C, and D fit into CM
2	B, C, and D fit into CM but not with A_R (ICASE = 3, 4, 5) A_R and B must also fit into CM together
3	B, C, and D do not fit into CM, but A_X and $F = D - CA^{-1}B$ fit into CM for the LU decomposition of F
4	Same as 3 but $D - CA^{-1}B$ requires file storage for LU decomposition

When a NGF file (TAPE20) is written with ICASE = 3 or 5, files 13 and 14 are both written to TAPE20. When the NGF file is read these data are written on the single file 13 with the blocks in ascending order first and then in descending order. If A_F is stored on file 13 then space for A_R in CM is needed only when A_R is used in a solution in CM. This accounts for the definition of A_X.

File usage for ICASX = 2, 3, and 4 is outlined in Figures 13 and 14. The value for ICASX is chosen in subroutine FBNGF as the smallest value possible. The number of blocks into which matrices B, C, and D are divided is also chosen in FBNGF.

NGF Procedure for ICASX = 2	Contents of CM	Files 11	12	13	14	15	16
				A_F			
1. (CMNGF) Compute B, C and D in CM. Write to files 12, 14 and 15.	B, C, D		D	A_F	B	C	
2. (FACGF) Read 13 and 14. Compute $A^{-1}B$. Write 14.	A_F, B		D	A_F	$A^{-1}B$	C	
3. Compute $F = D - CA^{-1}B$. Store over D in CM.	$A^{-1}B$, C, D		D	A_F	$A^{-1}B$	C	
4. Factor F. Write on 11.	$A^{-1}B$, C, F	F_F		A_F	$A^{-1}B$	C	
Solution for excitation $(E_1, E_2)^T$ (SOLGF)							
1. Compute $I_1' = A^{-1}E_1$	A_F	F_F		A_F	$A^{-1}B$	C	
2. $I_2' = E_2 - CA^{-1}E_1 = E_2 - CI_1'$	C						
3. $I_2 = F^{-1}I_2'$	F_F						
4. $I_1 = I_1' - (A^{-1}B)I_2$	$A^{-1}B$						

(Subscript F indicates that the matrix has been factored into L and U triangular parts)

Figure 13. NGF File Usage for ICASX = 2.

NGF Procedure for ICASX = 3,4	Contents of CM	11	12	13	14	15	16
				A_F			
1. (CMNGF) Compute B by blocks of rows. Write to file 14.	A_X, B_R			A_F	B_R		
2. Compute C and D by blocks by rows. Write to 15 and 12.	A_X, C_R, D_R		D_R	A_F	B_R	C_R	
3. (REBLK) Read 14. Write B by blocks of columns on file 16.	A_X, B_C, B_R		D_R	A_F	B_R	C_R	B_C
4. (FACGF) Read 16; compute $A^{-1}B$; write 14.	A_F, B_C		D_R	A_F	$A^{-1}B_C$	C_R	
5. Read blocks from 12, 14 and 15 and compute $F = D - C(A^{-1}B)$ by blocks of rows. Write on 11.	A_X, $A_{-1}B_C$ C_R, D_R	F	D_R	A_F	$A^{-1}B_C$	C_R	
6. For ICASX = 4 call FACIO to factor F. FACIO reads 11 and writes 12, using 11 and 16 as scratch storage.	A_X and 2 blocks of F	X	F_F'	A_F	$A^{-1}B_C$	C_R	X
7. LUNSCR reads 12 and writes blocks of FF on 111 in forward order and on 16 in reversed order.	A_X and block of F	F_F		A_F	$A^{-1}B_C$	C_R	F_F
6' For ICASX = 3, read all blocks of F into CM; Factor F; write to 11.	A_X, F	F_F		A_F	$A^{-1}B_C$	C_R	

Figure 14. NGF File Usage for ICASX = 3 or 4.

Section VIII
NEC Subroutine Linkage

Figures 15 and 16 show the organization of subroutines in the NEC-2 program. All possible subroutine calls are traced, although in a particular run only certain of the traces will be followed. Routines that are called at more than one point in the program are shown as separate blocks for each call.

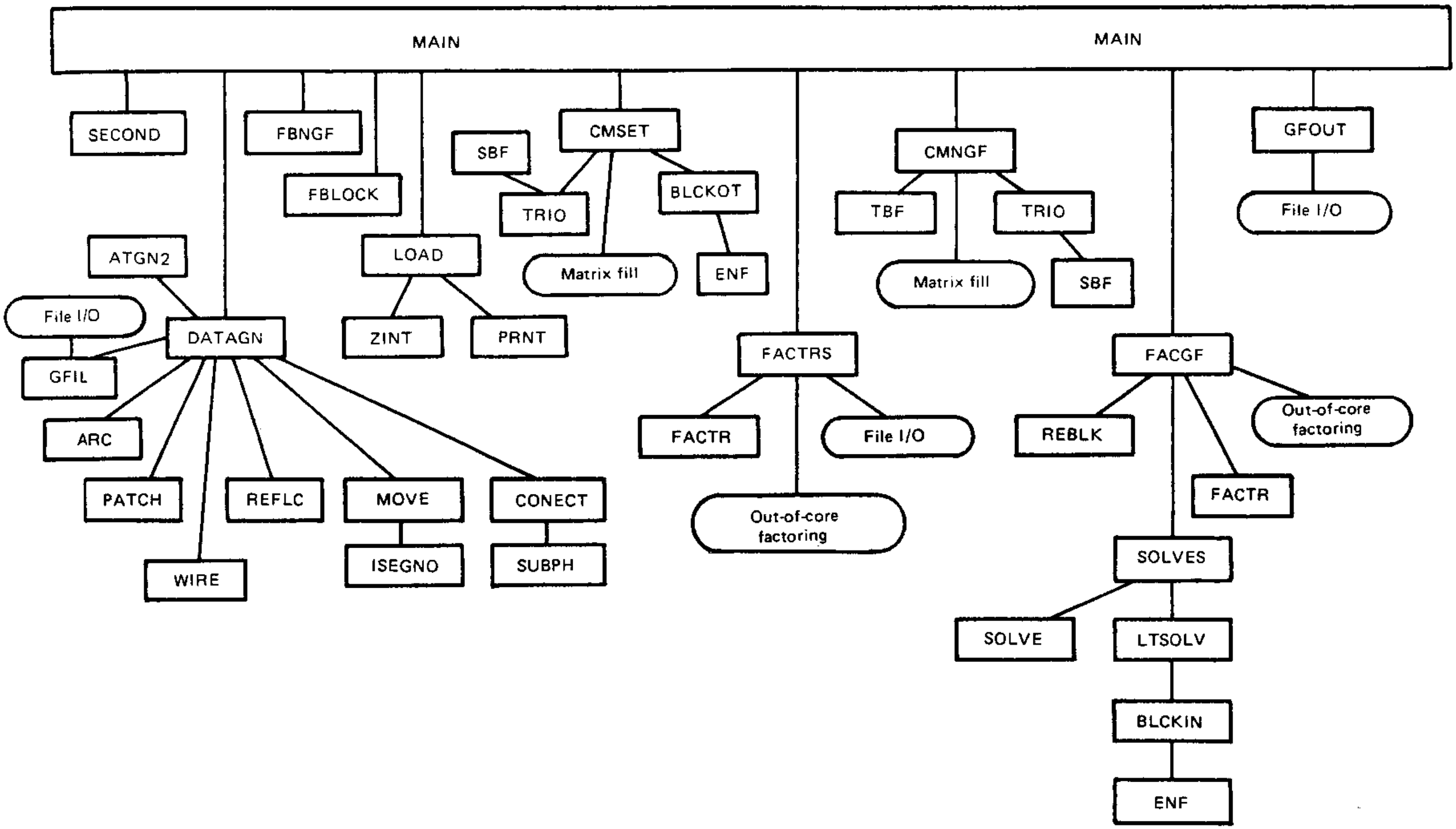


Figure 15. NEC Subroutine Linkage Chart. For Block Definitions, see Figure 16

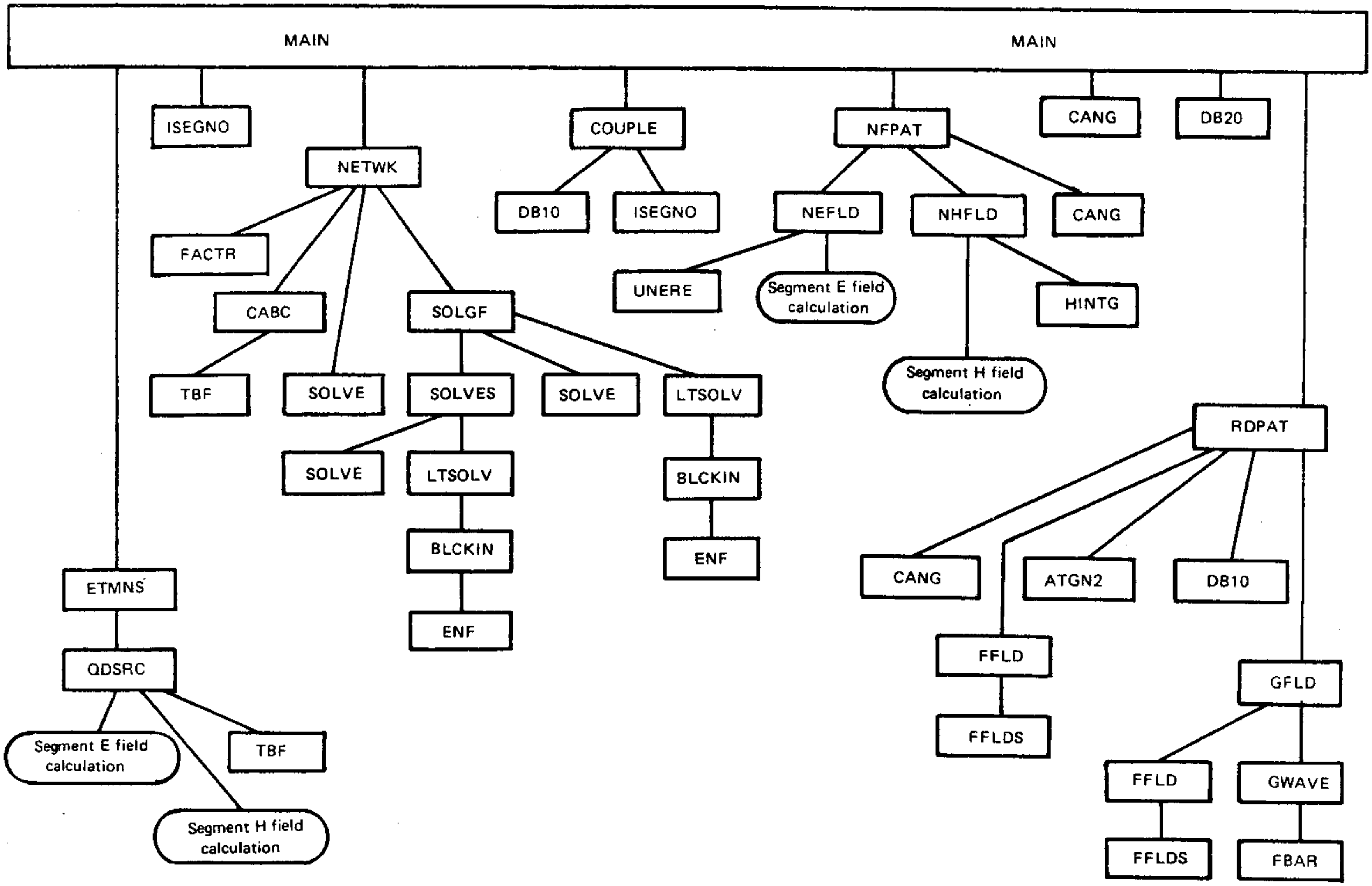


Figure 15 (continued)

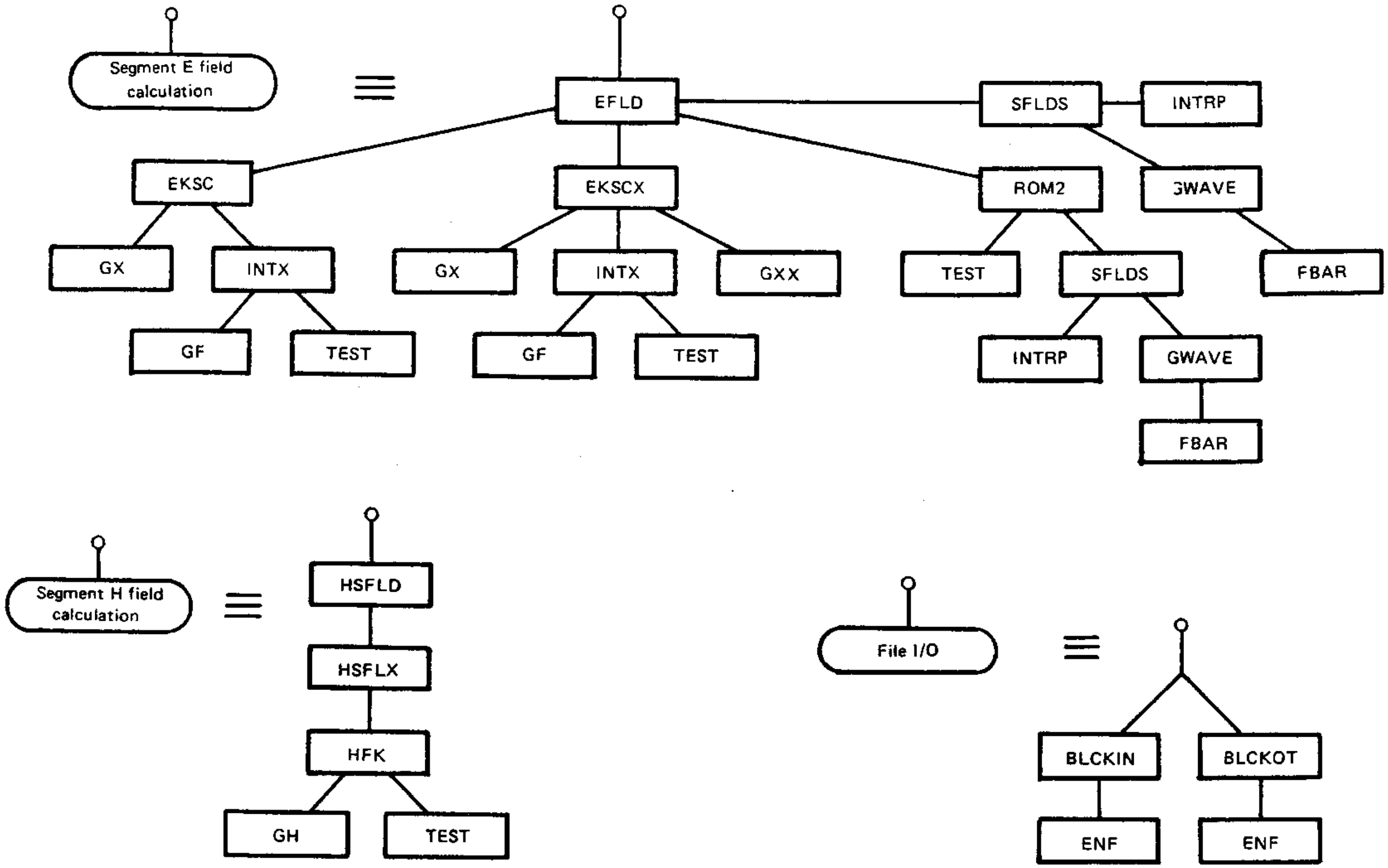


Figure 16. Block Definitions for NEC Subroutine Linkage Chart. See Figure 15

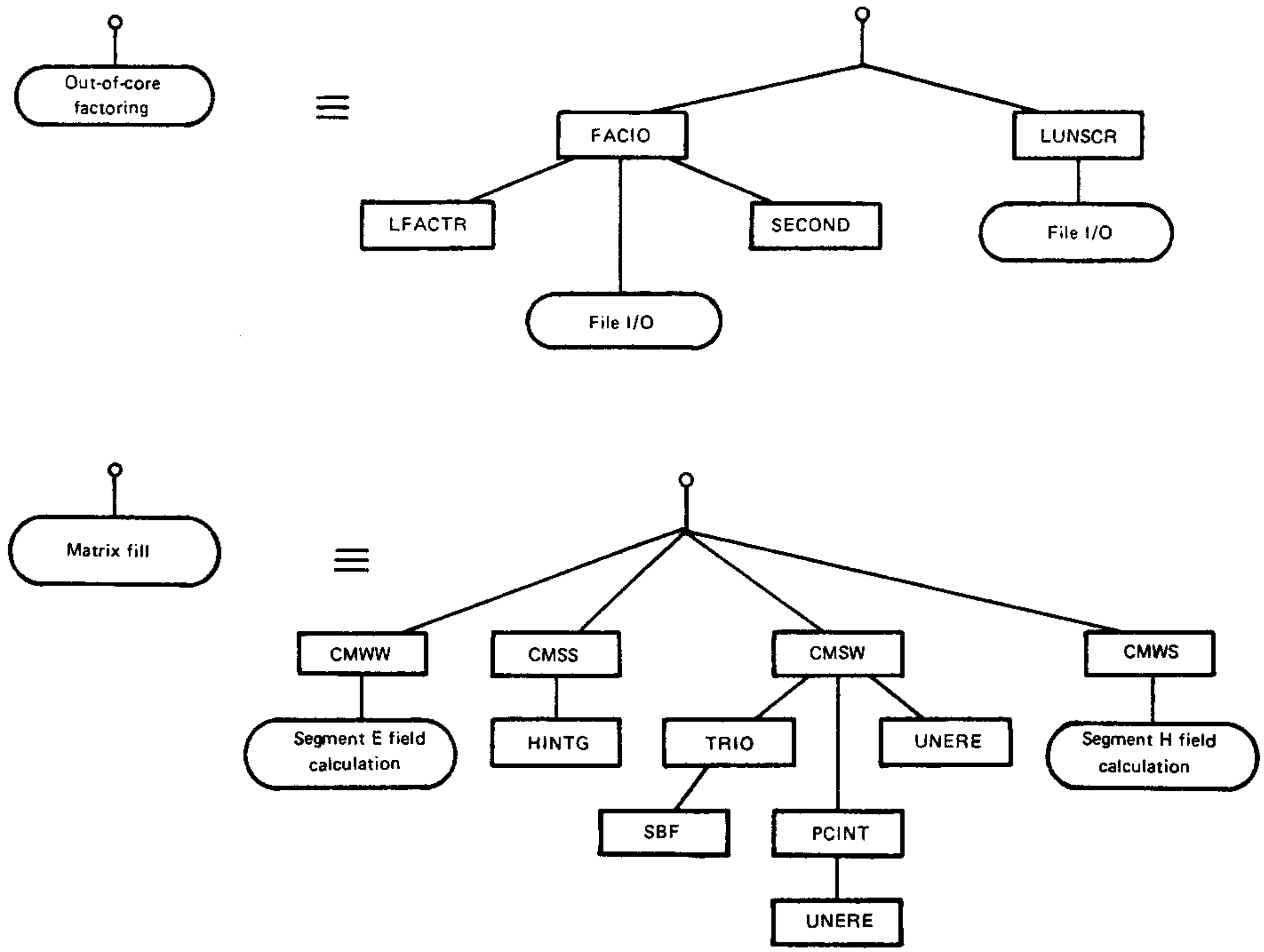


Figure 16 (continued)

Section IX
SOMNEC

1. SOMNEC CODE DESCRIPTION

SOMNEC is an independent code that generates the interpolation tables for the Sommerfeld/Norton ground option for NEC. The tables are written on file TAPE21 which becomes an input file to NEC. Coding of the routines in SOMNEC is described in this section.

SOMNEC (main program)

PURPOSE

To generate interpolation tables for the Sommerfeld/Norton ground option and write them on file TAPE21.

METHOD

The code from SN17 to SN51 reads the input data and sets parameters in COMMON/EVLCOM/. Since all equations are scaled to a free-space wavelength of one meter the results depend only on the complex dielectric constant

$$\varepsilon_c = \varepsilon_1 - j\sigma_1/(\omega\varepsilon_0) \ .$$

In the routines that evaluate the Sommerfeld integrals the time dependence is exp(-jωt) rather than exp(+jωt) which is used in the remainder of NEC. Hence the conjugate of ε_c (EPSCF) is taken before computing the parameters in COMMON/EVLCOM/. The conjugate of the results is taken at the end of EVLUA, so the results returned to SOMNEC and written on TAPE21 are for exp(+jωt).

Three interpolation tables, as shown in Figure 12 of Part I, are generated in the code from SN55 to SN123. For each R_1, θ pair in the tables the values of ρ and z + z' are computed and stored in COMMON/EVLCOM/. Subroutine EVLUA is then called and returns the quantities

$$\text{ERV} = \frac{\partial^2}{\partial\rho\partial z} k_1^2 V'_{22}$$

$$\text{EZV} = \left(\frac{\partial^2}{\partial z^2} + k_2^2\right) k_1^2 V'_{22}$$

$$\text{ERH} = \left(\frac{\partial^2}{\partial\rho^2} k_2^2 V'_{22} + k_2^2 U'_{22}\right)$$

$$\text{EPH} = -\left(\frac{1}{\rho}\frac{\partial}{\partial\rho} k_2^2 V'_{22} + k_2^2 U'_{22}\right)$$

These are multiplied by $C_1\ R_1\ \exp(jkR_1)$ to form the quantities in equation (156) through (159) in Part I. When R_1 is zero the limiting forms in equations (169) through (172) of Part I are used. The expressions from

SN116 to SN118 are obtained by letting θ go to zero in the expressions for $R_1 = 0$.

The data are stored in COMMON/GGRID/ which is identical to the common block in NEC. File 21 is written at SN127 and includes coordinates of the grid boundaries, number of points, and increments for R_1 and θ. Hence those grid parameters can be changed in SOMNEC without changing NEC. If the number of grid points is increased, however, the arrays in COMMON/GGRID/ must be increased in both SOMNEC and NEC. Also, the parameters NDA and NDPA in subroutine INTRP must be changed.

SYMBOL DICTIONARY

AR1 = array for grid 1

AR2 = array for grid 2

AR3 = array for grid 3

CK1 = k_1

CK1R = real part of k_1

CK1SQ = k_1^2

CK2 = k_2 (= 2π since $\lambda = 1$)

CK2SQ = k_2^2

CKSM = $k_2^2/(k_1^2 + k_2^2)$

CL1 = $k_2^2 C_1 C_3$ (see Part 1 for C_1, C_2, and C_3)

CL2 = $k_2^2 C_1 C_2$

CON = $C_1 R_1 \exp(jkR_1)$

CT1 = $(k_1^2 - k_2^2)/2$

CT2 = $(k_1^4 - k_2^4)/8$

CT3 = $(k_1^6 - k_2^6)/16$

DR = ΔR_1

DTH = $\Delta\theta$

DXA = ΔR_1 for each grid

DYA = $\Delta\theta$ for each grid (radians)

EPH = EPH

EPR = ϵ_1

EPSCF = ϵ_c

ERH = ERH

ERV = ERV

EZV = EZV

FMHZ = frequency in MHz

IPT = flag to control printing of grid

IR = index for R_1 values

IRS = starting value for IR

ITH = index for θ values

LCOMP = labels for output

NR = number of R_1 values

NTH = number of θ values

NXA = number of R_1 values for each grid

NYA = number of θ values for each grid

R = R_1

RHO = ρ

RK = k_2R

SIG = σ_1

TFAC1 = $(1 - \sin\theta)/\cos\theta$

TFAC2 = $(1 - \sin\theta)/\cos^2\theta$

THET = θ

TIM = time to fill arrays

TKMAG = $100.|k_1|$

TSMAG = $100.|k_1|^2$

TST = starting time

WLAM = wavelength in free space

XSA = starting value of R_1 in each grid

YSA = starting value of θ in each grid

ZPH = $Z + Z'$

CONSTANTS

299.8 = 10^{-6} times velocity of light in m/s

59.96 = $1/(2\pi c\epsilon_o)$, c = velocity of light

6.283185308 = 2π

```
      PROGRAM SOMNEC(INPUT,OUTPUT,TAPE21)                                SN   1
C                                                                        SN   2
C     PROGRAM TO GENERATE NEC INTERPOLATION GRIDS FOR FIELDS DUE TO      SN   3
C     GROUND.  FIELD COMPONENTS ARE COMPUTED BY NUMERICAL EVALUATION     SN   4
C     OF MODIFIED SOMMERFELD INTEGRALS.                                  SN   5
C                                                                        SN   6
      COMPLEX CK1,CK1SQ,ERV,EZV,ERH,EPH,AR1,AR2,AR3,EPSCF,CKSM,CT1,CT2,C SN   7
     1T3,CL1,CL2,CON                                                     SN   8
      COMMON /EVLCOM/ CKSM,CT1,CT2,CT3,CK1,CK1SQ,CK2,CK2SQ,TKMAG,TSMAG,C SN   9
     1K1R,ZPH,RHO,JH                                                     SN  10
      COMMON /GGRID/ AR1(11,10,4),AR2(17,5,4),AR3(9,8,4),EPSCF,DXA(3),DY SN  11
     1A(3),XSA(3),YSA(3),NXA(3),NYA(3)                                   SN  12
      DIMENSION LCOMP(4)                                                 SN  13
      DATA NXA/11,17,9/,NYA/10,5,8/,XSA/0.,.2,.2/,YSA/0.,0.,.3490658504/ SN  14
      DATA DXA/.02,.05,.1/,DYA/.1745329252,.0872664626,.1745329252/      SN  15
      DATA LCOMP/3HERV,3HEZV,3HERH,3HEPH/                                SN  16
C                                                                        SN  17
C     READ GROUND PARAMETERS - EPR = RELATIVE DIELECTRIC CONSTANT        SN  18
C                              SIG = CONDUCTIVITY (MHOS/M)               SN  19
C                              FMHZ = FREQUENCY (MHZ)                    SN  20
C                              IPT = 1 TO PRINT GRIDS.  =0 OTHERWISE.    SN  21
C     IF SIG .LT. 0. THEN COMPLEX DIELECTRIC CONSTANT = EPR + J*SIG      SN  22
C     AND FMHZ IS NOT USED                                               SN  23
C                                                                        SN  24
      READ 15, EPR,SIG,FMHZ,IPT                                          SN  25
      IF (SIG.LT.0.) GO TO 1                                             SN  26
      WLAM=299.8/FMHZ                                                    SN  27
      EPSCF=CMPLX(EPR,-SIG*WLAM*59.96)                                   SN  28
      GO TO 2                                                            SN  29
    1 EPSCF=CMPLX(EPR,SIG)                                               SN  30
    2 CALL SECOND (TST)                                                  SN  31
      CK2=6.283185308                                                    SN  32
      CK2SQ=CK2*CK2                                                      SN  33
C                                                                        SN  34
C     SOMMERFELD INTEGRAL EVALUATION USES EXP(-JWT), NEC USES EXP(+JWT), SN  35
C     HENCE NEED CONJG(EPSCF).  CONJUGATE OF FIELDS OCCURS IN SUBROUTINE SN  36
C     EVLUA.                                                             SN  37
C                                                                        SN  38
      CK1SQ=CK2SQ*CONJG(EPSCF)                                           SN  39
      CK1=CSQRT(CK1SQ)                                                   SN  40
      CK1R=REAL(CK1)                                                     SN  41
      TKMAG=100.*CABS(CK1)                                               SN  42
      TSMAG=100.*CK1*CONJG(CK1)                                          SN  43
      CKSM=CK2SQ/(CK1SQ+CK2SQ)                                           SN  44
      CT1=.5*(CK1SQ-CK2SQ)                                               SN  45
      ERV=CK1SQ*CK1SQ                                                    SN  46
      EZV=CK2SQ*CK2SQ                                                    SN  47
      CT2=.125*(ERV-EZV)                                                 SN  48
      ERV=ERV*CK1SQ                                                      SN  49
      EZV=EZV*CK2SQ                                                      SN  50
      CT3=.0625*(ERV-EZV)                                                SN  51
C                                                                        SN  52
C     LOOP OVER 3 GRID REGIONS                                           SN  53
C                                                                        SN  54
      DO 6 K=1,3                                                         SN  55
      NR=NXA(K)                                                          SN  56
      NTH=NYA(K)                                                         SN  57
      DR=DXA(K)                                                          SN  58
      DTH=DYA(K)                                                         SN  59
      R=XSA(K)-DR                                                        SN  60
      IRS=1                                                              SN  61
      IF (K.EQ.1) R=XSA(K)                                               SN  62
      IF (K.EQ.1) IRS=2                                                  SN  63
```

```
C
C     LOOP OVER R.  (R=SQRT(RHO**2 + (Z+H)**2))
C
      DO 6 IR=IRS,NR
      R=R+DR
      THET=YSA(K)-DTH
C
C     LOOP OVER THETA.  (THETA=ATAN((Z+H)/RHO))
C
      DO 6 ITH=1,NTH
      THET=THET+DTH
      RHO=R*COS(THET)
      ZPH=R*SIN(THET)
      IF (RHO.LT.1.E-7) RHO=1.E-8
      IF (ZPH.LT.1.E-7) ZPH=0.
      CALL EVLUA (ERV,EZV,ERH,EPH)
      RK=CK2*R
      CON=-(0.,4.77147)*R/CMPLX(COS(RK),-SIN(RK))
      GO TO (3,4,5), K
3     AR1(IR,ITH,1)=ERV*CON
      AR1(IR,ITH,2)=EZV*CON
      AR1(IR,ITH,3)=ERH*CON
      AR1(IR,ITH,4)=EPH*CON
      GO TO 6
4     AR2(IR,ITH,1)=ERV*CON
      AR2(IR,ITH,2)=EZV*CON
      AR2(IR,ITH,3)=ERH*CON
      AR2(IR,ITH,4)=EPH*CON
      GO TO 6
5     AR3(IR,ITH,1)=ERV*CON
      AR3(IR,ITH,2)=EZV*CON
      AR3(IR,ITH,3)=ERH*CON
      AR3(IR,ITH,4)=EPH*CON
6     CONTINUE
C
C     FILL GRID 1 FOR R EQUAL TO ZERO.
C
      CL2=-(0.,188.370)*(EPSCF-1.)/(EPSCF+1.)
      CL1=CL2/(EPSCF+1.)
      EZV=EPSCF*CL1
      THET=-DTH
      NTH=NYA(1)
      DO 9 ITH=1,NTH
      THET=THET+DTH
      IF (ITH.EQ.NTH) GO TO 7
      TFAC2=COS(THET)
      TFAC1=(1.-SIN(THET))/TFAC2
      TFAC2=TFAC1/TFAC2
      ERV=EPSCF*CL1*TFAC1
      ERH=CL1*(TFAC2-1.)+CL2
      EPH=CL1*TFAC2-CL2
      GO TO 8
7     ERV=0.
      ERH=CL2-.5*CL1
      EPH=-ERH
8     AR1(1,ITH,1)=ERV
      AR1(1,ITH,2)=EZV
      AR1(1,ITH,3)=ERH
9     AR1(1,ITH,4)=EPH
      CALL SECOND (TIM)
C
C     WRITE GRID ON TAPE21
C
      WRITE (21) AR1,AR2,AR3,EPSCF,DXA,DYA,XSA,YSA,NXA,NYA
```

```
      REWIND 21                                                         SN 128
      IF (IPT.EQ.0) GO TO 14                                            SN 129
C                                                                       SN 130
C     PRINT GRID                                                        SN 131
C                                                                       SN 132
      PRINT 17, EPSCF                                                   SN 133
      DO 13 K=1,3                                                       SN 134
      NR=NXA(K)                                                         SN 135
      NTH=NYA(K)                                                        SN 136
      PRINT 18, K,XSA(K),DXA(K),NR,YSA(K),DYA(K),NTH                    SN 137
      DO 13 L=1,4                                                       SN 138
      PRINT 19, LCOMP(L)                                                SN 139
      DO 13 IR=1,NR                                                     SN 140
      GO TO (10,11,12), K                                               SN 141
10    PRINT 20, IR,(AR1(IR,ITH,L),ITH=1,NTH)                            SN 142
      GO TO 13                                                          SN 143
11    PRINT 20, IR,(AR2(IR,ITH,L),ITH=1,NTH)                            SN 144
      GO TO 13                                                          SN 145
12    PRINT 20, IR,(AR3(IR,ITH,L),ITH=1,NTH)                            SN 146
13    CONTINUE                                                          SN 147
14    TIM=TIM-TST                                                       SN 148
      PRINT 16,TIM                                                      SN 149
      STOP                                                              SN 150
C                                                                       SN 151
15    FORMAT (3E10.3,I5)                                                SN 152
16    FORMAT (6H TIME=,E12.5)                                           SN 153
17    FORMAT (30H1NEC GROUND INTERPOLATION GRID,/,21H DIELECTRIC CONSTAN SN 154
     1T=,2E12.5)                                                        SN 155
18    FORMAT (///,5H GRID,I2,/,4X,5HR(1)=,F7.4,4X,3HDR=,F7.4,4X,3HNR=,I3 SN 156
     1,/,9H THET(1)=,F7.4,3X,4HDTH=,F7.4,3X,4HNTH=,I3,//)               SN 157
19    FORMAT (///,A3)                                                   SN 158
20    FORMAT (4H IR=,I3,/,(10E12.5))                                    SN 159
      END                                                               SN 160-
```

PURPOSE

To compute the Bessel function of order zero and its derivative for a complex argument.

METHOD

For argument magnitudes less than a limit Z_s the functions are evaluated by the ascending series and for larger magnitudes by Hankel's asymptotic expansion (ref. 5). The ascending series are

$$J_0(Z) = \sum_{k=0}^{\infty} \frac{(-Z^2/4)^k}{(k!)^2}$$

$$J_0'(Z) = -J_1(Z) = -\frac{Z}{2} \sum_{k=0}^{\infty} \frac{(-Z^2/4)^k}{k!(k+1)!}$$

The number of terms used with an argument Z is M(IZ) where IZ = 1. + $|Z|^2$. The array M is filled for IZ from 1 to 101 on the first call to BESSEL by determining the value of k at which the term in the series for J_0 is less than 10^{-6}.

When $|Z|$ is greater than Z_s Hankel's asymptotic expansions are used with two or three terms. These are

$$J_\nu(Z) = \sqrt{\frac{2}{\pi Z}}\,[P(\nu,Z)\cos\chi - Q(\nu,Z)\sin\chi] \qquad (|\arg Z| < \pi)$$

$$\chi = Z - (\tfrac{1}{2}\nu + \tfrac{1}{4})\pi$$

$$P(\nu,Z) = 1 - \frac{(\mu-1)(\mu-9)}{2!(8Z)^2} + \frac{(\mu-1)(\mu-9)(\mu-25)(\mu-49)}{4!(8Z)^4}$$

$$Q(\nu,Z) = \frac{(\mu-1)}{8} - \frac{(\mu-1)(\mu-9)(\mu-25)}{3!(8Z)^3}$$

where $\mu = 4\nu^2$.

When $Z_s < |Z| < Z_s + \Delta$ both the series and asymptotic forms are evaluated and are combined as

$$J(Z) = \frac{1}{2}\left[J_s(Z)(1+C) + J_a(Z)(1-C)\right]$$

where $C = \cos\left(\frac{\pi}{\Delta}(|Z| - Z_s)\right)$

$J_s(Z)$ = result of series evaluation

$J_a(Z)$ = result of asymptotic evaluation

This combination ensures a smooth transition between the two regions. In the code Z_s is 6 and Δ is 0.1.

SYMBOL DICTIONARY

A1	=	$-1./(4k^2)$
A2	=	$1./(k + 1)$
C3	=	$\sqrt{2/\pi} = 0.7978845608$
CZ	=	$\cos \chi$
FJ	=	$\sqrt{-1}$.
FJX	=	FJ
IB	=	1 to indicate that both the series and asymptotic forms will be evaluated and combined
INIT	=	flag to indicate that initialization of constants has been completed
IZ	=	$1. + \|Z\|^2$ truncated to an integer
J0	=	$J_0(Z)$
J0P	=	$J_0'(Z)$
J0PX	=	$J_0'(Z)$ from series to be combined with asymptotic result
J0X	=	$J_0(Z)$, same as J0PX
K	=	summation index k, summed from 1 to limit

M	= array of upper limits for k
MIZ	= upper limit for k
POZ	= P(0,z)
P10	= coefficient in POZ = $9/(2 \times 8^2)$
P11	= coefficient in P1Z = $-(4 - 1)(4 - 9)/(2 \times 8^2)$
P1Z	= P(1,Z)
P20	= coefficient in POZ = $9 \times 25 \times 49/(4!8^4)$
P21	= coefficient in P1Z = $-(4 - 1)(4 - 9)(4 - 25)(4 - 49)/(4!8^4)$
PI	= π
POF	= $\pi/4$
QOZ	= Q(0,Z)
Q10	= coefficient in Q(0,Z) = 1/8
Q11	= coefficient in Q(1,Z) = 3/8
Q1Z	= Q(1,Z)
Q20	= coefficient in Q(0,Z) = $9 \times 25/(3!8^3)$
Q21	= coefficient in Q(1,Z) = $(4 - 1)(4 - 9)(4 - 25)/(3!8^3)$
SZ	= sin χ
TEST	= magnitude of the term in the series
Z	= Z
ZI	= Z^2 or $1/Z$
ZI2	= $1/Z^2$ or $\exp(-j\chi)$
ZK	= $(-Z^2/4)^k/(k!)^2$ for series. Also temporary storage for asymptotic method
ZMS	= $\lvert Z\rvert^2$ or temporary storage

CONSTANTS

31.41592654	= $10.\pi$
36.	= 6^2
37.21	= 6.1^2

```
      SUBROUTINE BESSEL (Z,J0,J0P)                                      BE   1
C                                                                       BE   2
C     BESSEL EVALUATES THE ZERO-ORDER BESSEL FUNCTION AND ITS DERIVATIVE BE   3
C     FOR COMPLEX ARGUMENT Z.                                           BE   4
C                                                                       BE   5
      COMPLEX J0,J0P,P0Z,P1Z,Q0Z,Q1Z,Z,ZI,ZI2,ZK,FJ,CZ,SZ,J0X,J0PX      BE   6
      DIMENSION M(101), A1(25), A2(25), FJX(2)                          BE   7
      EQUIVALENCE (FJ,FJX)                                              BE   8
      DATA PI,C3,P10,P20,Q10,Q20/3.141592654,.7978845608,.0703125,.11215 BE   9
     120996,.125,.0732421875/                                           BE  10
      DATA P11,P21,Q11,Q21/.1171875,.1441955566,.375,.1025390625/       BE  11
      DATA POF,INIT/.7853981635,0/,FJX/0.,1./                           BE  12
      IF (INIT.EQ.0) GO TO 5                                            BE  13
    1 ZMS=Z*CONJG(Z)                                                    BE  14
      IF (ZMS.GT.1.E-12) GO TO 2                                        BE  15
      J0=(1.,0.)                                                        BE  16
      J0P=-.5*Z                                                         BE  17
      RETURN                                                            BE  18
    2 IB=0                                                              BE  19
      IF (ZMS.GT.37.21) GO TO 4                                         BE  20
      IF (ZMS.GT.36.) IB=1                                              BE  21
C     SERIES EXPANSION                                                  BE  22
      IZ=1.+ZMS                                                         BE  23
      MIZ=M(IZ)                                                         BE  24
      J0=(1.,0.)                                                        BE  25
      J0P=J0                                                            BE  26
      ZK=J0                                                             BE  27
      ZI=Z*Z                                                            BE  28
      DO 3 K=1,MIZ                                                      BE  29
      ZK=ZK*A1(K)*ZI                                                    BE  30
      J0=J0+ZK                                                          BE  31
    3 J0P=J0P+A2(K)*ZK                                                  BE  32
      J0P=-.5*Z*J0P                                                     BE  33
      IF (IB.EQ.0) RETURN                                               BE  34
      J0X=J0                                                            BE  35
      J0PX=J0P                                                          BE  36
C     ASYMPTOTIC EXPANSION                                              BE  37
    4 ZI=1./Z                                                           BE  38
      ZI2=ZI*ZI                                                         BE  39
      P0Z=1.+(P20*ZI2-P10)*ZI2                                          BE  40
      P1Z=1.+(P11-P21*ZI2)*ZI2                                          BE  41
      Q0Z=(Q20*ZI2-Q10)*ZI                                              BE  42
      Q1Z=(Q11-Q21*ZI2)*ZI                                              BE  43
      ZK=CEXP(FJ*(Z-POF))                                               BE  44
      ZI2=1./ZK                                                         BE  45
      CZ=.5*(ZK+ZI2)                                                    BE  46
      SZ=FJ*.5*(ZI2-ZK)                                                 BE  47
      ZK=C3*CSQRT(ZI)                                                   BE  48
      J0=ZK*(P0Z*CZ-Q0Z*SZ)                                             BE  49
      J0P=-ZK*(P1Z*SZ+Q1Z*CZ)                                           BE  50
      IF (IB.EQ.0) RETURN                                               BE  51
      ZMS=COS((SQRT(ZMS)-6.)*31.41592654)                               BE  52
      J0=.5*(J0X*(1.+ZMS)+J0*(1.-ZMS))                                  BE  53
      J0P=.5*(J0PX*(1.+ZMS)+J0P*(1.-ZMS))                               BE  54
      RETURN                                                            BE  55
C     INITIALIZATION OF CONSTANTS                                       BE  56
    5 DO 6 K=1,25                                                       BE  57
      A1(K)=-.25/(K*K)                                                  BE  58
    6 A2(K)=1./(K+1.)                                                   BE  59
      DO 8 I=1,101                                                      BE  60
      TEST=1.                                                           BE  61
      DO 7 K=1,24                                                       BE  62
      INIT=K                                                            BE  63
      TEST=-TEST*I*A1(K)                                                BE  64
```

```
      IF (TEST.LT.1.E-6) GO TO 8                                        BE  65
    7 CONTINUE                                                          BE  66
    8 M(I)=INIT                                                         BE  67
      GO TO 1                                                           BE  68
      END                                                               BE  69-
```

EVLUA

PURPOSE

To control the evaluation of the Sommerfeld integrals.

METHOD

The integration contour of either Figures 13, 14 or 15 of Part I is used depending on the values of ρ, $Z + Z'$ and k_1. Figures 13, 14, and 15 should be inverted, however, since they are for a time dependence of $\exp(j\omega t)$ and the coding for the Sommerfeld integrals is for $\exp(-j\omega t)$. Thus the contours and branch cuts in EVALUA are the conjugate of those shown. The conjugate of the results is taken at the end of EVLUA to conform to the NEC time dependence of $\exp(j\omega t)$.

The code from EV 19 to EV 34 evaluates the Bessel function form of the Sommerfeld integrals using the contour of Figure 13 of Part I. ROM1 is called to integrate from $\lambda = 0$ to $(p - jp)$ and GSHANK is called for the path from $(p - jp)$ to infinity where p^{-1} is the maximum of ρ and $Z + Z'$ (p = DEL). If p is greater than $100.|k_1|$ then ROM1 is called for the interval 0 to $(p_1 - jp_1)$ where $p_1 = 10|k_1|$. This is done to avoid exceeding the limit by which ROM1 can cut the interval width. Larger steps can then be used from $(p_1 - jp_1)$ to $(p - jp)$ since $\gamma_1 \approx \gamma_2 \approx \lambda$.

The code from EV 39 to EV 86 evaluates the Hankel function form of the integrals using either the contour of Figure 14 or 15. At EV 50 SUM is the negative of the integral from a* to c*. GSHANK is then called to integrate from a* to $-\infty$. The decision whether to use the contour of Figures 14 or 15 is made from EV 58 to EV 64. Figure 15 is used if the real part of $\rho(k_1 - k_2)$ exceeds $2k_2$ and

$$\frac{-u}{|v|} > \frac{4\rho}{Z+Z'}$$

where $u + jv = [-(Z + Z') + j\rho][d^* - c^*]$ is the argument of the exponential function approximating the Sommerfeld integrand for large λ with $\lambda = d^* - c^*$. The left side of the inequality is proportional to the decay per cycle along the c to d path and $\rho/(Z + Z')$ is the same for the vertical path. This condition was chosen arbitrarily but gives some indication of when the contour of Figure 16 may be advantageous.

For the contour of Figure 15 GSHANK is called to integrate from e* to infinity. ROM1 is then called from e* to f*. The sign of the contribution from other parts of the path is switched since they were integrated in reverse direction. Finally, GSHANK is called for the paths from c* to infinity and f* to infinity.

For the contour of Figure 14 (GS 79 to GS 86) GSHANK is called to integrate from c* to d* and on to infinity. The increment changes from DELTA to DELTA2 if d* is reached before the integral converges.

From EV 89 to EV 92 the integrals are combined to form the field components and the conjugates are taken.

SYMBOL DICTIONARY

A = start of integration interval
ANS = temporary storage
B = end of integration interval
BK = break point (d*) in path for GSHANK
CK1 = k_1
CK1SQ = k_1^2
CK2 = k_2

CK2SQ = k_2^2
CP1 = a*
CP2 = b*
CP3 = c*
DEL = p
DELTA = increment along path
DELTA2 = alternate increment
EPH = (see SOMNEC)
ERH = (see SOMNEC)
ERV = (see SOMNEC)
EZV = (see SOMNEC)
JH = 0 for Bessel function form, 1 for Hankel function form
PTP = 0.2π
RHØ = ρ

RMIS = temporary storage

SLOPE = slope of paths to infinity

SUM = temporary storage

TKMAG = $100|k_1|$

ZPH = $Z + Z'$

```
      SUBROUTINE EVLUA (ERV,EZV,ERH,EPH)                                EV   1
C                                                                       EV   2
C     EVALUA CONTROLS THE INTEGRATION CONTOUR IN THE COMPLEX LAMBDA     EV   3
C     PLANE FOR EVALUATION OF THE SOMMERFELD INTEGRALS.                 EV   4
C                                                                       EV   5
      COMPLEX ERV,EZV,ERH,EPH,A,B,CK1,CK1SQ,BK,SUM,DELTA,ANS,DELTA2,CP1, EV   6
     1CP2,CP3,CKSM,CT1,CT2,CT3                                          EV   7
      COMMON /CNTOUR/ A,B                                               EV   8
      COMMON /EVLCOM/ CKSM,CT1,CT2,CT3,CK1,CK1SQ,CK2,CK2SQ,TKMAG,TSMAG,C EV   9
     1K1R,ZPH,RHO,JH                                                    EV  10
      DIMENSION SUM(6), ANS(6)                                          EV  11
      DATA PTP/.6283185308/                                             EV  12
      DEL=ZPH                                                           EV  13
      IF (RHO.GT.DEL) DEL=RHO                                           EV  14
      IF (ZPH.LT.2.*RHO) GO TO 4                                        EV  15
C                                                                       EV  16
C     BESSEL FUNCTION FORM OF SOMMERFELD INTEGRALS                      EV  17
C                                                                       EV  18
      JH=0                                                              EV  19
      A=(0.,0.)                                                         EV  20
      DEL=1./DEL                                                        EV  21
      IF (DEL.LE.TKMAG) GO TO 2                                         EV  22
      B=CMPLX(.1*TKMAG,-.1*TKMAG)                                       EV  23
      CALL ROM1 (6,SUM,2)                                               EV  24
      A=B                                                               EV  25
      B=CMPLX(DEL,-DEL)                                                 EV  26
      CALL ROM1 (6,ANS,2)                                               EV  27
      DO 1 I=1,6                                                        EV  28
1     SUM(I)=SUM(I)+ANS(I)                                              EV  29
      GO TO 3                                                           EV  30
2     B=CMPLX(DEL,-DEL)                                                 EV  31
      CALL ROM1 (6,SUM,2)                                               EV  32
3     DELTA=PTP*DEL                                                     EV  33
      CALL GSHANK (B,DELTA,ANS,6,SUM,0,B,B)                             EV  34
      GO TO 10                                                          EV  35
C                                                                       EV  36
C     HANKEL FUNCTION FORM OF SOMMERFELD INTEGRALS                      EV  37
C                                                                       EV  38
4     JH=1                                                              EV  39
      CP1=CMPLX(0.,.4*CK2)                                              EV  40
      CP2=CMPLX(.6*CK2,-.2*CK2)                                         EV  41
      CP3=CMPLX(1.02*CK2,-.2*CK2)                                       EV  42
      A=CP1                                                             EV  43
      B=CP2                                                             EV  44
      CALL ROM1 (6,SUM,2)                                               EV  45
      A=CP2                                                             EV  46
      B=CP3                                                             EV  47
      CALL ROM1 (6,ANS,2)                                               EV  48
      DO 5 I=1,6                                                        EV  49
5     SUM(I)=-(SUM(I)+ANS(I))                                           EV  50
C     PATH FROM IMAGINARY AXIS TO -INFINITY                             EV  51
      SLOPE=1000.                                                       EV  52
      IF (ZPH.GT..001*RHO) SLOPE=RHO/ZPH                                EV  53
      DEL=PTP/DEL                                                       EV  54
      DELTA=CMPLX(-1.,SLOPE)*DEL/SQRT(1.+SLOPE*SLOPE)                   EV  55
      DELTA2=-CONJG(DELTA)                                              EV  56
      CALL GSHANK (CP1,DELTA,ANS,6,SUM,0,BK,BK)                         EV  57
      RMIS=RHO*(REAL(CK1)-CK2)                                          EV  58
      IF (RMIS.LT.2.*CK2) GO TO 8                                       EV  59
      IF (RHO.LT.1.E-10) GO TO 8                                        EV  60
      IF (ZPH.LT.1.E-10) GO TO 6                                        EV  61
      BK=CMPLX(-ZPH,RHO)*(CK1-CP3)                                      EV  62
      RMIS=-REAL(BK)/ABS(AIMAG(BK))                                     EV  63
      IF(RMIS.GT.4.*RHO/ZPH)GO TO 8                                     EV  64
```

```
C     INTEGRATE UP BETWEEN BRANCH CUTS, THEN TO + INFINITY
6     CP1=CK1-(.1,.2)
      CP2=CP1+.2
      BK=CMPLX(0.,DEL)
      CALL GSHANK (CP1,BK,SUM,6,ANS,0,BK,BK)
      A=CP1
      B=CP2
      CALL ROM1 (6,ANS,1)
      DO 7 I=1,6
7     ANS(I)=ANS(I)-SUM(I)
      CALL GSHANK (CP3,BK,SUM,6,ANS,0,BK,BK)
      CALL GSHANK (CP2,DELTA2,ANS,6,SUM,0,BK,BK)
      GO TO 10
C     INTEGRATE BELOW BRANCH POINTS, THEN TO + INFINITY
8     DO 9 I=1,6
9     SUM(I)=-ANS(I)
      RMIS=REAL(CK1)*1.01
      IF (CK2+1..GT.RMIS) RMIS=CK2+1.
      BK=CMPLX(RMIS,.99*AIMAG(CK1))
      DELTA=BK-CP3
      DELTA=DELTA*DEL/CABS(DELTA)
      CALL GSHANK (CP3,DELTA,ANS,6,SUM,1,BK,DELTA2)
10    ANS(6)=ANS(6)*CK1
C     CONJUGATE SINCE NEC USES EXP(+JWT)
      ERV=CONJG(CK1SQ*ANS(3))
      EZV=CONJG(CK1SQ*(ANS(2)+CK2SQ*ANS(5)))
      ERH=CONJG(CK2SQ*(ANS(1)+ANS(6)))
      EPH=-CONJG(CK2SQ*(ANS(4)+ANS(6)))
      RETURN
      END
```

GSHANK

PURPOSE

To apply the Shanks transformation (ref. 6) to accelerate the convergence of a semi-infinite integral.

METHOD

Six integrals (NANS = 6) are evaluated simultaneously in this routine. The integrals over semi-infinite sections of the contours (Figures 13, 14 and 15 of Part I) are evaluated by using the Romberg variable interval width integration method on subsections to obtain a converging sequence of partial sums

$$S_i = S_0 + \int_{A_0}^{A_0+i\Delta} f(\lambda)d\lambda \qquad i = 1, 2, \ldots$$

where A_0 is the start of the semi-infinite path, S_0 is the contribution from other parts of the contour and Δ is a complex increment with

$$|\Delta| = \text{minimum of} \begin{cases} 0.2\pi/\rho \\ 0.2\pi/(Z + Z') \end{cases}$$

$$\arg(\Delta) = \text{direction of integration path in } \lambda\text{-plane}$$

The Shanks interated first order transformation is applied to S_i to accelerate convergence. Starting with the sequence of M elements $Q_{i,0} = S_i$, $i = 1, \ldots M$ the j^{th} iterated transform is the sequence of $M - 2j$ elements

$$Q_{ij} = \frac{Q_{i-1,j-1}Q_{i+1,j-1} - Q^2_{i,\,j-1}}{Q_{i-1,j-1} + Q_{i+1,j-1} - 2Q_{i,\,j-1}}$$

$$= Q_{i-1,j-1} - \frac{(Q_{i,j-1} - Q_{i-1,j-1})^2}{Q_{i-1,j-1} + Q_{i+1,j-1} - 2Q_{i,j-1}}$$

$i = j + 1, \ldots M - j$

$j = 1, \ldots [(M - 1)/2]$.

The second form for $Q_{i,j}$ is used since it suffers less numerical error as the sequence converges. Each iteration of the transform should produce a sequence that converges more rapidly to the limit of the original sequence.

In this subroutine the starting value S_0 comes in as SEED. With each pass through the loop over INT, starting at GS 21, two new values are added to the sequence by calling ROM1 to evaluate the integrals

$$S_{2N-1} = S_{2N-2} + \int_{A_0+(2N-2)\Delta}^{A_0+(2N-1)\Delta} f(\lambda)d\lambda$$

$$S_{2N} = S_{2N-1} + \int_{A_0+(2N-1)\Delta}^{A_0+(2N)\Delta} f(\lambda)d\lambda$$

where N = INT. The $(N - 1)^{th}$ interated Shanks transformation, consisting of the two elements $Q_{N,N-1}$ and $Q_{N+1,N-1}$, is then computed. At the end of each pass through the loop over INT the arrays Q1 and Q2 contain the last two elements in each sequence. For function I,

$$Q1(I,J) = Q_{2N-J,J-1}$$

$$Q2(I,J) = Q_{2N-J+1,J-1}, \quad J = 1, \ldots N.$$

For the path from c to infinity in Figure 14 of Part I the point d is a break point at which Δ may change. If d is reached before convergence the Shanks transformation is started over with the final value of S_i becoming S_0 for the new sequence.

Convergence is tested from GS 78 to GS 89 by comparing the last two values in the transformed sequences. Although the last sequence, consisting of two elements, should have the highest convergence the last four sequences are tested to avoid a false indication of convergence. The relative difference is computed for each of the six functions and compared with CRIT. If convergence does not occur by INT = MAXH a message is printed and the average of the two values in the last sequence is used for each integral. In computing the relative difference for each function the denominator is not allowed to be less than 10^{-3} times the magnitude of the largest of the six functions to avoid convergence problems when one function goes to zero.

SYMBOL DICTIONARY

A	= beginning of integration subinterval
A1	= new value for Q1 array
A2	= new value for Q2 array
AA	= temporary storage
AMG	= approximate magnitude of function
ANS1	= S_i for i odd
ANS2	= S_i for i even
AS1	= S_i for i odd
AS2	= S_i for i even
B	= end of integration subinterval
BK	= break point in integration contour
CRIT	= limit for relative error in convergence test
DEL	= Δ
DELA	= Δ before break point
DELB	= Δ after break point
DEN	= approximate magnitude of the largest of the six functions (GS 76)
DENM	= minimum denominator for relative error test
IBK	= 1 if path contains break point
IBX	= 0 if path contains break point and it has not been passed
INT	= N

INX = INT

JM = J - 1

MAXH = maximum for index J in Q1 and Q2

NANS = number of functions (6)

Q1, Q2 = (see description of method)

RBK = real part of BK

SEED = S_0

START = A_0

SUM = increment to integral

```
      SUBROUTINE GSHANK (START,DELA,SUM,NANS,SEED,IBK,BK,DELB)
C
C     GSHANK INTEGRATES THE 6 SOMMERFELD INTEGRALS FROM START TO
C     INFINITY (UNTIL CONVERGENCE) IN LAMBDA.  AT THE BREAK POINT, BK,
C     THE STEP INCREMENT MAY BE CHANGED FROM DELA TO DELB.  SHANK S
C     ALGORITHM TO ACCELERATE CONVERGENCE OF A SLOWLY CONVERGING SERIES
C     IS USED
C
      COMPLEX START,DELA,SUM,SEED,BK,DELB,A,B,Q1,Q2,ANS1,ANS2,A1,A2,AS1,
     1AS2,DEL,AA
      COMMON /CNTOUR/ A,B
      DIMENSION Q1(6,20), Q2(6,20), ANS1(6), ANS2(6), SUM(6), SEED(6)
      DATA CRIT/1.E-4/,MAXH/20/
      RBK=REAL(BK)
      DEL=DELA
      IBX=0
      IF (IBK.EQ.0) IBX=1
      DO 1 I=1,NANS
1     ANS2(I)=SEED(I)
      B=START
2     DO 20 INT=1,MAXH
      INX=INT
      A=B
      B=B+DEL
      IF (IBX.EQ.0.AND.REAL(B).GE.RBK) GO TO 5
      CALL ROM1 (NANS,SUM,2)
      DO 3 I=1,NANS
3     ANS1(I)=ANS2(I)+SUM(I)
      A=B
      B=B+DEL
      IF (IBX.EQ.0.AND.REAL(B).GE.RBK) GO TO 6
      CALL ROM1 (NANS,SUM,2)
      DO 4 I=1,NANS
4     ANS2(I)=ANS1(I)+SUM(I)
      GO TO 11
C     HIT BREAK POINT.  RESET SEED AND START OVER.
5     IBX=1
      GO TO 7
6     IBX=2
7     B=BK
      DEL=DELB
      CALL ROM1 (NANS,SUM,2)
      IF (IBX.EQ.2) GO TO 9
      DO 8 I=1,NANS
8     ANS2(I)=ANS2(I)+SUM(I)
      GO TO 2
9     DO 10 I=1,NANS
10    ANS2(I)=ANS1(I)+SUM(I)
      GO TO 2
11    DEN=0.
      DO 18 I=1,NANS
      AS1=ANS1(I)
      AS2=ANS2(I)
      IF (INT.LT.2) GO TO 17
      DO 16 J=2,INT
      JM=J-1
      AA=Q2(I,JM)
      A1=Q1(I,JM)+AS1-2.*AA
      IF (REAL(A1).EQ.0..AND.AIMAG(A1).EQ.0.) GO TO 12
      A2=AA-Q1(I,JM)
      A1=Q1(I,JM)-A2*A2/A1
      GO TO 13
12    A1=Q1(I,JM)
13    A2=AA+AS2-2.*AS1
```

```
      IF (REAL(A2).EQ.0..AND.AIMAG(A2).EQ.0.) GO TO 14
      A2=AA-(AS1-AA)*(AS1-AA)/A2
      GO TO 15
   14 A2=AA
   15 Q1(I,JM)=AS1
      Q2(I,JM)=AS2
      AS1=A1
   16 AS2=A2
   17 Q1(I,INT)=AS1
      Q2(I,INT)=AS2
      AMG=ABS(REAL(AS2))+ABS(AIMAG(AS2))
      IF (AMG.GT.DEN) DEN=AMG
   18 CONTINUE
      DENM=1.E-3*DEN*CRIT
      JM=INT-3
      IF (JM.LT.1) JM=1
      DO 19 J=JM,INT
      DO 19 I=1,NANS
      A1=Q2(I,J)
      DEN=(ABS(REAL(A1))+ABS(AIMAG(A1)))*CRIT
      IF (DEN.LT.DENM) DEN=DENM
      A1=Q1(I,J)-A1
      AMG=ABS(REAL(A1))+ABS(AIMAG(A1))
      IF (AMG.GT.DEN) GO TO 20
   19 CONTINUE
      GO TO 22
   20 CONTINUE
      PRINT 24
      DO 21 I=1,NANS
   21 PRINT 25, Q1(I,INX),Q2(I,INX)
   22 DO 23 I=1,NANS
   23 SUM(I)=.5*(Q1(I,INX)+Q2(I,INX))
      RETURN
C
   24 FORMAT (46H **** NO CONVERGENCE IN SUBROUTINE GSHANK ****)
   25 FORMAT (10E12.5)
      END
```

HANKEL

PURPOSE

To compute the Hankel function of the first kind, zeroth order, and its derivative for a complex argument.

METHOD

For argument magnitudes less than a limit Z_s the functions are evaluated by the ascending series and for larger magnitudes by Hankel's asymptotic expansion (ref. 5). The series are

$$Y_0(Z) = \frac{2}{\pi} \ln(Z/2) J_0(Z) - \frac{2}{\pi} \sum_{k=0}^{\infty} \psi(k+1) \frac{(-Z^2/4)^k}{(k!)^2}$$

$$Y_0'(Z) = \frac{2}{\pi Z} + \frac{2}{\pi} \ln(Z/2) J_0'(Z) + \frac{Z}{2\pi} \sum_{k=0}^{\infty} [\psi(k+1) + \psi(k+2)] \frac{(-Z^2/4)^k}{k!(k+1)!}$$

where $\psi(k+1) = -\gamma + \sum_{j=1}^{k} j^{-1}$

$\psi(1) = -\gamma$,

γ = Euler's constant = 0.5772156649

The Hankel functions are

$$H_0^{(1)}(Z) = J_0(Z) + j\, Y_0(Z)$$

$$H_0^{(1)'}(Z) = J_0'(Z) + j\, Y_0'(Z)$$

The series for $J_0(Z)$ and $J_0'(Z)$ are given in the description of subroutine BESSEL. The number of terms used with an argument Z is M(IZ) where IZ = 1. + $|Z|^2$.

The array M is filled for IZ from 1 to 101 on the first call to HANKEL by determining the value of k at which the term in the series of Y_0 is less than 10^{-6}.

When $|Z|$ is greater than Z_s Hankel's asymptotic expansions are used with two or three terms. These are

$$H_\nu^{(1)}(Z) = \sqrt{\frac{2}{\pi Z}}\,[P(\nu,Z) + jQ(\nu,Z)]e^{jX}$$

$$X = Z - (\frac{1}{2}\nu + \frac{1}{4})\pi$$

$P(\nu,Z)$ and $Q(\nu,Z)$ are given in the description of subroutine BESSEL.

When $Z_s < |Z| < Z_s + \Delta$ both the series and asymptotic forms are evaluated and are combined as in BESSEL to eliminate any discontinity. In HANKEL Z_s is 4 and Δ is 0.1.

SYMBOL DICTIONARY

A1	$= -1./(4k^2)$
A2	$= 1./(k+1)$
A3	$= 2\psi(k+1)$
A4	$= [\psi(k+1) + \psi(k+2)]/(k+1)$
C1	$= [\psi(1) + \psi(2)]/(2\pi)$
C2	$= 2\gamma/\pi$
C3	$= \sqrt{2/\pi}$
CLOGZ	$= \ln(Z)$
FJ	$= \sqrt{-1}$
FJX	= FJ
GAMMA	$= \gamma$
HO	$= H_0^{(1)}(Z)$
HOP	$= H_0^{(1)'}(Z)$
IB	= 1 to indicate that both the series and asymptotic forms will be evaluated and combined
INIT	= flag to indicate that initialization of constants has been completed
IZ	$= 1. + \|Z\|^2$
JO	$= J_0(Z)$
JOP	$= J_0'(Z)$

K = summation index k, summed from 1 to limit

M = array of upper limits for k

MIZ = upper limit for k

POZ, P10, P11, P1Z, P20, P21: see BESSEL

PI = π

POF = $\pi/4$

PSI = ψ

QOZ, Q10, Q11, Q1Z, Q20, Q21: see BESSEL

TEST = magnitude of term in the series

YO = $Y_0(Z)$

YOP = $Y_0'(Z)$

Z = Z

ZI = Z^2 or $1/Z$

ZI2 = $1/Z^2$

ZK = $(-Z^2/4)^k/(k!)^2$; also temporary storage

ZMS = $|Z^2|$ or temporary storage

CONSTANTS

16. = 4^2

16.81 = 4.1^2

31.41592654 = $10.\pi$

```
      SUBROUTINE HANKEL (Z,H0,HOP)
C
C     HANKEL EVALUATES HANKEL FUNCTION OF THE FIRST KIND, ORDER ZERO,
C     AND ITS DERIVATIVE FOR COMPLEX ARGUMENT Z.
C
      COMPLEX CLOGZ,H0,HOP,J0,J0P,P0Z,P1Z,Q0Z,Q1Z,Y0,Y0P,Z,ZI,ZI2,ZK,FJ
      DIMENSION M(101), A1(25), A2(25), A3(25), A4(25), FJX(2)
      EQUIVALENCE (FJ,FJX)
      DATA PI,GAMMA,C1,C2,C3,P10,P20/3.141592654,.5772156649,-.024578509
     15,.3674669052,.7978845608,.0703125,.1121520996/
      DATA Q10,Q20,P11,P21,Q11,Q21/.125,.0732421875,.1171875,.1441955566
     1,.375,.1025390625/
      DATA POF,INIT/.7853981635,0/,FJX/0.,1./
      IF (INIT.EQ.0) GO TO 5
1     ZMS=Z*CONJG(Z)
      IF (ZMS.NE.0.) GO TO 2
      PRINT 9
      STOP
2     IB=0
      IF (ZMS.GT.16.81) GO TO 4
      IF (ZMS.GT.16.) IB=1
C     SERIES EXPANSION
      IZ=1.+ZMS
      MIZ=M(IZ)
      J0=(1.,0.)
      J0P=J0
      Y0=(0.,0.)
      Y0P=Y0
      ZK=J0
      ZI=Z*Z
      DO 3 K=1,MIZ
      ZK=ZK*A1(K)*ZI
      J0=J0+ZK
      J0P=J0P+A2(K)*ZK
      Y0=Y0+A3(K)*ZK
3     Y0P=Y0P+A4(K)*ZK
      J0P=-.5*Z*J0P
      CLOGZ=CLOG(.5*Z)
      Y0=(2.*J0*CLOGZ-Y0)/PI+C2
      Y0P=(2./Z+2.*J0P*CLOGZ+.5*Y0P*Z)/PI+C1*Z
      H0=J0+FJ*Y0
      HOP=J0P+FJ*Y0P
      IF (IB.EQ.0) RETURN
      Y0=H0
      Y0P=HOP
C     ASYMPTOTIC EXPANSION
4     ZI=1./Z
      ZI2=ZI*ZI
      P0Z=1.+(P20*ZI2-P10)*ZI2
      P1Z=1.+(P11-P21*ZI2)*ZI2
      Q0Z=(Q20*ZI2-Q10)*ZI
      Q1Z=(Q11-Q21*ZI2)*ZI
      ZK=CEXP(FJ*(Z-POF))*CSQRT(ZI)*C3
      H0=ZK*(P0Z+FJ*Q0Z)
      HOP=FJ*ZK*(P1Z+FJ*Q1Z)
      IF (IB.EQ.0) RETURN
      ZMS=COS((SQRT(ZMS)-4.)*31.41592654)
      H0=.5*(Y0*(1.+ZMS)+H0*(1.-ZMS))
      HOP=.5*(Y0P*(1.+ZMS)+HOP*(1.-ZMS))
      RETURN
C     INITIALIZATION OF CONSTANTS
5     PSI=-GAMMA
      DO 6 K=1,25
      A1(K)=-.25/(K*K)
```

```
      A2(K)=1./(K+1.)
      PSI=PSI+1./K
      A3(K)=PSI+PSI
    6 A4(K)=(PSI+PSI+1./(K+1.))/(K+1.)
      DO 8 I=1,101
      TEST=1.
      DO 7 K=1,24
      INIT=K
      TEST=-TEST*I*A1(K)
      IF (TEST*A3(K).LT.1.E-6) GO TO 8
    7 CONTINUE
    8 M(I)=INIT
      GO TO 1
C
    9 FORMAT (34H ERROR - HANKEL NOT VALID FOR Z=0.)
      END
```

LAMBDA

PURPOSE

To compute the complex value of λ from the real integration parameter in ROM1.

METHOD

For integration along a straight path between the points a and b in the λ plane, λ and $d\lambda$ are

$$\lambda = a + (b - a)t$$

$$d\lambda = (b - a)dt$$

SYMBOL DICTIONARY

A	=	a
B	=	b
DXLAM	=	$b - a$
T	=	t
XLAM	=	λ

```
      SUBROUTINE LAMBDA (T,XLAM,DXLAM)
C
C     COMPUTE INTEGRATION PARAMETER XLAM=LAMBDA FROM PARAMETER T.
C
      COMPLEX A,B,XLAM,DXLAM
      COMMON /CNTOUR/ A,B
      DXLAM=B-A
      XLAM=A+DXLAM*T
      RETURN
      END
```

ROM1

PURPOSE

To integrate the Sommerfeld integrands between two points in λ by the method of variable interval-width Romberg integration.

METHOD

A and B in common block /CNTOUR/ are the ends of the integration path and are set before ROM1 is called. The integration parameter Z in ROM1 starts at zero and ends at one. The corresponding value of λ is determined by subroutine LAMBDA as

$$\lambda = A + (B - A)Z$$

Subroutine SAOA returns six integrand values which are handled simultaneously in loops throughout the code. The Romberg variable interval-width integration method will not be described in detail since it is the same as that used in subroutine INTX in the main NEC program. The convergence test in ROM1 requires that all six components satisfy the relative error tests simultaneously.

```
      SUBROUTINE ROM1 (N,SUM,NX)
C
C     ROM1 INTEGRATES THE 6 SOMMERFELD INTEGRALS FROM A TO B IN LAMBDA.
C     THE METHOD OF VARIABLE INTERVAL WIDTH ROMBERG INTEGRATION IS USED.
C
      COMPLEX A,B,SUM,G1,G2,G3,G4,G5,T00,T01,T10,T02,T11,T20
      COMMON /CNTOUR/ A,B
      DIMENSION SUM(6), G1(6), G2(6), G3(6), G4(6), G5(6), T01(6), T10(6
     1), T20(6)
      DATA NM,NTS,RX/131072,4,1.E-4/
      LSTEP=0
      Z=0.
      ZE=1.
      S=1.
      EP=S/(1.E4*NM)
      ZEND=ZE-EP
      DO 1 I=1,N
1     SUM(I)=(0.,0.)
      NS=NX
      NT=0
      CALL SAOA (Z,G1)
2     DZ=S/NS
      IF (Z+DZ.LE.ZE) GO TO 3
      DZ=ZE-Z
      IF (DZ.LE.EP) GO TO 17
3     DZOT=DZ*.5
      CALL SAOA (Z+DZOT,G3)
      CALL SAOA (Z+DZ,G5)
4     NOGO=0
      DO 5 I=1,N
      T00=(G1(I)+G5(I))*DZOT
      T01(I)=(T00+DZ*G3(I))*.5
      T10(I)=(4.*T01(I)-T00)/3.
C     TEST CONVERGENCE OF 3 POINT ROMBERG RESULT
      CALL TEST (REAL(T01(I)),REAL(T10(I)),TR,AIMAG(T01(I)),AIMAG(T10(I)
     1),TI,0.)
      IF (TR.GT.RX.OR.TI.GT.RX) NOGO=1
5     CONTINUE
      IF (NOGO.NE.0) GO TO 7
      DO 6 I=1,N
6     SUM(I)=SUM(I)+T10(I)
      NT=NT+2
      GO TO 11
7     CALL SAOA (Z+DZ*.25,G2)
      CALL SAOA (Z+DZ*.75,G4)
      NOGO=0
      DO 8 I=1,N
      T02=(T01(I)+DZOT*(G2(I)+G4(I)))*.5
      T11=(4.*T02-T01(I))/3.
      T20(I)=(16.*T11-T10(I))/15.
C     TEST CONVERGENCE OF 5 POINT ROMBERG RESULT
      CALL TEST (REAL(T11),REAL(T20(I)),TR,AIMAG(T11),AIMAG(T20(I)),TI,0
     1.)
      IF (TR.GT.RX.OR.TI.GT.RX) NOGO=1
8     CONTINUE
      IF (NOGO.NE.0) GO TO 13
9     DO 10 I=1,N
10    SUM(I)=SUM(I)+T20(I)
      NT=NT+1
11    Z=Z+DZ
      IF (Z.GT.ZEND) GO TO 17
      DO 12 I=1,N
12    G1(I)=G5(I)
      IF (NT.LT.NTS.OR.NS.LE.NX) GO TO 2
```

```
      NS=NS/2
      NT=1
      GO TO 2
13    NT=0
      IF (NS.LT.NM) GO TO 15
      IF (LSTEP.EQ.1) GO TO 9
      LSTEP=1
      CALL LAMBDA (Z,T00,T11)
      PRINT 18, T00
      PRINT 19, Z,DZ,A,B
      DO 14 I=1,N
14    PRINT 19, G1(I),G2(I),G3(I),G4(I),G5(I)
      GO TO 9
15    NS=NS*2
      DZ=S/NS
      DZOT=DZ*.5
      DO 16 I=1,N
      G5(I)=G3(I)
16    G3(I)=G2(I)
      GO TO 4
17    CONTINUE
      RETURN
C
18    FORMAT (38H ROM1 -- STEP SIZE LIMITED AT LAMBDA =,2E12.5)
19    FORMAT (10E12.5)
      END
```

SAOA

PURPOSE

To compute the integrands for the Sommerfeld integrals.

METHOD

The input to SAOA is the integration parameter T and constants in common block /EVLCOM/. The integration variable λ corresponding to T is obtained by calling subroutine LAMBDA. The values returned in array ANS are

$$ANS(1) = D_2 H_0^{(1)''}(\lambda\rho) e^{-\gamma_2(Z+Z')} \lambda^3 d\lambda/dT$$

$$ANS(2) = D_2 \gamma_2^2 H_0^{(1)}(\lambda\rho) e^{-\gamma_2(Z+Z')} \lambda\, d\lambda/dT$$

$$ANS(3) = -D_2 \gamma_2 H_0^{(1)'}(\lambda\rho) e^{-\gamma_2(Z+Z')} \lambda^2 d\lambda/dT$$

$$ANS(4) = \rho^{-1} D_2 H_0^{(1)'}(\lambda\rho) e^{-\gamma_2(Z+Z')} \lambda^2 d\lambda/dT$$

$$ANS(5) = D_2 H_0^{(1)}(\lambda\rho) e^{-\gamma_2(Z+Z')} \lambda d\lambda/dT$$

$$ANS(6) = k_1^{-1} D_1 H_0^{(1)}(\lambda\rho) e^{-\gamma_2(Z+Z')} \lambda d\lambda/dT$$

where $D_1 = \dfrac{1}{\gamma_1+\gamma_2} - \dfrac{k_2^2}{\gamma_2(k_1^2+k_2^2)}$

$$D_2 = \frac{1}{k_1^2\gamma_2+k_2^2\gamma_1} - \frac{1}{\gamma_2(k_1^2+k_2^2)} = \frac{k_2^2(\gamma_2-\gamma_1)}{\gamma_2(k_1^2+k_2^2)(k_1^2\gamma_2+k_2^2\gamma_1)}$$

$$\gamma_1 = [\lambda^2 - k_1^2]^{1/2}$$

$$\gamma_2 = [\lambda^2 - k_2^2]^{1/2}$$

$$k_1 = k_2(\epsilon_1 - j\sigma_1/\omega\epsilon_0)^{1/2}$$

$$k_2 = \omega\sqrt{\mu_0\epsilon_0}$$

The integrands given above are computed when JH >0. When JH $\leq$ 0, $H_0^{(1)}(\lambda\rho)$ is replaced by $2J_0(\lambda\rho)$. The functions γ_1 and γ_2 are computed from SA 24 to SA 29 so that the branch cuts are vertical. This is not necessary from SA 17 to SA 20 since for the Bessel function form the integration contour is confined to a different quadrant than the branch cuts.

To avoid loss of accuracy due to cancellation when λ is large, D_2 is computed from the approximation for $\gamma_2 - \gamma_1$:

$$\gamma_2 - \gamma_1 \approx \pm \left[\frac{1}{2}\,\frac{k_1^2-k_2^2}{\lambda} + \frac{1}{8}\,\frac{k_1^4-k_2^4}{\lambda^3} + \frac{1}{16}\,\frac{k_1^6-k_2^6}{\lambda^5} \right]$$

when $|\lambda|^2 \geq 100.|k_1|^2$.

The sign is:

$-$ for $\lambda_R < k_{2_R}$, $\lambda_I \geq 0$
$-$ for $\lambda_R < -k_{1_R}$, $\lambda_I < 0$
$+$ for $\lambda_R > k_{1_R}$, $\lambda_I \geq 0$
$+$ for $\lambda_R > -k_{2_R}$, $\lambda_I < 0$.

There is no cancellation and this approximation is not valid when

$k_{2_R} \leq \lambda_R \leq k_{1_R}$, $\lambda_I \geq 0$
or $-k_{1_r} \leq \lambda_R \leq -k_{2_R}$, $\lambda_I < 0$.

D_1 and D_2 are computed from SA 30 to SA 44.

ANS = integrand values
BO = $2J_\rho(\lambda\rho)$ or $H_0^{(1)}(\lambda\rho)$
BOP = $2J_0(\lambda\rho)/\rho$ or $H_0^{(1)'}(\lambda\rho)/\rho$
CGAM1 = γ_1
CGAM2 = γ_2
CK1 = k_1
CK1R = real part of k_1
CK1SQ = k_1^2
CK2 = k_2
CK2SQ = k_2^2
CKSM = $k_2^2/(k_1^2+k_2^2)$
COM = $\exp[-\gamma_2(Z+Z')]\lambda d\lambda/dT$ at SA 45
CT1 = $(k_1^2-k_2^2)/2$
CT2 = $(k_1^4-k_2^4)/8$
CT3 = $(k_1^6-k_2^6)/16$
DEN1 = D_1
DEN2 = D_2
DGAM = $\gamma_2 - \gamma_1$
DXL = $d\lambda/dT$
JH = flag to select Bessel or Hankel function form
RHO = ρ
SIGN = sign in approximation for $\gamma_2 - \gamma_1$
T = integration parameter
TKMAG = $100.|k_1|$
TSMAG = $100.|k_1|^2$
XL = λ
XLR = real part of λ
ZPH = $Z + Z'$

```
      SUBROUTINE SAOA (T,ANS)                                           SA   1
C                                                                       SA   2
C     SAOA COMPUTES THE INTEGRAND FOR EACH OF THE 6                     SA   3
C     SOMMERFELD INTEGRALS FOR SOURCE AND OBSERVER ABOVE GROUND         SA   4
C                                                                       SA   5
      COMPLEX ANS,XL,DXL,CGAM1,CGAM2,B0,B0P,COM,CK1,CK1SQ,CKSM,CT1,CT2,C SA   6
     1T3,DGAM,DEN1,DEN2                                                 SA   7
      COMMON /EVLCOM/ CKSM,CT1,CT2,CT3,CK1,CK1SQ,CK2,CK2SQ,TKMAG,TSMAG,C SA   8
     1K1R,ZPH,RHO,JH                                                    SA   9
      DIMENSION ANS(6)                                                  SA  10
      CALL LAMBDA (T,XL,DXL)                                            SA  11
      IF (JH.GT.0) GO TO 1                                              SA  12
C     BESSEL FUNCTION FORM                                              SA  13
      CALL BESSEL (XL*RHO,B0,B0P)                                       SA  14
      B0=2.*B0                                                          SA  15
      B0P=2.*B0P                                                        SA  16
      CGAM1=CSQRT(XL*XL-CK1SQ)                                          SA  17
      CGAM2=CSQRT(XL*XL-CK2SQ)                                          SA  18
      IF (REAL(CGAM1).EQ.0.) CGAM1=CMPLX(0.,-ABS(AIMAG(CGAM1)))         SA  19
      IF (REAL(CGAM2).EQ.0.) CGAM2=CMPLX(0.,-ABS(AIMAG(CGAM2)))         SA  20
      GO TO 2                                                           SA  21
C     HANKEL FUNCTION FORM                                              SA  22
    1 CALL HANKEL (XL*RHO,B0,B0P)                                       SA  23
      COM=XL-CK1                                                        SA  24
      CGAM1=CSQRT(XL+CK1)*CSQRT(COM)                                    SA  25
      IF (REAL(COM).LT.0..AND.AIMAG(COM).GE.0.) CGAM1=-CGAM1            SA  26
      COM=XL-CK2                                                        SA  27
      CGAM2=CSQRT(XL+CK2)*CSQRT(COM)                                    SA  28
      IF (REAL(COM).LT.0..AND.AIMAG(COM).GE.0.) CGAM2=-CGAM2            SA  29
    2 XLR=XL*CONJG(XL)                                                  SA  30
      IF (XLR.LT.TSMAG) GO TO 3                                         SA  31
      IF (AIMAG(XL).LT.0.) GO TO 4                                      SA  32
      XLR=REAL(XL)                                                      SA  33
      IF (XLR.LT.CK2) GO TO 5                                           SA  34
      IF (XLR.GT.CK1R) GO TO 4                                          SA  35
    3 DGAM=CGAM2-CGAM1                                                  SA  36
      GO TO 7                                                           SA  37
    4 SIGN=1.                                                           SA  38
      GO TO 6                                                           SA  39
    5 SIGN=-1.                                                          SA  40
    6 DGAM=1./(XL*XL)                                                   SA  41
      DGAM=SIGN*((CT3*DGAM+CT2)*DGAM+CT1)/XL                            SA  42
    7 DEN2=CKSM*DGAM/(CGAM2*(CK1SQ*CGAM2+CK2SQ*CGAM1))                  SA  43
      DEN1=1./(CGAM1+CGAM2)-CKSM/CGAM2                                  SA  44
      COM=DXL*XL*CEXP(-CGAM2*ZPH)                                       SA  45
      ANS(6)=COM*B0*DEN1/CK1                                            SA  46
      COM=COM*DEN2                                                      SA  47
      IF (RHO.EQ.0.) GO TO 8                                            SA  48
      B0P=B0P/RHO                                                       SA  49
      ANS(1)=-COM*XL*(B0P+B0*XL)                                        SA  50
      ANS(4)=COM*XL*B0P                                                 SA  51
      GO TO 9                                                           SA  52
    8 ANS(1)=-COM*XL*XL*.5                                              SA  53
      ANS(4)=ANS(1)                                                     SA  54
    9 ANS(2)=COM*CGAM2*CGAM2*B0                                         SA  55
      ANS(3)=-ANS(4)*CGAM2*RHO                                          SA  56
      ANS(5)=COM*B0                                                     SA  57
      RETURN                                                            SA  58
      END                                                               SA  59-
```

SECOND - see SECOND in main NEC program.

TEST - see TEST in main NEC program.

2. COMMON BLOCKS IN SOMNEC

COMMON/CNTOUR/ A, B

Routines Using /CNTOUR/

EVLUA, GSHANK, LAMBDA, ROM1

Parameters

A = start of integration interval

B = end of integration interval

A and B are used by subroutine LAMBDA to compute the complex value of λ from the real parameter supplied by ROM1.

COMMON/EVLCOM/ CKSM, CT1, CT2, CT3, CK1, CK1SQ, CK2, CK2SQ, TKMAG, TSMAG, CK1R, ZPH, RHO, JH

Routines Using /EVLCOM/

SOMNEC, EVLUA, SAOA

Parameters

See symbol dictionaries for subroutines

COMMON/GGRID/ AR1 (11, 10, 4), AR2 (17, 5, 4), AR3 (9, 8, 4), EPSCF, DXA(3), DYA(3), XSA(3), YSA(3), NXA(3), NYA(3)

Routines Using /GGRID/

SOMNEC (main program)

Parameters

AR1 = array for grid 1 (see Figure 12, Part I)

AR2 = array for grid 2

AR3 = array for grid 3

EPSCF = ϵ_c

For grid i, ARi(j, k, m) is the value of I_ρ^V, I_z^V, I_ρ^H, or I_ϕ^H for M = 1, ... 4 respectively at the point

$$R_1/\lambda = S_i + (j-1)\Delta R_i \qquad j = 1, \ldots N_i$$
$$\theta = T_i + (k-1)\Delta\theta_i \qquad k = 1, \ldots M_i$$

where S_i = XSA(i)

ΔR_i = DXA(i)

N_i = NXA(i)

T_i = YSA(i)

$\Delta\theta_i$ = DYA(i)

M_i = NYA(i)

XSA and DXA are in units of wavelength. YSA and DYA are in units of radians. The upper limit of grid 1 (XSA(2) = XSA(3)) and the upper limit of grid 2 (YSA(3)) may be changed and the densities of points may be changed. Boundaries that are zero should not be changed without modifying subroutine INTRP in NEC. The three grids must cover the region $0 \le R_1/\lambda \le 1$ and $0 \le \theta \le \pi/2$.

3. ARRAY DIMENSION LIMITATIONS

Number of Points in Interpolation Grids

Arrays:

COMMON/GGRID/AR1 (N_1, M_1, 4), AR2 (N_2, M_2, 4), AR3 (N_3, M_3, 4)

where $N_i \geq$ NXA(i) and $M_i \geq$ NYA(i)

The dimensions in common /GGRID/ in SOMNEC must be the same as the dimension of /GGRID/ in NEC.

Maximum Number of Iterations in GSHANK

Arrays:

Subroutine GSHANK: Q1 (6, MAXH), Q2 (6, MAXH)

where MAXH = maximum value of INT in GSHANK set at GS 13.

4. SOMNEC SUBROUTINE LINKAGE

Figure 17 shows the organization of subroutines in SOMNEC.

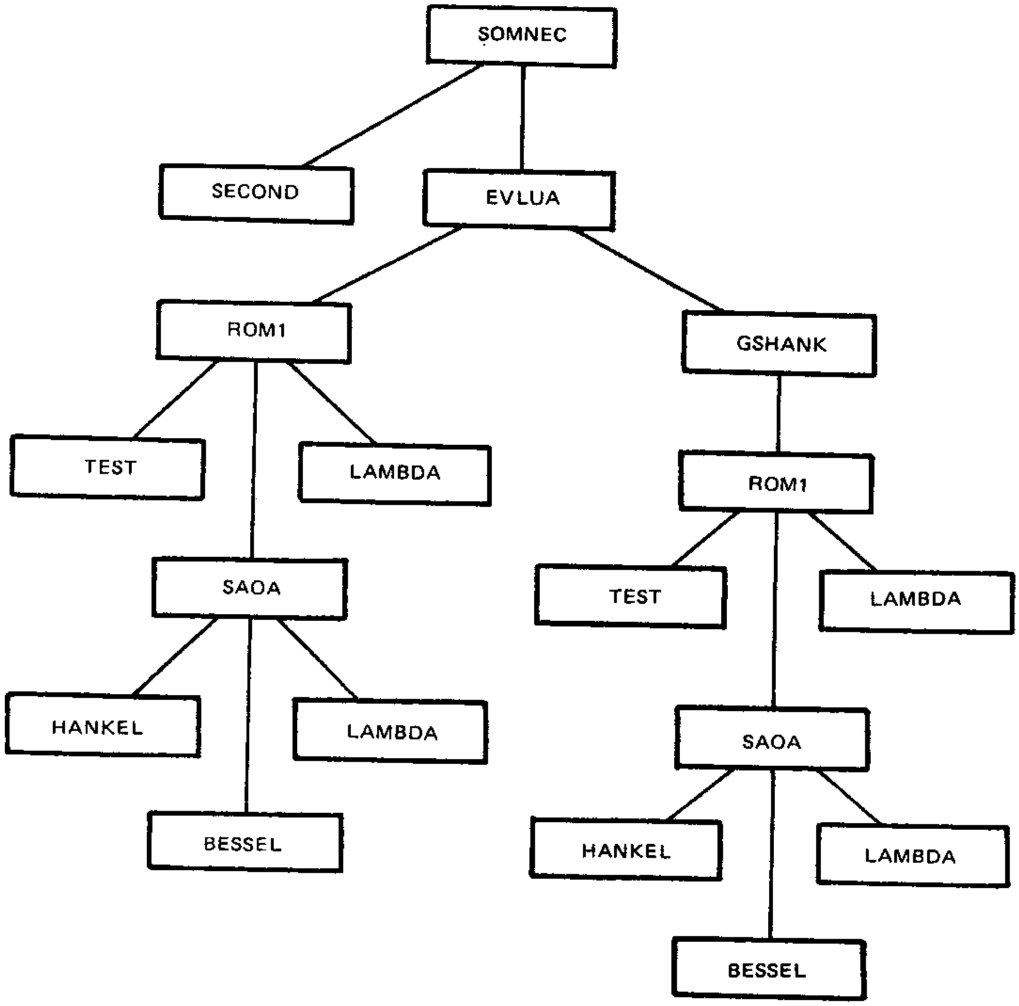


Figure 17. SOMNEC Subroutine Linkage Chart.

References

1. Ralston, A., A First Course in Numerical Analysis, McGraw-Hill, New York, 1965.

2. Norton, K. A., "The Propagation of Radio Waves Over the Surface of the Earth and in the Upper Atmosphere," Proceedings of the Institute of Radio Engineers, Vol. 25, No. 9, Sept. 1937.

3. Miller, E. K., "A Variable Interval Width Quadrature Technique Based on Romberg's Method," Journal of Computational Physics, Vol. 5, No. 2, April 1970.

4. Miller, E. K., and G. J. Burke, "Numerical Integration Methods," IEEE Transactions, Vol. Ap-17, No. 5, Sept. 1969.

5. Handbook of Mathematical Functions, edited by M. Abramowitz, National Bureau of Standards (U.S.), Applied Mathematics Series 55, 1964.

6. Shanks, D., "Non-Linear Transformations of Divergent and Slowly Convergent Sequences," J. Math. Phys. 24, 1, 1955.