SELF IMPEDANCE OF A CAVITY BACKED SLOT EXCITED BY A STRIPLINE

by

Norm VandenBerg
Linda P.B. Katehi

Radiation Laboratory
Electrical Engineering and
Computer Science Department
The University of Michigan
Ann Arbor, MI 48109-2122

March 1990

RL-870 = RL-870

SELF IMPEDANCE OF A CAVITY BACKED SLOT EXCITED BY A STRIPLINE

by

Norm VandenBerg
Linda P.B. Katehi

Radiation Laboratory
Electrical Engineering and
Computer Science Department
The University of Michigan
Ann Arbor, MI 48109-2122

March 1990

PROGRAM BX.FTN

I. COMMENTS - INPUT FILE

II. COMPARISON WITH SHAVIT'S DATA

III. OUTPUT FILE

IV. LIST OF PROGRAM

I.COMMENTS - INPUT FILE

I. bx.ftn PROGRAM DESCRIPTION:

1. The program included on the enclosed disk is called "bx.ftn". This is a method of moments based program which analyzes the coupling of a stripline to a slot in one of the ground planes. The slot region is enclosed in a boxed in area by vertical conducting walls which run parallel and perpendicular to the stripline center conductor. Small openings are left in the perpendicular walls to allow the strip to enter and exit the cavity. Also the slot passes over the strip at a right angle.

2. The slot is represented by magnetic currents backed by a conducting wall (which then replaces the slot) through the Equivalence Principle. The program is based on an exact Green's function for the cavity and a very accurate numerical evaluation of the exact Green's function for the external dielectric covered half-space above the slot.

   Because the internal problem involves the summation of cavity modes over two indices, it takes a lot of computer time to compute the necessary emittance matrix elements since many modes are needed for an accurate analysis (note that future analytical work may improve this situation). The program also could be rearranged to provide more data per run but it would be a more involved program and we are not planning to do this until later. However, from what we have seen so far, the results are VERY accurate.

3. The program is menu driven and should be quite self-explanatory. A sample menu is shown below. When the program starts it will ask for the name of a configuration file. This file contains the geometry definition for the problem of interest. When you run the program the first time you just enter a name of your choosing. The program will then prompt you for the necessary information.

On later runs, you may enter an existing file name and it will read the information from that file. You may then change things through the menu and run the program with the new changes (what you see in the menu is the data that will be used for the current run). On exiting the menu phase, you have the option to over-write the existing configuration file; run the new configuration without saving the configuration data (you must keep track yourself then); or you change the file name and save the configuration in the new file.

4. Sample menu listing:

```
$ bx

ENTER NAME OF CONFIGURATION DATA FILE;
bx.cfg
WARNING:FILE bx.cfg ALREADY EXISTS.READ EXISTING FILE?(<RETURN>=YES)
Name of geometry configuration file  [NAME] BX.CFG
Cavity rel.dielectric constant; REAL [ECR]: 2.2000000 IMAG [ECI]: ()
Cover rel. dielectric constant; REAL[EDR]:  2.2000000 IMAG [EDI]: ()

DIMENSIONS
ENTRY NORMALIZED TO HOST WAVELENGTH     [NL]F  normalized   (inches)
                                               ----------   --------
Frequency (GHz) of operation (lambda_o)   [F] 12.0000000  (0.9835712)
cavity dimension along microstrip (x)     [A]  0.6409062  (0.4250000)
cavity dimension along slot (y)           [B]  0.7781355  (0.5160000)
cavity height dimension (z)               [C]  0.0942509  (0.0625000)
width of microstrip line                  [W]  0.0384544  (0.0255000)
y center of microstrip (normalized to be) [YO] 0.2875969  (0.1484000)
width of slot                             [WS] 0.0452404  (0.0300000)
x center of slot (normalized to a)        [SO] 0.5000000  (0.2125000)
slot offset (normalized to be)            [OF] 0.0411822  (0.0212500)
thickness of cover                        [TD] 0.1885018  (0.1250000)
```

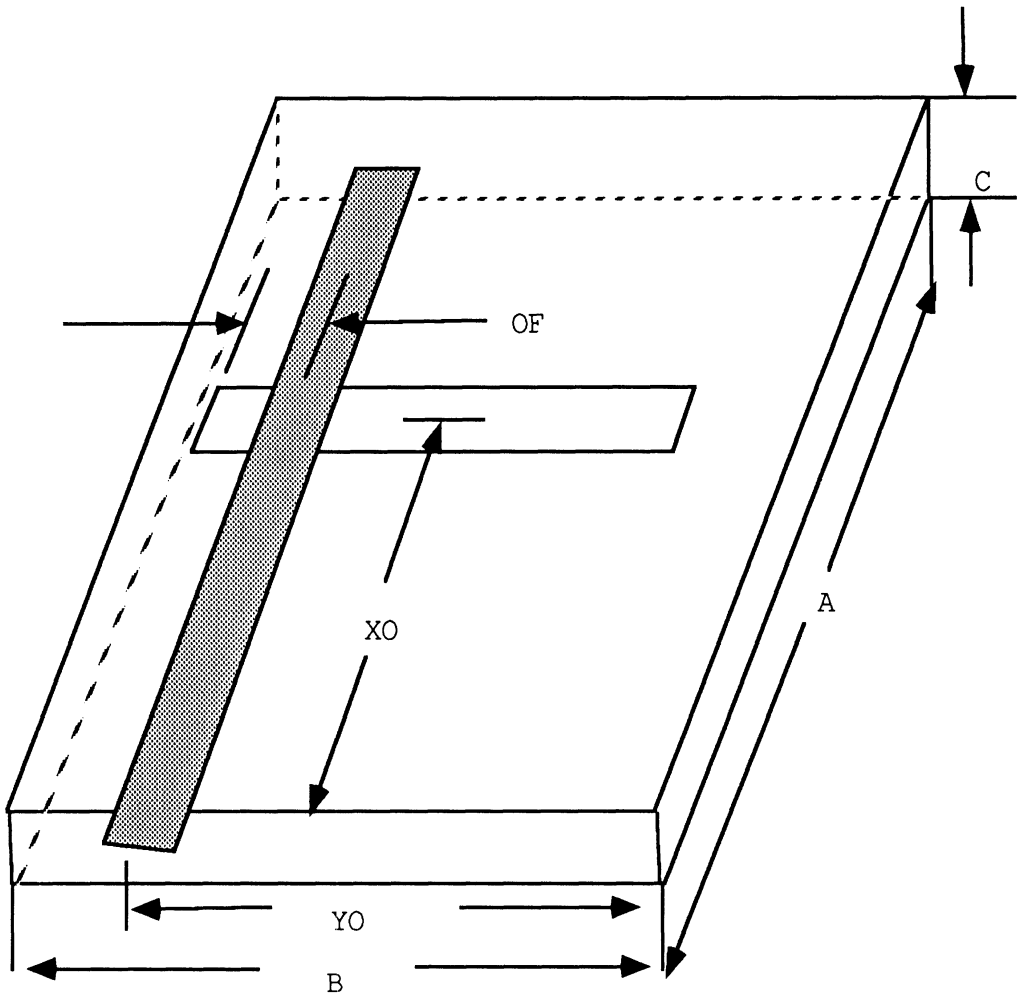


FIGURE 1

```
ENTER VARIABLE NAME [**]
FILE ALREADY EXISTS ---
Rename [R] to preserve existing '?*.cfg' file.
<RETURN> : runs new configuration but doesn't save in '?*.cfg' file.
[OVERRIDE] writes over existing '?*.cfg' file.
<return>
CALCULATING STRIP SELF-COUPLING SUB-MATRIX
.
.
.
.
```

The characters contained in [here] are the entries which allow you to change the variables. The [NL] entry (entered as just NL) allows you to change the units of entry. Currently, entries can be given normalized, in inches, or in centimeters.

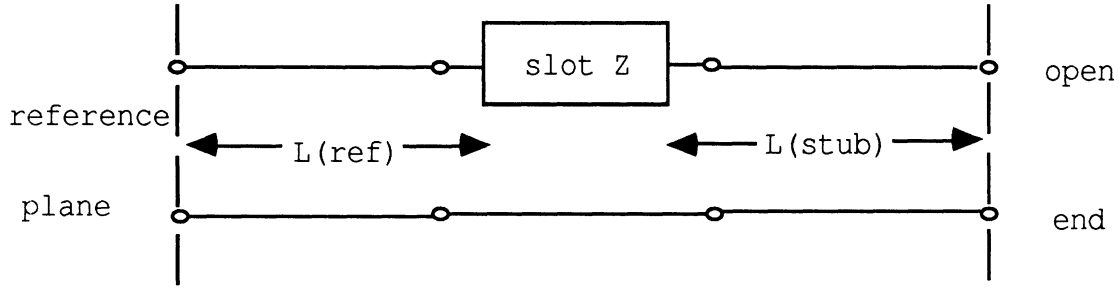
5. The program actually does the analysis twice. The first time gives a set of data which usually is pretty good by itself. The second time gives interleaves points which further refines and improves the accuracy of the results. The second pass is slightly faster than the first run for various reasons. You will have to experiment to see if it is worth your time on your computer to let the second pass run. The program stops after the first pass and prompts you to see if you want the second pass to be performed. (P.S. Let us know which computer you have and how long it takes.)

6. The output of the program is a listing of slot impedance vs. slot length for the geometry given. I should also appear in a file named 'slotdata' for each run. Be sure to copy this file to a new file name before the next run if you want to save the old data.

7. A slot of this type can be represented by a series impedance on the line. The output of this program is the value of this series impedance. The reference plane is located directly

below the slot and the impedance output is normalized to the line impedance (which is listed in the program output - but not in the 'slotdata' file. Sorry - I will add this also for later versions. Make a note when you do your runs of the line impedance. Also, when the line impedance is listed it says something about lossless cases. Losses are introduced by entering a non-zero imaginary part of the dielectric constant, however, this aspect of the program has not been tested AT ALL and I would not recommend using it).

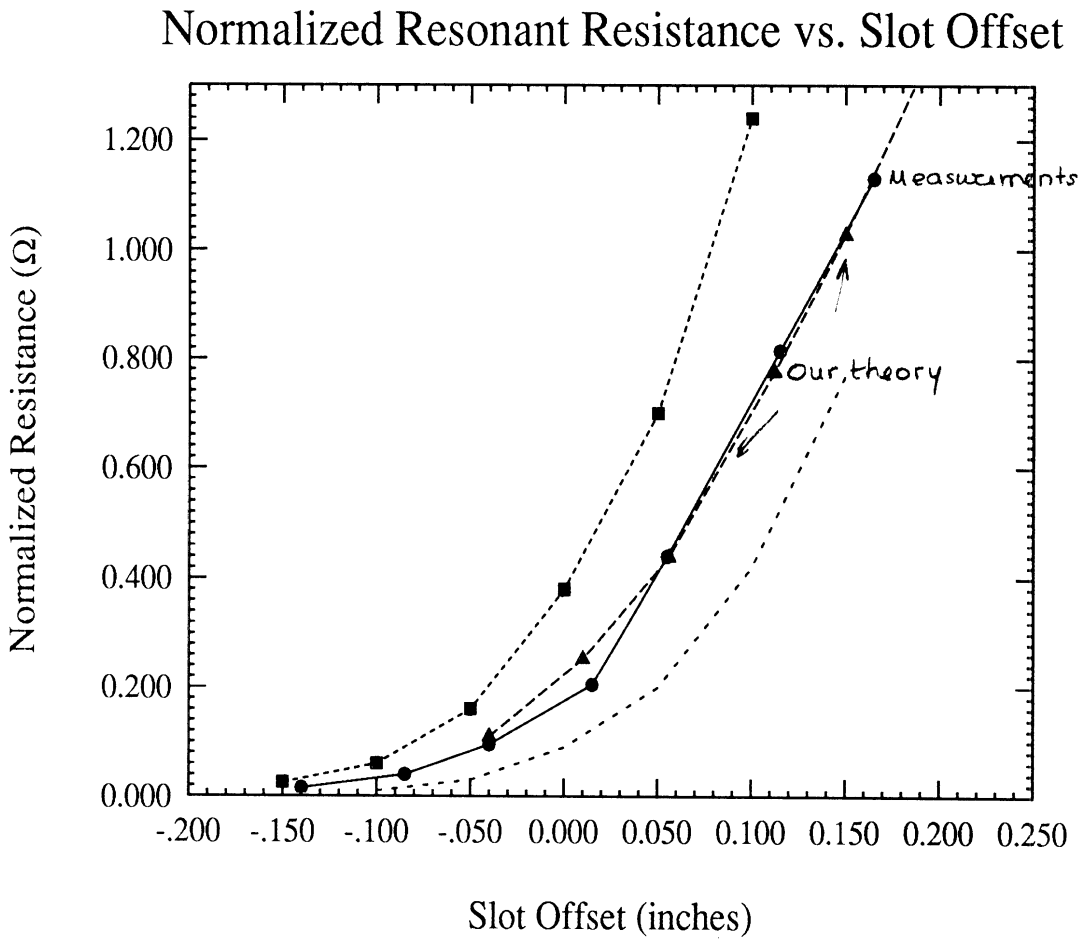
8. In order to evaluate the input impedance of your circuits, you will need to analyze the appropriate equivalent circuit. What you should use is something like:

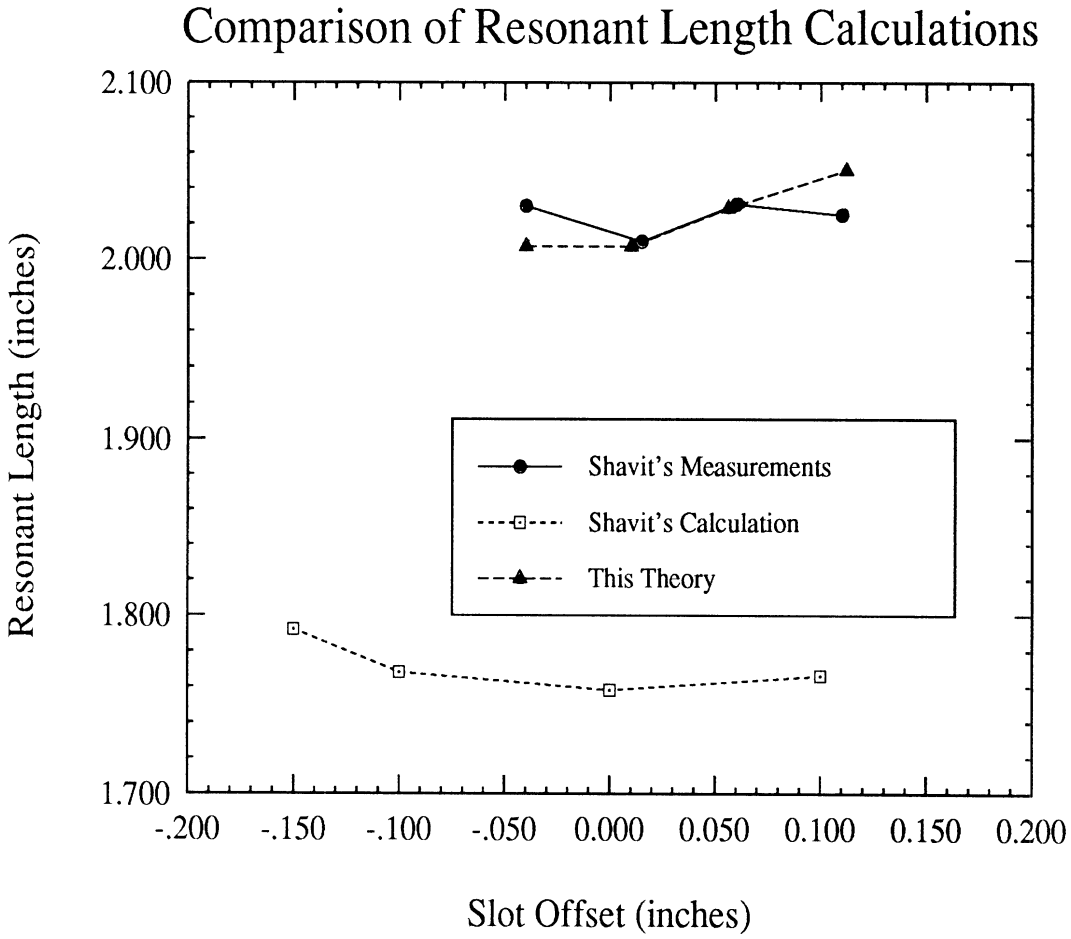


where L(ref) and L(stub) is measured to the center of the slot. The phase constant is the same as the material wave number since the structure supports the TEM mode. You might also have to find a way to compensate for the open end effect for the stripline if you haven't already done so, to get the most accurate results.

9. Also we have included some plots of sample data for your case. The parameters are as listed in the sample menu above (except the offset is variable). Note that the offset is the distance measured from the nearest end of the slot to the center of the stripline. We think this is the same as the definition you have been using since it is standard usage in the literature but one of the drawings we have received from you is misdrawn in this respect.

II. COMPARISON WITH SHAVIT'S DATA (THEORETICAL AND EXPERIMENTAL)


Normalized Resonant Resistance vs. Slot Offset
Normalized Resistance (Ω)
1.200
1.000
0.800
0.600
0.400
0.200
0.000
-.200
-.150
-.100
-.050
0.000
0.050
0.100
0.150
0.200
0.250
Slot Offset (inches)
Measurements
Our theory
Shavit's Measurements
Shavit's Calculation
This Theory
Cosine Assumption


Comparison of Resonant Length Calculations
Resonant Length (inches)
2.100
2.000
1.900
1.800
1.700
-.200
-.150
-.100
-.050
0.000
0.050
0.100
0.150
0.200
Slot Offset (inches)
Shavit's Measurements
Shavit's Calculation
This Theory

III. OUTPUT FILES

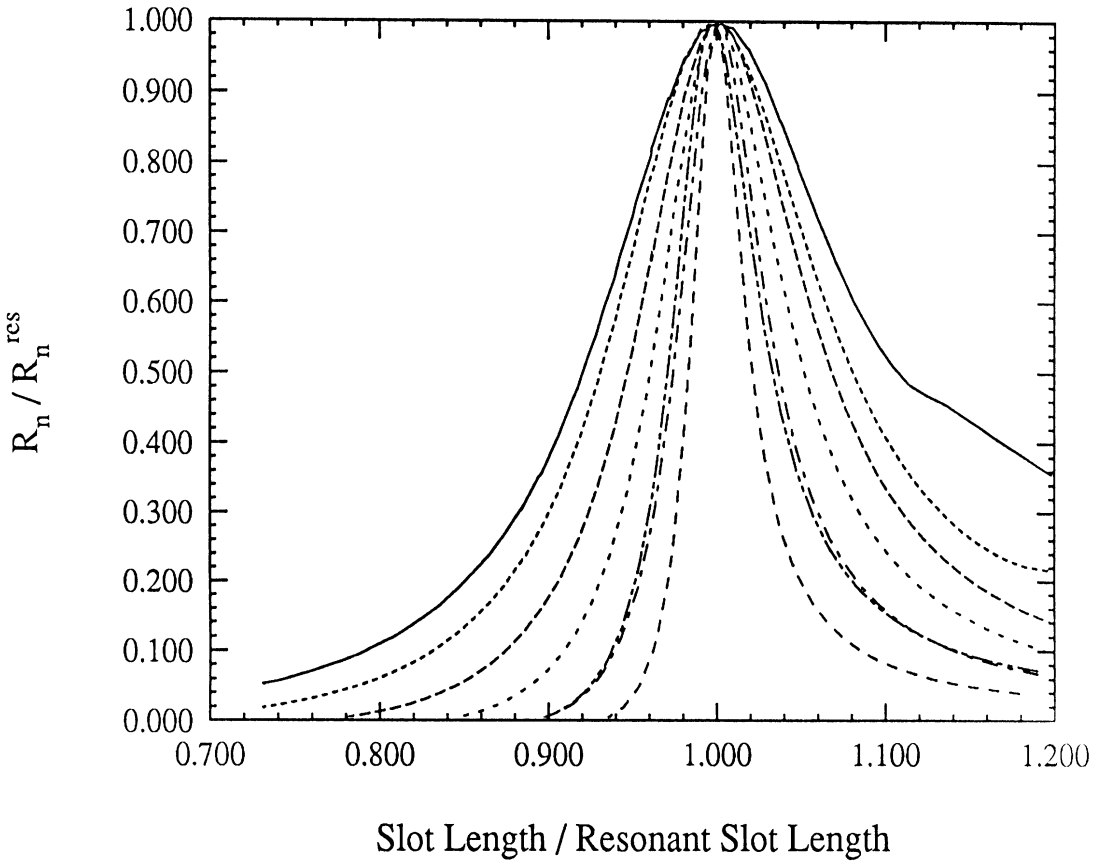
SLOT IMPEDANCE AS A FUNCTION OF SLOT LENGTH:
OFFSET : 0.0212 INCHES

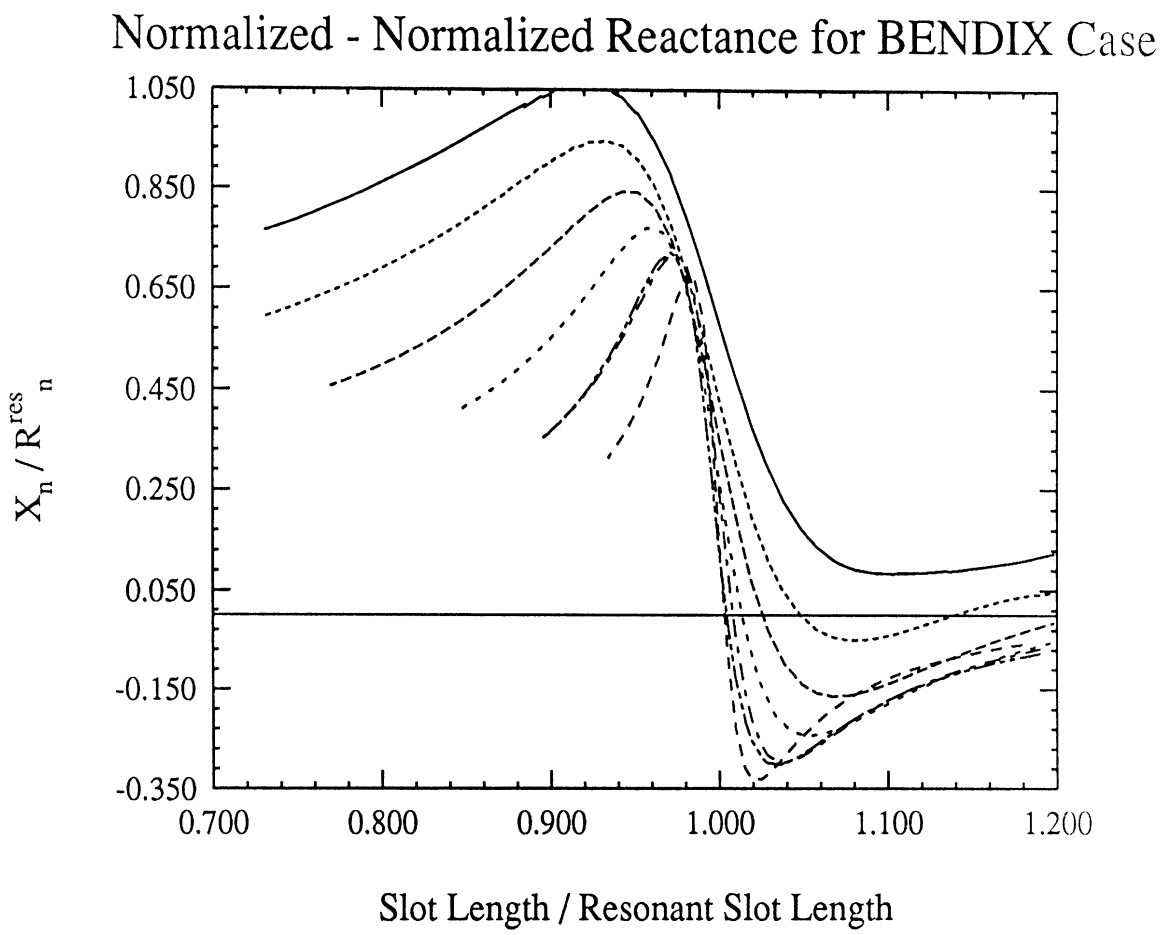
LENGTH	RESISTANCE	REACTANCE
0.3934	0.0537	0.0118
0.3903	0.0549	0.0107
0.3871	0.0559	0.0100
0.3840	0.0580	0.0084
0.3808	0.0610	0.0064
0.3777	0.0648	0.0041
0.3745	0.0694	0.0015
0.3723	0.0732	-0.0005
0.3714	0.0749	-0.0012
0.3687	0.0802	-0.0036
0.3682	0.0812	-0.0040
0.3651	0.0886	-0.0066
0.3651	0.0887	-0.0066
0.3620	0.0973	-0.0091
0.3616	0.0985	-0.0094
0.3588	0.1072	-0.0111
0.3580	0.1102	-0.0115
0.3557	0.1187	-0.0124
0.3544	0.1238	-0.0126
0.3525	0.1319	-0.0126
0.3508	0.1397	-0.0119
0.3494	0.1468	-0.0111
0.3472	0.1579	-0.0086
0.3462	0.1634	-0.0073
0.3437	0.1781	-0.0015
0.3431	0.1814	-0.0002
0.3401	0.1994	0.0109
0.3399	0.2003	0.0111
0.3368	0.2188	0.0275
0.3365	0.2202	0.0298
0.3336	0.2351	0.0495
0.3329	0.2379	0.0559
0.3305	0.2469	0.0768
0.3293	0.2491	0.0885
0.3273	0.2521	0.1080
0.3258	0.2508	0.1249
0.3242	0.2491	0.1404
0.3222	0.2419	0.1606
0.3210	0.2380	0.1708
0.3186	0.2235	0.1914
0.3179	0.2203	0.1965
0.3150	0.1992	0.2145
0.3147	0.1983	0.2160
0.3116	0.1748	0.2290
0.3114	0.1725	0.2291
0.3084	0.1516	0.2362
0.3079	0.1465	0.2364
0.3053	0.1302	0.2387
0.3043	0.1229	0.2381
0.3022	0.1111	0.2379
0.3007	0.1025	0.2360
0.2990	0.0946	0.2348
0.2971	0.0852	0.2315
0.2959	0.0804	0.2302

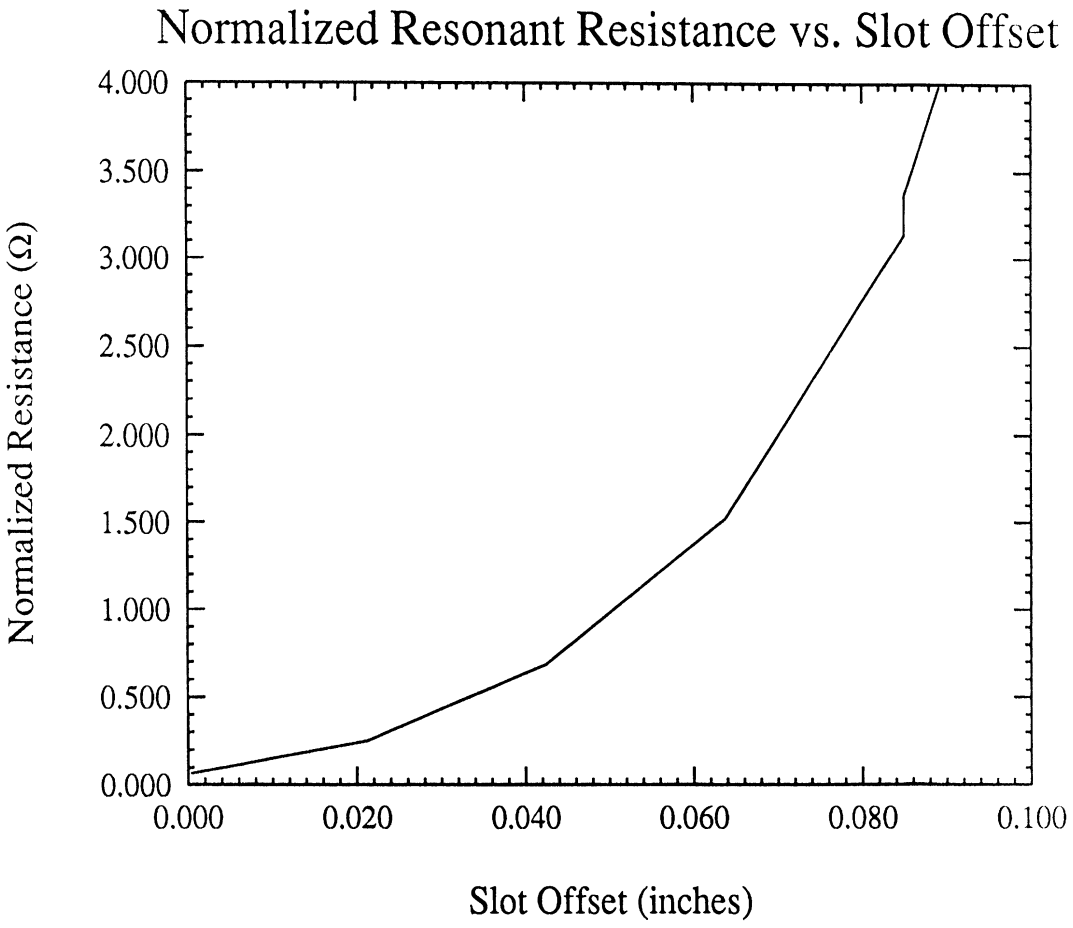
0.2935	0.0709	0.2256
0.2927	0.0684	0.2249
0.2900	0.0591	0.2191
0.2896	0.0583	0.2191
0.2864	0.0498	0.2132
0.2864	0.0494	0.2125
0.2833	0.0426	0.2074
0.2801	0.0365	0.2017
0.2770	0.0314	0.1963
0.2738	0.0270	0.1911
0.2707	0.0233	0.1862
0.2675	0.0200	0.1816
0.2644	0.0173	0.1772
0.2612	0.0149	0.1730
0.2581	0.0128	0.1691
0.2549	0.0110	0.1655
0.2518	0.0094	0.1620
0.2486	0.0080	0.1587
0.2455	0.0068	0.1556
0.2424	0.0057	0.1526
0.2392	0.0047	0.1498

Length for resistance maximum found between 0.3273 and 0.3258
Length for zero reactance found between 0.2392 and UUUUUUU

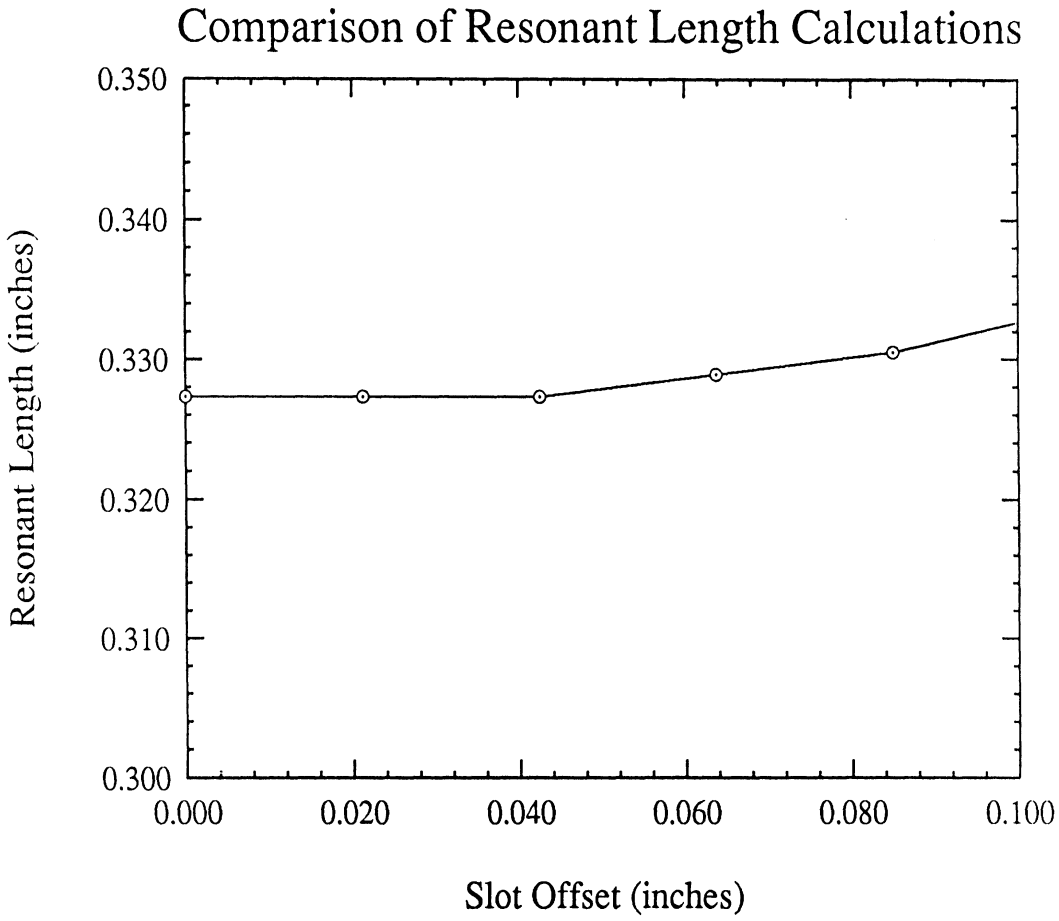
Normalized - Normalized Resistance for BENDIX Case




Normalized - Normalized Reactance for BENDIX Case
1.050
0.850
0.650
0.450
0.250
0.050
-0.150
-0.350
X_n / R^{res}_n
0.700
0.800
0.900
1.000
1.100
1.200
Slot Length / Resonant Slot Length
s = 0.00 in., y0 = .1414 in.
s = 0.02125 in., y0 = .1414 in.
s = 0.0425 in., y0 = .1414 in.
s = 0.06375 in., y0 = .1414 in.
s = 0.085 in., y0 = .1414 in.
s = 0.085 in., y0 = .1909 in.
s = 0.10625 in., y0 = .1414 in.



disregard
jump


Comparison of Resonant Length Calculations
Resonant Length (inches)
0.350
0.340
0.330
0.320
0.310
0.300
0.000
0.020
0.040
0.060
0.080
0.100
Slot Offset (inches)

IV. LIST OF PROGRAMS

```
      PROGRAM SERSLOT

C********* parameter declarations

      INTEGER ZDIM,HDIM,SLTDIM,RANGE,MNSTOP

      PARAMETER(HDIM=101)
      PARAMETER(SLTDIM=130)
C                                   ALSO SPECIFIED IN 'YIJDIE' SUBROUTINE
      PARAMETER(ZDIM=180)
C                                   ALSO SPECIFIED IN 'INVERT' SUBROUTINE
C                                   ALSO SPECIFIED IN 'TRIAG' SUBROUTINE
C     Maximum square dimension of zmatrx(i,j)
C     For greater than 500 change DIMENSION of cosi,cosi$ in MATRIX?? subroutines
C
      PARAMETER(RANGE=50)
C                                       NUMBER OF SCANS ON FIRST INVERT
C                                       SECOND INVERT USES NINT(.5*RANGE)
      PARAMETER(MNSTOP=1500)
C                                       MAXIMUM MODE NUMBER
      REAL*8 DUMMY
      COMPLEX*16 CDUMMY

C********* electrical declarations

      REAL*8 FREQ,MU8O,LAMBDA,K8B
      COMPLEX*16 EPSCAV,EPSLY1,EPSDIE,K8C,K8CS2
      COMPLEX*16  COEF11,COEF12,COEF33

C*********  GEOMETRY DECLARATIONS

      INTEGER NSECM,NSECS
      REAL*8 A,B,C,H,X8O,Y8O,L,L8U,W,STRIP(ZDIM)
      REAL*8 S8O,T8O,LS,L8S,WS,SLOT(ZDIM)
      REAL*8 SEGSLT(0:ZDIM),SL8OFF

C*********  matrix handling

      INTEGER DIMENZ
      CHARACTER*12 ANS
      REAL*8 E1,E2
      COMPLEX*16 ZMATRX(ZDIM,ZDIM),EVECT(ZDIM),HSLOT(ZDIM),REACT

C******** slot impedance results

      INTEGER POINTS
      REAL*8  LENGTH(2*RANGE),S1,S2
      COMPLEX*16 ZSLOT(2*RANGE)

C********** for splines and reaction integrations

      REAL*8 SPLPOS(HDIM)
      COMPLEX*16 HYSPLN(HDIM),HFIELD(HDIM),HY

C*********  loop declarations

      INTEGER I,J,ITER
```

```
C*********  program flow declarations

      LOGICAL DOF11,FROMLO,POINT
      INTEGER PASS

C******* FOR HALF SPACE PROGRAM *****************************

      INTEGER NS1,NS2,NOFF,INS,INS1S2
      REAL*8 DLX,WW,EER,ER,HH,TT,AA,AK0,AK,AKK,OFFSET(7)
      COMPLEX*16 YSD(SLTDIM)

C*************************************************************************

      CALL DEFINE(STRIP,SLOT,SL8OFF,
     &            DLX,WW,EER,NS1,NS2,NOFF,INS,INS1S2,
     &            ER,HH,TT,AA,AK0,AK,AKK,OFFSET,
     &            A,B,C,H,L,L8U,X8O,Y8O,W,NSECM,
     &            LS,L8S,S8O,T8O,WS,NSECS,
     &            FREQ,MU8O,EPSCAV,EPSDIE,EPSLY1,K8C,K8B,
     &            K8CS2,COEF11,COEF12,COEF33,POINT)

      WRITE(*,*)

      NSECS = 124
      NS1 = NSECS
      DIMENZ = NSECM + NSECS
      LAMBDA = 2.997925D10/(2.54D0 * FREQ )

C     SET INITIAL SLOT LENGTH TO .4 * LAMBDA

      LS = .4
      IF ( Y8O.LE.(B/2.D0)) THEN
         T8O = Y8O - SL8OFF + LS/2.D0
      ELSE
         T8O = Y8O + SL8OFF - LS/2.D0
      ENDIF
      L8S = LS/FLOAT(NSECS+1)
      DLX = L8S * DSQRT(EER)
      DO 1000 I=1,NSECS
         SLOT(I) = FLOAT(I) * L8S - LS/2.D0
 1000 CONTINUE

C********   INITIALIZE ZMATRX ARRAY TO ZERO  ***************************************

      DO 1001 I = 1, DIMENZ
         DO 1002 J = 1, DIMENZ
            ZMATRX(I,J) = DCMPLX(0.D0,0.D0)
 1002    CONTINUE
 1001 CONTINUE

      DOF11 = .TRUE.
      CALL TRIAG(STRIP,SLOT,ZMATRX,DOF11,
     &           DLX,WW,EER,NS1,NS2,NOFF,INS,INS1S2,
     &           ER,HH,TT,AA,AK0,AK,AKK,OFFSET,YSD,SLTDIM,
     &           A,B,C,H,L,L8U,X8O,Y8O,W,NSECM,
     &           LS,L8S,S8O,T8O,WS,NSECS,
     &           EPSCAV,EPSLY1,K8C,K8B,K8CS2,
     &           COEF11,COEF12,COEF33,MNSTOP,POINT)
```

```
C*******************************************************************************
C     DIVIDE HALF SECTION ROWS/COLUMNS BY TWO TO MAKE HALF SECTIONS OF THEM
C*******************************************************************************

      DO 1003 I = 1, DIMENZ
         ZMATRX(I,1) = .5D0 * ZMATRX(I,1)
         ZMATRX(1,I) = .5D0 * ZMATRX(1,I)
         ZMATRX(NSECM,I) = .5D0 * ZMATRX(NSECM,I)
         ZMATRX(I,NSECM) = .5D0 * ZMATRX(I,NSECM)
 1003 CONTINUE

C*******************************************************************************
C     CALCULATE INCIDENT FIELD

      CALL SCATT(REACT,SPLPOS,HFIELD,HDIM,
     &           B,C,Y8O,W,K8C,FREQ,MU8O,MNSTOP)

C   FIT CUBIC SPLINE TO HFIELD FOR LATER INTERPOLATION FOR REFL. COEF. INTEGRATION

      CALL CSPLNE( SPLPOS,HFIELD,HDIM,HYSPLN )

C*******************************************************************************
C     STRIP/SLOT DISCRETIZATION

      SEGSLT(0) = ( T8O - LS/2.D0 ) * LAMBDA
      DO 1004 I = 1, NSECS + 1
         SEGSLT(I) = SEGSLT(0) + FLOAT(I) * L8S*LAMBDA
 1004 CONTINUE

      DO 1005 I = 1, NSECS
         CALL CSPLNT( SPLPOS,HFIELD,HYSPLN,HDIM,SLOT(I) + T8O,HY )
         HSLOT(I) = HY
 1005 CONTINUE

C*******************************************************************************
C  MULTIPLY BY GALERKIN'S WEIGHTING FUNCTION ON SLOT FOR EXCITATION VECTOR

      E1 = - 2.D0 * ( SIN(.5D0*K8B*L8S)**2 )/(K8B * SIN(K8B*L8S) )
      E2 = - ( SIN(K8B*L8S)/K8B - L8S * COS(K8B*L8S) )
     &           / (L8S* K8B * SIN(K8B*L8S))

      EVECT(NSECM+1) = E1*HSLOT(2) + E2*(2*HSLOT(1)-HSLOT(2))
      DO 1006 I = 2, NSECS-1
         EVECT(NSECM+I) = E1 * ( HSLOT(I+1) + HSLOT(I-1) )
     &                  + E2*(2*HSLOT(I)-HSLOT(I-1)-HSLOT(I+1))
 1006 CONTINUE
      EVECT(NSECM+NSECS) = E1*HSLOT(NSECS-1)
     &                   + E2*(2*HSLOT(NSECS)-HSLOT(NSECS-1))

C*******************************************************************************

      PASS = 1
      ITER = RANGE

      IF (Y8O.GT.B/2) THEN
         FROMLO = .TRUE.
      ELSE
```

```
         FROMLO = .FALSE.
       ENDIF

       WRITE(*,*)
       WRITE(*,*) 'CALCULATING SLOT IMPEDANCE AS A FUNCTION OF LENGTH',
      &           ' (FIRST PASS)     '
       WRITE(*,*)

C*******************************************************************************

       CALL INVERT(PASS,SLOT,NSECM,L8S,NSECS,
      &            SEGSLT,ITER,LAMBDA,K8B,FROMLO,
      &            ZMATRX,EVECT,REACT,SPLPOS,HFIELD,HYSPLN,
      &            ZSLOT,LENGTH)

       POINTS = ITER

C*******************************************************************************
C      FIND RESONANT LENGTH
C      DETECT MAXIMUM IN RESISTANCE

       DO 1007 I = 1, POINTS

          IF ( DREAL(ZSLOT(I+1)).LT.DREAL(ZSLOT(I)) ) THEN
             IF ( DREAL(ZSLOT(I+1)).LT.DREAL(ZSLOT(I-1)) ) THEN
                LS = ( LENGTH(I-1) + LENGTH(I) ) / (2.D0*LAMBDA)
                WRITE(*,1999) LENGTH(I-1),LENGTH(I)
                GOTO 93
             ELSE
                LS = ( LENGTH(I) + LENGTH(I+1) ) / (2.D0*LAMBDA)
                WRITE(*,1999) LENGTH(I),LENGTH(I+1)
                GOTO 93
             ENDIF
          ENDIF
 1007  CONTINUE

C*******************************************************************************
   93  WRITE(*,*) ' '
       WRITE(*,*) 'RESONANT SLOT LENGTH ESTIMATE : ',LS*LAMBDA
       WRITE(*,*) ' '
       WRITE(*,*) 'THE SECOND PASS WILL FURTHER REFINE THE ESTIMATE '
       WRITE(*,*) 'OF RESONANT LENTH AND THE SLOT IMPEDANCE FUNCTION. '
       WRITE(*,*) ' '
       WRITE(*,*) 'IT WILL TAKE ALMOST AS LONG AS THE FIRST PASS. '
       WRITE(*,*) 'DO YOU WANT TO CONTINUE (<return> = yes )? '

       READ(*,'(A1)') ANS
       IF ((ANS(1:1).EQ.'Y').OR.
      &    (ANS(1:1).EQ.'y').OR.(ANS(1:1).EQ.' ')) THEN
       ELSE
          WRITE(*,*) ' '
          WRITE(*,*) 'SLOT IMPEDANCE AS A FUNCTION OF SLOT LENGTH: '
          WRITE(*,*) ' '
          WRITE(*,1995) SL8OFF*LAMBDA
          WRITE(*,*) ' '
          DO 1008 I = 1, POINTS
             WRITE(*,1998) LENGTH(I),DREAL(ZSLOT(I)),DIMAG(ZSLOT(I))
 1008     CONTINUE
```

```
         GOTO 4000
      ENDIF

      PASS = ITER+1
      ITER = NINT(.5*ITER)

C     RESET FOR NEW SLOT LENGTH AT MID-RESONANT POINT AFTER 1/2 OF ITERATIONS

      NSECS = NINT( 124 * LS/.4 )
      IF (NSECS.LT.99) NSECS = 99
      LS = LS * FLOAT(NSECS+1)
     &        / ( (NSECS+1) - FLOAT(NINT(ITER/2.D0)) )

      NS1   = NSECS
      DIMENZ = NSECM + NSECS

      IF ( Y8O.LE.(B/2.D0)) THEN
         T8O = Y8O - SL8OFF + LS/2.D0
      ELSE
         T8O = Y8O + SL8OFF - LS/2.D0
      ENDIF
      L8S = LS/FLOAT(NSECS+1)
      DLX = L8S * DSQRT(EER)
      DO 1009 I=1,NSECS
          SLOT(I) = FLOAT(I) * L8S - LS/2.D0
 1009 CONTINUE

      DOF11 = .FALSE.
      CALL TRIAG(STRIP,SLOT,ZMATRX,DOF11,
     &           DLX,WW,EER,NS1,NS2,NOFF,INS,INS1S2,
     &           ER,HH,TT,AA,AK0,AK,AKK,OFFSET,YSD,SLTDIM,
     &           A,B,C,H,L,L8U,X8O,Y8O,W,NSECM,
     &           LS,L8S,S8O,T8O,WS,NSECS,
     &           EPSCAV,EPSLY1,K8C,K8B,K8CS2,
     &           COEF11,COEF12,COEF33,MNSTOP,POINT)

C*****************************************************************************
C     DIVIDE HALF SECTION ROWS/COLUMNS BY TWO TO MAKE HALF SECTIONS OF THEM
C*****************************************************************************

      DO 1010 I = 1, NSECS
         ZMATRX(I+NSECM,1) = .5D0 * ZMATRX(I+NSECM,1)
         ZMATRX(1,I+NSECM) = .5D0 * ZMATRX(1,I+NSECM)
         ZMATRX(NSECM,I+NSECM) = .5D0 * ZMATRX(NSECM,I+NSECM)
         ZMATRX(I+NSECM,NSECM) = .5D0 * ZMATRX(I+NSECM,NSECM)
 1010 CONTINUE

C*****************************************************************************
C     STRIP/SLOT DISCRETIZATION

      SEGSLT(0) = ( T8O - LS/2.D0 ) * LAMBDA
      DO 9994 I = 1, NSECS + 1
         SEGSLT(I) = SEGSLT(0) + FLOAT(I) * L8S*LAMBDA
 9994 CONTINUE

      DO 1011 I = 1, NSECS
         CALL CSPLNT( SPLPOS,HFIELD,HYSPLN,HDIM,SLOT(I) + T8O,HY )
         HSLOT(I) = HY
```

```
 1011 CONTINUE

C*******************************************************************************
C  MULTIPLY BY GALERKIN'S WEIGHTING FUNCTION ON SLOT FOR EXCITATION VECTOR

      E1 = - 2.D0 * ( SIN(.5D0*K8B*L8S)**2 )/(K8B * SIN(K8B*L8S) )
      E2 = - ( SIN(K8B*L8S)/K8B - L8S * COS(K8B*L8S) )
     &           / (L8S* K8B * SIN(K8B*L8S))

      EVECT(NSECM+1) = E1*HSLOT(2) + E2*(2*HSLOT(1)-HSLOT(2))
      DO 1012 I = 2, NSECS-1
         EVECT(NSECM+I) = E1 * ( HSLOT(I+1) + HSLOT(I-1) )
     &                  + E2*(2*HSLOT(I)-HSLOT(I-1)-HSLOT(I+1))
 1012 CONTINUE
      EVECT(NSECM+NSECS) = E1*HSLOT(NSECS-1)
     &                   + E2*(2*HSLOT(NSECS)-HSLOT(NSECS-1))

      WRITE(*,*)
      WRITE(*,*) 'CALCULATING SLOT IMPEDANCE AS A FUNCTION OF LENGTH',
     &           ' (SECOND PASS) '
      WRITE(*,*)

C*******************************************************************************

      CALL INVERT(PASS,SLOT,NSECM,L8S,NSECS,
     &            SEGSLT,ITER,LAMBDA,K8B,FROMLO,
     &            ZMATRX,EVECT,REACT,SPLPOS,HFIELD,HYSPLN,
     &            ZSLOT,LENGTH )

      POINTS = POINTS + ITER
C*******************************************************************************
C   SORT LENGTHS AND IMPEDANCES INTO DESCENDING ORDER

      DO 1013 J = 2, POINTS
         DUMMY  = LENGTH(J)
         CDUMMY = ZSLOT(J)
         DO 1014 I = J-1, 1, -1
            IF (LENGTH(I).GE.DUMMY) GOTO 99
            LENGTH(I+1) = LENGTH(I)
            ZSLOT(I+1)  = ZSLOT(I)
 1014    CONTINUE
         I = 0
   99    LENGTH(I+1) = DUMMY
         ZSLOT(I+1)  = CDUMMY
 1013 CONTINUE

      WRITE(*,*)
      WRITE(*,*) 'SLOT IMPEDANCE AS A FUNCTION OF SLOT LENGTH: '
      WRITE(*,*)
      WRITE(*,1995) SL8OFF*LAMBDA
      WRITE(*,*)

 4000 OPEN(UNIT=10,FILE='SLOTDATA')
      WRITE(10,*)
      WRITE(10,*) 'SLOT IMPEDANCE AS A FUNCTION OF SLOT LENGTH: '
      WRITE(10,*) ' (data written to file: SLOTDATA )'
      WRITE(10,1995) SL8OFF*LAMBDA
      WRITE(10,*)
```

```
      WRITE(10,*) 'LENGTH     RESISTANCE     REACTANCE '
      WRITE(10,*) '------     ----------     --------- '
      DO 1015 I = 1, POINTS
         WRITE(*,1998) LENGTH(I),DREAL(ZSLOT(I)),DIMAG(ZSLOT(I))
         WRITE(10,'(1X,3(F7.4,5X))')
     &                          LENGTH(I),DREAL(ZSLOT(I)),DIMAG(ZSLOT(I))
 1015 CONTINUE

C*******************************************************************************
C     FIND RESONANT LENGTH

C     DETECT MAXIMUM IN RESISTANCE

      DO 1016 I = 1, POINTS

         IF ( DREAL(ZSLOT(I+1)).LT.DREAL(ZSLOT(I)) ) THEN
            IF ( DREAL(ZSLOT(I+1)).LT.DREAL(ZSLOT(I-1)) ) THEN
               WRITE(*,1997) LENGTH(I-1),LENGTH(I)
               WRITE(10,1997) LENGTH(I-1),LENGTH(I)
               GOTO 1017
            ELSE
               WRITE(*,1997) LENGTH(I),LENGTH(I+1)
               WRITE(10,1997) LENGTH(I),LENGTH(I+1)
               GOTO 1017
            ENDIF
         ENDIF
 1016 CONTINUE

C     FIND ZERO REACTANCE POINT

 1017 DO 1018 I = MAX0(1,I-5), POINTS
         S1 = SIGN(1.D0,DIMAG(ZSLOT(I)))
         S2 = SIGN(1.D0,DIMAG(ZSLOT(I+1)))
         IF ((S1*S2).EQ.-1.D0) THEN
            WRITE(*,1996) LENGTH(I),LENGTH(I+1)
            WRITE(10,1996) LENGTH(I),LENGTH(I+1)
            GOTO 1019
         ENDIF
 1018 CONTINUE

 1019 CLOSE(10)

C*******************************************************************************
C*******************************************************************************

 1995 FORMAT(1X,'OFFSET : ',F7.4,' INCHES')
 1996 FORMAT(1X,'Length for zero reactance found between',F7.4,
     &                                          ' and ',F7.4)
 1997 FORMAT(1X,'Length for resistance maximum found between',F7.4,
     &                                            ' and ',F7.4)
 1998 FORMAT(1X,'LENGTH : '     , F7.4,
     &          ' RESISTANCE : ', F7.4,
     &          ' REACTANCE : ' , F7.4 )
 1999 FORMAT(1X,'Resonant length found between',F7.4,' and ',F7.4)

      END
```

```
C*******************************************************************************
C*******************************************************************************
C*******************************************************************************
      SUBROUTINE INVERT(PASS,SLOT,NSECM,L8S,NSECS,
     &                  SEGSLT,ITER,LAMBDA,K8B,FROMLO,
     &                  ZMATRX,EVECT,REACT,SPLPOS,HFIELD,HYSPLN,
     &                  ZSLOT,LENGTH)

C********* parameter declarations

      INTEGER ZDIM,HDIM

      PARAMETER(ZDIM=180)
      PARAMETER(HDIM=101)

C********* electrical declarations

      REAL*8 LAMBDA,K8B

C*********  GEOMETRY DECLARATIONS

      INTEGER NSECM,NSECS

      REAL*8 L8S,SLOT(1),SEGSLT(0:ZDIM)

C*********  matrix handling

      INTEGER DIMENZ
      REAL*8     LENGTH(1)
      COMPLEX*16 ZMATRX(ZDIM,ZDIM),AMATRX(ZDIM,ZDIM)
      COMPLEX*16 EVECT(ZDIM),CURRNT(ZDIM),REACT,ZSLOT(1)

C********** for splines and reaction integrations

      INTEGER TRPNTS
      REAL*8 X,STEP,SPLPOS(HDIM),SLTPOS(ZDIM)

      COMPLEX*16 HYSPLN(HDIM),KYSPLN(ZDIM)
      COMPLEX*16 HFIELD(HDIM),KCURR(ZDIM),HY,KY,HY1,KY1

C********** reflection coefficient declarations

      COMPLEX*16 GAMMA

C*********  loop declarations

      INTEGER I,J,ITER,LOOPS

C*********  program flow declarations

      LOGICAL FROMLO
      INTEGER PASS

C*******************************************************************************
C                     FOR INVERTING MATRIX

      INTEGER IV(ZDIM)
```

```
      REAL*8 RC
      COMPLEX*16 DET(2),SV1(ZDIM),WORK(ZDIM)

C*************************************************************************
C*************************************************************************

      DIMENZ = NSECM + NSECS
      LOOPS = 1

C*************************************************************************
C                  INVERT MATRIX
C*************************************************************************

  998 IF (LOOPS.LE.ITER) THEN

         DO 10 I = 1, DIMENZ
            DO 11 J = 1, DIMENZ
               AMATRX(I,J) = ZMATRX(I,J)
   11       CONTINUE
   10    CONTINUE

C*************************************************************************
C*************************************************************************

         CALL CGECO(AMATRX,ZDIM,DIMENZ,IV,RC,SV1)
         CALL CGEDI(AMATRX,ZDIM,DIMENZ,IV,DET,WORK,01)

C*************************************************************************
C*************************************************************************

C*************************************************************************
C  COMPUTE CURRENTS BY MULTIPLICATION OF INVERTED MATRIX WITH EXCITATION VECTOR
C*************************************************************************

         DO 12 I = 1, DIMENZ + 1
            CURRNT(I) = DCMPLX(0.D0,0.D0)
   12    CONTINUE

C*************************************************************************
C                                          DETERMINE CURRENTS FOR ALL CASES
C*************************************************************************
C                                        MULTIPLY MATRIX AND EXCITATION VECTOR
         DO 13 I = 1, DIMENZ
            DO 14 J = 1, DIMENZ
               CURRNT(I) = CURRNT(I) + AMATRX(I,J)*EVECT(J)
   14       CONTINUE
   13    CONTINUE

C     FIT A CUBIC SPLINE TO THE MAGNETIC SLOT VOLTAGE
C     INTEGRATE USING SPLINES TO INTERPOLATE BETWEEN POINTS

         SLTPOS(1) = SEGSLT(0) / LAMBDA
         KCURR(1) = DCMPLX(0.D0,0.D0)
         DO 15 I = 1, NSECS
            SLTPOS(I+1) = SEGSLT(I) / LAMBDA
            KCURR(1+I) = CURRNT(NSECM+I)
   15    CONTINUE
         SLTPOS(NSECS+2) = SEGSLT(NSECS+1) / LAMBDA
```

```
         KCURR(NSECS+2) = DCMPLX(0.D0,0.D0)

         CALL CSPLNE(SLTPOS,KCURR,NSECS+2,KYSPLN)

         GAMMA = DCMPLX(0.D0,0.D0)

C SET NUMBER OF POINTS ON SLOT FOR TRAPEZOIDAL INTEGRATION
C SET CONSTANT STEP SIZE FOR SIMPLE ROUTINE
         TRPNTS = 1000
         STEP = ((SEGSLT(NSECS+1)-SEGSLT(0))/LAMBDA)
     &            /FLOAT(TRPNTS)

         X = SEGSLT(0) / LAMBDA
         CALL CSPLNT(SLTPOS,KCURR,KYSPLN,NSECS+2,X,KY1)
         CALL CSPLNT(SPLPOS,HFIELD,HYSPLN,HDIM,X,HY1)
         DO 16 I = 1, TRPNTS
            X = X + STEP
            HY = HY1
            KY = KY1
            CALL CSPLNT(SLTPOS,KCURR,KYSPLN,NSECS+2,X,KY1)
            CALL CSPLNT(SPLPOS,HFIELD,HYSPLN,101,X,HY1)
            GAMMA = GAMMA + .5D0 * STEP * ( HY * KY + HY1 * KY1 )
   16    CONTINUE

C*******************************************************************************

         GAMMA =  .5D0 * GAMMA/REACT
         ZSLOT(PASS+LOOPS-1) =   2.D0 * GAMMA / ( 1.D0 - GAMMA )
         LENGTH(PASS+LOOPS-1) = L8S*(NSECS+1)*LAMBDA
         WRITE(*,1999) DREAL(ZSLOT(PASS+LOOPS-1)),
     &                 DIMAG(ZSLOT(PASS+LOOPS-1)),
     &                 LENGTH(PASS+LOOPS-1)
         IF (DREAL(ZSLOT(PASS+LOOPS-1)).LE.0.D0) THEN
            ITER = LOOPS - 1
            GOTO 999
         ENDIF

 1999 FORMAT(1X,' Slot Resistance : ',F7.4,' Reactance : ',F7.4,
     &          '   Length    : ',F7.4,' inches')

C*******************************************************************************
C ROUTINES TO MANIPULATE MATRIX FOR SHORTENING SLOT LENGTH BY REMOVING SUBSECTIONS
C*******************************************************************************

         IF (FROMLO) THEN
C                                        REMOVES FIRST SLOT SUBSECTION - ROWS FIRST
            DO 17 J = NSECM+1, DIMENZ - 1
               DO 18 I = 1, DIMENZ
                  ZMATRX(J,I) = ZMATRX(J+1,I)
   18          CONTINUE
               EVECT(J) = EVECT(J+1)
   17       CONTINUE
            DO 19 J = 1, DIMENZ - 1
C                                        REMOVES FIRST SLOT SUBSECTION - NOW COLUMN
               DO 20 I = NSECM+1,DIMENZ - 1
                  ZMATRX(J,I) = ZMATRX(J,I+1)
   20          CONTINUE
   19       CONTINUE
```

```
            DO 21 I = 0, NSECS
               SEGSLT(I) = SEGSLT(I+1)
   21       CONTINUE
         ENDIF
C                                      FOR even NUMBERS NEED ONLY REDUCE DIMENSION
         NSECS = NSECS - 1
         DIMENZ = DIMENZ - 1
C                                      REDUCES DIMENSION FOR BOTH EVEN AND ODD CASE

         WRITE(*,*)
         LOOPS = LOOPS + 1
         GOTO 998
      ENDIF

  999 RETURN
      END

C*******************************************************************************
C*******************************************************************************
C*******************************************************************************
C
C        TRIAG CALCULATES THE ELEMENTS OF THE IMPEDANCE MATRIX
C        FOR A METHOD OF MOMENTS SOLUTION TO THE COUPLING OF A
C        MICROSTRIP LINE IN A SLOTTED CAVITY PROBLEM.
C
C*******************************************************************************

C*******************************************************************************
C            M AND N LOOP ITERATIONS:
C               SUM UP MODES FROM M = MN :
C                           -  FIRST N FROM N = MN TO NSTOP
C                           -  THEN M FROM M = MN TO MSTOP
C                           -  THEN INCREMENT MN AND REPEAT
C            THIS SUMS UP MODES ALONG A DIAGONAL M = N
C*******************************************************************************

      SUBROUTINE TRIAG(STRIP,SLOT,ZMATRX,DOF11,
     &                 DLX,WW,EER,NS1,NS2,NOFF,INS,INS1S2,
     &                 ER,HH,TT,AA,AK0,AK,AKK,OFFSET,YSD,SLTDIM,
     &                 A,B,C,H,L,L8U,X8O,Y8O,W,NSECM,
     &                 LS,L8S,S8O,T8O,WS,NSECS,
     &                 EPSCAV,EPSLY1,K8C,K8B,K8CS2,
     &                 COEF11,COEF12,COEF33,MNSTOP,POINT)

C******** parameter declarations

      INTEGER SLTDIM,ZDIM
      PARAMETER(ZDIM=180)

      REAL*8 PI
      PARAMETER(PI=3.14159265358979324D0)

C*********  ELECTRICAL PARAMETER DECLARATIONS

      REAL*8 K8M,K8N,K8B
      COMPLEX*16 EPSCAV,EPSLY1,K8C,K8CS2,COEF11,COEF12,COEF33

C*********  GEOMETRY DECLARATIONS
```

```
      REAL*8 A,B,C,H,X8O,Y8O,L,L8U,W,STRIP(ZDIM)
      REAL*8 S8O,T8O,LS,L8S,WS,SLOT(ZDIM)
      INTEGER NSECM,NSECS

C*********  GREEN'S FUNCTION AND IMMITTANCE MATRIX DECLARATIONS

      COMPLEX*16 IEOPL(2*ZDIM),IEOMN(ZDIM)
      COMPLEX*16 ZMATRX(ZDIM,ZDIM),AMATRX(ZDIM,ZDIM)

C*********  CONVERGENCE AND LOOP DECLARATIONS

      INTEGER I,J,JJ,M,N,MNSTOP,MN
      LOGICAL DONE,DOF11,POINT

C******* FOR HALF SPACE PROGRAM *****************************

      INTEGER NS1,NS2,NOFF,INS,INS1S2
      REAL*8 DLX,WW,EER,ER,HH,TT,AA,AK0,AK,AKK,OFFSET(1)
      COMPLEX*16 YSD(SLTDIM)

C*************************************************************************
C     FOR THE ELEMENTS IN THE CAVITY
C*************************************************************************

      IF (DOF11) THEN
C*************************************************************************
C                                                        FIRST THE Z11 MATRIX
C*************************************************************************
C********   INITIALIZE IEO VECTORS TO ZERO  ***********************************

      WRITE(*,*) 'CALCULATING STRIP SELF-COUPLING SUB-MATRIX'

      DO 1101 I = 1, 2 * NSECM - 1
         IEOPL(I) = DCMPLX(0.D0,0.D0)
 1101 CONTINUE
      DO 1102 I = 1, NSECM
         IEOMN(I) = DCMPLX(0.D0,0.D0)
 1102 CONTINUE

C****** SET UP M AND N LOOP ITERATIONS

      DO 1103 MN = 1, MNSTOP
         M = MN
         N = 1
 1104    IF (M.GE.0) THEN
            K8M = FLOAT(M)*PI/A
            K8N = FLOAT(N)*PI/B
            CALL MTRX11(K8M,K8N,STRIP,IEOPL,IEOMN,ZDIM,
     &                  A,B,C,H,L,L8U,X8O,Y8O,W,NSECM,
     &                  EPSCAV,EPSLY1,K8C,K8B,K8CS2,COEF11)
            M = M - 1
            N = N + 1
            GOTO 1104
         ENDIF
 1103 CONTINUE

C******* WRITE IEO VECTORS TO ZMATRX
```

```
      DO 1105 I = 1, NSECM
         DO 1106 J = I, NSECM
            ZMATRX(I,J) = .5D0 * ( IEOPL(I+J-1) + IEOMN(J-I+1) )
            ZMATRX(J,I) = ZMATRX(I,J)
 1106    CONTINUE
 1105 CONTINUE

      ENDIF

C*************************************************************************
C                                                        NOW THE Y12/Z31 MATRIX
C*************************************************************************

 9996 WRITE(*,*) 'CALCULATING SLOT-TO-STRIP COUPLING SUB-MATRIX'

      DO 1200 I = 1, NSECM
         DO 1201 J = 1, NSECS
            ZMATRX(I,J+NSECM) = DCMPLX(0.D0,0.D0)
 1201    CONTINUE
 1200 CONTINUE

      MN = MNSTOP - 1

      DO 1202 N = 1, MNSTOP
         DONE = .FALSE.
         M = 0
 1203    IF ((M.LE.MN).AND.(.NOT.DONE)) THEN
            K8M = FLOAT(M)*PI/A
            K8N = FLOAT(N)*PI/B
            CALL MTRX12(K8M,K8N,DONE,STRIP,SLOT,ZMATRX,ZDIM,
     &                  A,B,C,H,L,L8U,X8O,Y8O,W,NSECM,
     &                  LS,L8S,S8O,T8O,WS,NSECS,
     &                  EPSCAV,EPSLY1,K8C,K8B,K8CS2,COEF12)
            M = M + 1
            GOTO 1203
         ENDIF
         MN = MN - 1
 1202 CONTINUE

C*************************************************************************
C                              NOW DO THE Y21 MATRIX IF POINT MATCH WAS SPECIFIED
C*************************************************************************

 9997 WRITE(*,*) 'CALCULATING STRIP-TO-SLOT COUPLING SUB-MATRIX'

      DO 2100 I = 1, NSECS
         DO 2101 J = 1, NSECM
            ZMATRX(I+NSECM,J) = DCMPLX(0.D0,0.D0)
 2101    CONTINUE
 2100 CONTINUE

      MN = MNSTOP - 1

      DO 2102 N = 1, MNSTOP
         DONE = .FALSE.
         M = 0
 2103    IF ((M.LE.MN).AND.(.NOT.DONE)) THEN
```

```
            K8M = FLOAT(M)*PI/A
            K8N = FLOAT(N)*PI/B
            CALL MTRX21(K8M,K8N,DONE,STRIP,SLOT,ZMATRX,ZDIM,
     &                  A,B,C,H,L,L8U,X8O,Y8O,W,NSECM,
     &                  LS,L8S,S8O,T8O,WS,NSECS,
     &                  EPSCAV,EPSLY1,K8C,K8B,K8CS2,COEF12,POINT)
            M = M + 1
            GOTO 2103
         ENDIF
         MN = MN - 1
 2102 CONTINUE

C*************************************************************************
C                                                          FINALLY THE Y33 MATRIX
C*************************************************************************

 9998 WRITE(*,*) 'CALCULATING INTERNAL SLOT SELF-COUPLING SUB-MATRIX'

C********   INITIALIZE IEO VECTORS TO ZERO  ************************************

      DO 3301 I = 1, 2 * NSECS - 1
         IEOPL(I) = DCMPLX(0.D0,0.D0)
 3301 CONTINUE
      DO 3302 I = 1, NSECS
         IEOMN(I) = DCMPLX(0.D0,0.D0)
 3302 CONTINUE

C********   SET UP M AND N LOOP ITERATIONS  ************************************

      DO 3303 MN = 1, MNSTOP
         M = MN
         N = 0
 3304    IF (M.GE.0) THEN
            K8M = FLOAT(M)*PI/A
            K8N = FLOAT(N)*PI/B
            CALL MTRX33(K8M,K8N,SLOT,IEOPL,IEOMN,ZDIM,
     &                  A,B,C,H,LS,L8S,S8O,T8O,WS,NSECS,
     &                  EPSCAV,EPSLY1,K8C,K8B,K8CS2,COEF33,POINT)
            M = M - 1
            N = N + 1
            GOTO 3304
         ENDIF
 3303 CONTINUE

C******* WRITE IEO VECTORS TO ZMATRX

      DO 3305 I = 1, NSECS
         DO 3306 J = I, NSECS
            ZMATRX(I+NSECM,J+NSECM)
     &             = - .5D0 * ( IEOPL(I+J-1) - IEOMN(J-I+1) )
            ZMATRX(J+NSECM,I+NSECM) = ZMATRX(I+NSECM,J+NSECM)
 3306    CONTINUE
 3305 CONTINUE

C*************************************************************************
C     FOR THE HALF-SPACE IN THE CAVITY
C*************************************************************************
```

```
C*******************************************************************************
C       CALL KATEHI'S PROGRAM TO CALCULATE THE GREEN'S FUNCTION ON THE SLOT ON
C       THE HALF-SPACE SIDE AND MODIFY THE ELEMENTS IN THE Y55 MATRIX.
C*******************************************************************************

 9999 WRITE(*,*) 'CALCULATING EXTERNAL SLOT SELF-COUPLING SUB-MATRIX'

      CALL YIJDIE(YSD,DLX,WW,EER,NS1,NS2,NOFF,INS,INS1S2,
     &            ER,HH,TT,AA,AK0,AK,AKK,OFFSET)

      DO 5501 I=1,NSECS
         DO 5502 J=-I,NSECS-I-1
            M=J+I+1
            JJ=IABS(J)
            IF(J.GE.0) JJ=JJ+2
C            AMATRX(I,M) = AMATRX(I,M)+YSD(JJ)
            AMATRX(I,M) = YSD(JJ)
 5502    CONTINUE
 5501 CONTINUE

      DO 5503 I=1,NSECS
         DO 5504 J=1,NSECS
            ZMATRX(I+NSECM,J+NSECM) =ZMATRX(I+NSECM,J+NSECM)+AMATRX(I,J)
 5504    CONTINUE
 5503 CONTINUE

C*******************************************************************************

      RETURN
      END

C*******************************************************************************
C*******************************************************************************
C*******************************************************************************

      SUBROUTINE DEFINE(STRIP,SLOT,SL8OFF,
     &                  DLX,WW,EER,NS1,NS2,NOFF,INS,INS1S2,
     &                  ER,HH,TT,AA,AK0,AK,AKK,OFFSET,
     &                  A,B,C,H,L,L8U,X8O,Y8O,W,NSECM,
     &                  LS,L8S,S8O,T8O,WS,NSECS,
     &                  FREQ,MU8O,EPSCAV,EPSDIE,EPSLY1,K8C,K8B,
     &                  K8CS2,COEF11,COEF12,COEF33,POINT)

C*******************************************************************************
C********* PARAMETER DECLARATIONS

      REAL*8 PI
      PARAMETER(PI=3.14159265358979324D0)

C****** ELECTRICAL DECLARATIONS

      REAL*8 EPS8O,MU8O,OP8FRQ,FREQ,LAMBDA,LAMCAV
      REAL*8 ECAV1,ECAV2,EDIE1,EDIE2
      REAL*8 K8O,K8B,COEF11,COEF12,COEF33

      COMPLEX*16 EPSCAV,EPSDIE,EPSLY1,K8C,K8CS2

C****** geometry declarations
```

```
      INTEGER NSECM,NSECS

      REAL*8 A,B,C,H,AAA(3),BBB(3),CCC(3)
      REAL*8 X8O,Y8O,L,L8U,W,STRIP(1),Y8OOO(3),WWW(3)
      REAL*8 S8O,T8O,LS,L8S,WS,SLOT(1)
      REAL*8 S8OOO(3),WSSS(3)
      REAL*8 SLTOFF(3),SL8OFF
      REAL*8 TD,TDDD(3)

C****** MENU DECLARATIONS

      LOGICAL NORMAL
      INTEGER UNITS
C                                          1: normalized to material wavelength
C                                          2: inches
C                                          3: centimeters
C****** PROGRAM OPERATIONAL DECLARATIONS

      LOGICAL POINT

C***** FILE HANDLING DECLARATIONS

      LOGICAL EXISTS
      CHARACTER*80 WARN
      CHARACTER*10 CHECK
      CHARACTER*8  ANS
      CHARACTER*50 CONF8F,CFG8F

C***** LOOP DECLARATIONS

      INTEGER I,I1,I2

C******* FOR KATEHI'S PROGRAM ******************************

      INTEGER NS1,NS2,NOFF,INS,INS1S2
      REAL*8 DLX,WW,EER,ER,HH,TT,AA,AK0,AK,AKK,OFFSET(1)

C*************************************************************************
C      DEFINE WRITES A FILE CONTAINING ALL OF THE PARAMETERS NEEDED FOR
C      THE ANALYSIS OF THE CAVITY PROBLEM.  THE VARIABLES AND PARAMETERS ARE
C      NAMED AS FOLLOWS:
C
C      NOTE: ALL DISTANCES ARE NORMALIZED TO FREE SPACE WAVELENGTH IN THE PROGRAM.
C            IN THE ENTRY FILE THEY ARE ENTERED NORMALIZED TO THE HOST MATERIAL
C            WAVELENGTH
C
C      A                    -  DIMENSION OF CAVITY ALONG X AXIS
C      B                    -  DIMENSION OF CAVITY ALONG Y AXIS
C      C                    -  Z AXIS POSITION OF INSIDE OF CAVITY COVER
C      EPSCAV               -  RELATIVE DIELECTRIC CONSTANT OF CAVITY ABOVE
C                              THE MICROSTIP
C      EPSDIE               -  RELATIVE DIELECTRIC CONSTANT OF DIELECTRIC COVER
C      EPS8O                -  FREE SPACE PERMITTIVITY
C      FREQ                 -  FREQUENCY OF ANALYSIS IN HERTZ
C      K8B                  -  SINUSOIDAL BASIS FUNCTION SCALE FACTOR
C      K8C                  -  CAVITY WAVE NUMBER -- REGION ABOVE MICROSTRIP
C      K8O                  -  FREE SPACE WAVENUMBER
```

```
C      K8GC                    -  PROGAGATION CONSTANT OF CAVITY MODE
C      L8U                     -  LENGTH OF LONGITUDINAL SUBSECTION ON MICROSTRIP
C      LAMBDA8O                -  FREESPACE WAVELENGTH
C      MU8O                    -  FREE SPACE PERMEABILITY
C      NSECM                   -  NUMBER OF SUBSECTIONS ON MICROSTRIP
C      TD                      -  THICKNESS OF DIELECTRIC COVER
C      W                       -  WIDTH OF MICROSTRIP LINE IN FREE SPACE WAVELENGTHS
C      WS                      -  WIDTH OF SLOT LINE IN FREE SPACE WAVELENGTHS
C      S8O                     -  X AXIS COORDINATE OF SLOT CENTER
C      Y8O                     -  Y AXIS COORDINATE OF MICROSTRIP CENTER
C*******************************************************************************

C*******************************************************************************
C     THE NEXT SEGMENT DETERMINES WHETHER AN EXISTING CONFIGURATION FILE IS
C     TO BE USED
C*******************************************************************************
  100 WRITE(*,*) 'ENTER NAME OF CONFIGURATION DATA FILE; '
      READ(*,'(A50)') CONF8F
      CALL TRIM(CONF8F,50,I1,I2)
      CONF8F = CONF8F(I1:I2)
      CFG8F=CONF8F
      INQUIRE(FILE=CFG8F,EXIST=EXISTS)
      IF (EXISTS ) THEN
          CALL TRIM(CFG8F,50,I1,I2)
          WARN = '  WARNING: FILE '//CFG8F(I1:I2)//
     &           ' ALREADY EXISTS.  READ EXISTING FILE?'//
     &           ' ( <RETURN> = YES )'
          WRITE (6,*) WARN
          READ (5,'(A1)') ANS
          IF ((ANS.NE.'Y').AND.(ANS.NE.'y').AND.(ANS.NE.' ')) THEN
             WRITE(*,*) 'USE EXISTING FILE NAME FOR NEW CONFIGURATION?'
             READ(5,'(A1)') ANS
             IF((ANS.NE.'Y').AND.(ANS.NE.'y').AND.(ANS.NE.' ')) GOTO 100
             GOTO 950
          ENDIF
          NORMAL = .FALSE.
          UNITS  = 2
          GOTO 200
      ENDIF

C*******************************************************************************
C     NEXT SECTION OF CODE IS ONLY REACHED IF NO CONFIGURATION FILE ALREADY
C     EXISTS OR THE EXISTING NAMED CONFIGURATION FILE IS TO BE OVERWRITTEN.
C     WILL COLLECT ALL OF THE NECESSARY INPUT AND GENERATE A NEW CONFIGURATION.
C*******************************************************************************

  950 WRITE(*,*) 'ENTER FREQUENCY (GHZ): '
      READ(*,*,ERR=950) OP8FRQ
      LAMBDA = 2.997925D10/(OP8FRQ * 1.D9 * 2.54D0)

      WRITE(*,*) ' ENTRIES NORMALIZED TO MATERIAL WAVELENGTH? ',
     &           '( <RETURN> = YES ): '
      READ (5,'(A1)') ANS
      IF (ANS.NE.'Y') THEN

         NORMAL = .FALSE.
         WRITE(*,*) 'SELECT UNITS: '
         WRITE(*,*) '[1] INCHES '
```

```
         WRITE(*,*) '[2] CENTIMETERS '
         READ(*,*) UNITS
         IF (UNITS.GE.3) THEN
            NORMAL = .TRUE.
            UNITS = 1
         ELSE IF (UNITS.EQ.2) THEN
            WRITE(*,*) 'DIMENSIONS ARE TO BE ENTERED IN CENTIMETERS '
            UNITS = 3
         ELSE IF (UNITS.EQ.1) THEN
            WRITE(*,*) 'DIMENSIONS ARE TO BE ENTERED IN INCHES '
            UNITS = 2
         ENDIF

      ELSE

         NORMAL = .TRUE.
         UNITS = 1
         WRITE(*,*) 'DIMENSIONS ARE TO BE ENTERED NORMALIZED TO ',
     &              'MATERIAL WAVELENGTH '

      ENDIF

      ECAV1=1.D0
      ECAV2=0.D0
      EDIE1=1.D0
      EDIE2=0.D0

  951 WRITE(*,*) 'ENTER REAL PART OF CAVITY RELATIVE ',
     &           'DIELECTRIC CONSTANT : '
      READ(*,*,ERR=951) ECAV1
      IF(ECAV1.EQ.0.D0) ECAV1=1.D0

      EPSCAV=DCMPLX(ECAV1,ECAV2)
      EPSLY1 = EPSCAV
      LAMCAV = LAMBDA/DSQRT(ECAV1)

  953 WRITE(*,*) 'ENTER REAL PART OF COVER RELATIVE ',
     &           'DIELECTRIC CONSTANT : '
      READ(*,*,ERR=953) EDIE1
      IF(EDIE1.EQ.0.D0) EDIE1=1.D0

      EPSDIE=DCMPLX(EDIE1,EDIE2)

C*******************************************************************************
C    CAVITY GEOMETRY
C*******************************************************************************
  960 IF (UNITS.EQ.1) THEN
         WRITE(*,*) 'ENTER CAVITY DIMENSION ALONG MICROSTRIP (X): ',
     &              ' (NORMALIZED)'
         READ(*,*,ERR=960) AAA(1)
         AAA(2)          = AAA(1)      * LAMCAV
         AAA(3)          = AAA(2)      * 2.54D0
      ELSE IF (UNITS.EQ.2) THEN
         WRITE(*,*) 'ENTER WAVEGUIDE DIMENSION ALONG MICROSTRIP (X): ',
     &              ' (INCHES)'
         READ(*,*,ERR=960) AAA(2)
         AAA(1)          = AAA(2)      / LAMCAV
         AAA(3)          = AAA(2)      * 2.54D0
```

```
      ELSE IF (UNITS.EQ.3) THEN
         WRITE(*,*) 'ENTER WAVEGUIDE DIMENSION ALONG MICROSTRIP (X): ',
     &              ' (CM)'
         READ(*,*,ERR=960) AAA(3)
         AAA(2)          = AAA(3)       / 2.54D0
         AAA(1)          = AAA(2)       / LAMCAV
      ENDIF

  961 IF (UNITS.EQ.1) THEN
         WRITE(*,*) 'ENTER CAVITY DIMENSION ALONG SLOT (Y): ',
     &              ' (NORMALIZED)'
         READ(*,*,ERR=961) BBB(1)
         BBB(2)          = BBB(1)       * LAMCAV
         BBB(3)          = BBB(2)       * 2.54D0
      ELSE IF (UNITS.EQ.2) THEN
         WRITE(*,*) 'ENTER WAVEGUIDE DIMENSION ALONG SLOT (Y): (INCHES)'
         READ(*,*,ERR=961) BBB(2)
         BBB(1)          = BBB(2)       / LAMCAV
         BBB(3)          = BBB(2)       * 2.54D0
      ELSE IF (UNITS.EQ.3) THEN
         WRITE(*,*) 'ENTER WAVEGUIDE DIMENSION ALONG SLOT (Y):  (CM)'
         READ(*,*,ERR=961) BBB(3)
         BBB(2)          = BBB(3)       / 2.54D0
         BBB(1)          = BBB(2)       / LAMCAV
      ENDIF

  962 IF (UNITS.EQ.1) THEN
         WRITE(*,*) 'ENTER CAVITY HEIGHT DIMENSION (Z): (NORMALIZED)'
         READ(*,*,ERR=962) CCC(1)
         CCC(2)          = CCC(1)       * LAMCAV
         CCC(3)          = CCC(2)       * 2.54D0
      ELSE IF (UNITS.EQ.2) THEN
         WRITE(*,*) 'ENTER CAVITY HEIGHT DIMENSION (Z): (INCHES)'
         READ(*,*,ERR=962) CCC(2)
         CCC(1)          = CCC(2)       / LAMCAV
         CCC(3)          = CCC(2)       * 2.54D0
      ELSE IF (UNITS.EQ.3) THEN
         WRITE(*,*) 'ENTER HEIGHT DIMENSION (Z): (CM)'
         READ(*,*,ERR=962) CCC(3)
         CCC(2)          = CCC(3)       / 2.54D0
         CCC(1)          = CCC(2)       / LAMCAV
      ENDIF

C*******************************************************************************
C     STRIP GEOMETRY
C*******************************************************************************

  965 IF (UNITS.EQ.1) THEN
         WRITE(*,*) 'ENTER WIDTH OF MICROSTRIP LINE: (NORMALIZED)'
         READ(*,*,ERR=965) WWW(1)
         WWW(2)          = WWW(1)       * LAMCAV
         WWW(3)          = WWW(2)       * 2.54D0
      ELSE IF (UNITS.EQ.2) THEN
         WRITE(*,*) 'ENTER WIDTH OF MICROSTRIP LINE: (INCHES)'
         READ(*,*,ERR=965) WWW(2)
         WWW(1)          = WWW(2)       / LAMCAV
         WWW(3)          = WWW(2)       * 2.54D0
      ELSE IF (UNITS.EQ.3) THEN
```

```
         WRITE(*,*) 'ENTER WIDTH OF MICROSTRIP LINE: (CM)'
         READ(*,*,ERR=965) WWW(3)
         WWW(2)           = WWW(3)       / 2.54D0
         WWW(1)           = WWW(2)       / LAMCAV
      ENDIF

  968 IF (UNITS.EQ.1) THEN
         WRITE(*,*) 'ENTER Y CENTER OF MICROSTRIP LINE ',
     &              '(NORMALIZED TO B): '
         READ(*,*,ERR=968) Y8000(1)
         Y8000(2) = Y8000(1) * BBB(2)
         Y8000(3) = Y8000(2) * 2.54D0
      ELSE IF (UNITS.EQ.2) THEN
         WRITE(*,*) 'ENTER Y CENTER OF MICROSTRIP LINE ',
     &              '(INCHES): '
         READ(*,*,ERR=968) Y8000(2)
         Y8000(1) = Y8000(2) / BBB(2)
         Y8000(3) = Y8000(2) * 2.54D0
      ELSE IF (UNITS.EQ.3) THEN
         WRITE(*,*) 'ENTER Y CENTER OF MICROSTRIP LINE ',
     &              '(CM): '
         READ(*,*,ERR=968) Y8000(3)
         Y8000(2) = Y8000(3) / 2.54D0
         Y8000(1) = Y8000(2) / BBB(2)
      ENDIF

C*************************************************************************************
C     SLOT GEOMETRY
C*************************************************************************************

  972 IF (UNITS.EQ.1) THEN
         WRITE(*,*) 'ENTER WIDTH OF SLOT: (NORMALIZED)'
         READ(*,*,ERR=972) WSSS(1)
         WSSS(2)          = WSSS(1)   * LAMCAV
         WSSS(3)          = WSSS(2)   * 2.54D0
      ELSE IF (UNITS.EQ.2) THEN
         WRITE(*,*) 'ENTER WIDTH OF SLOT: (INCHES)'
         READ(*,*,ERR=972) WSSS(2)
         WSSS(1)          = WSSS(2)   / LAMCAV
         WSSS(3)          = WSSS(2)   * 2.54D0
      ELSE IF (UNITS.EQ.3) THEN
         WRITE(*,*) 'ENTER WIDTH OF SLOT: (CM)'
         READ(*,*,ERR=972) WSSS(3)
         WSSS(2)          = WSSS(3)   / 2.54D0
         WSSS(1)          = WSSS(2)   / LAMCAV
      ENDIF

  974 IF (UNITS.EQ.1) THEN
         WRITE(*,*) 'ENTER X CENTER OF SLOT ',
     &              '(NORMALIZED TO A): '
         READ(*,*,ERR=974) S8000(1)
         S8000(2)         = S8000(1)     * AAA(2)
         S8000(3)         = S8000(2)     * 2.54D0
      ELSE IF (UNITS.EQ.2) THEN
         WRITE(*,*) 'ENTER X CENTER OF SLOT (IN INCHES): '
         READ(*,*,ERR=974) S8000(2)
         S8000(1)         = S8000(2)     / AAA(2)
         S8000(3)         = S8000(2)     * 2.54D0
```

```
      ELSE IF (UNITS.EQ.3) THEN
         WRITE(*,*) 'ENTER X CENTER OF SLOT (IN CM): '
         READ(*,*,ERR=974) S8000(3)
         S8000(2)         = S8000(3)    / 2.54D0
         S8000(1)         = S8000(2)    / AAA(2)
      ENDIF

       WRITE(*,*) 'Slot offset is defined as the distance '
       WRITE(*,*) 'from the nearest end of the slot to the '
       WRITE(*,*) 'center of the microstrip line.'
       WRITE(*,*) 'Defined as positive is center is under slot.'
       WRITE(*,*)
  975  IF (UNITS.EQ.1) THEN
         WRITE(*,*) 'ENTER SLOT OFFSET (NORMALIZED TO B) '
         READ(*,*,ERR=975) SLTOFF(1)
         SLTOFF(2) = SLTOFF(1) * BBB(2)
         SLTOFF(3) = SLTOFF(2) * 2.54D0
       ELSE IF (UNITS.EQ.2) THEN
         WRITE(*,*) 'ENTER SLOT OFFSET (INCHES)'
         READ(*,*,ERR=975) SLTOFF(2)
         SLTOFF(1) = SLTOFF(2) / BBB(2)
         SLTOFF(3) = SLTOFF(2) * 2.54D0
       ELSE IF (UNITS.EQ.3) THEN
         WRITE(*,*) 'ENTER SLOT OFFSET (CM)'
         READ(*,*,ERR=975) SLTOFF(3)
         SLTOFF(2) = SLTOFF(3) / 2.54D0
         SLTOFF(1) = SLTOFF(2) / BBB(2)
       ENDIF

C*******************************************************************************
C     COVER GEOMETRY
C*******************************************************************************

  977 IF (UNITS.EQ.1) THEN
         WRITE(*,*) 'ENTER THICKNESS OF COVER: (NORMALIZED)'
         READ(*,*,ERR=977) TDDD(1)
         TDDD(2)          = TDDD(1)   * LAMCAV * DSQRT(ECAV1/EDIE1)
         TDDD(3)          = TDDD(2)   * 2.54D0
      ELSE IF (UNITS.EQ.2) THEN
         WRITE(*,*) 'ENTER THICKNESS OF COVER: (INCHES)'
         READ(*,*,ERR=977) TDDD(2)
         TDDD(1)          = TDDD(2)   / (LAMCAV * DSQRT(ECAV1/EDIE1))
         TDDD(3)          = TDDD(2)   * 2.54D0
      ELSE IF (UNITS.EQ.3) THEN
         WRITE(*,*) 'ENTER THICKNESS OF COVER: (CM)'
         READ(*,*,ERR=977) TDDD(3)
         TDDD(2)          = TDDD(3)   / 2.54D0
         TDDD(1)          = TDDD(2)   / (LAMCAV * DSQRT(ECAV1/EDIE1))
      ENDIF

      GOTO 400

C*******************************************************************************
C     THE NEXT SEGMENT READS DATA FROM EXISTING FILE
C*******************************************************************************

  200 OPEN(10,FILE=CFG8F)
```

```
      READ(10,'(A50)') CONF8F
      READ(10,*) OP8FRQ
      READ(10,*) ECAV1,ECAV2
      READ(10,*) EDIE1,EDIE2
      READ(10,*) AAA(1)
      READ(10,*) BBB(1)
      READ(10,*) CCC(1)
      READ(10,*) WWW(1)
      READ(10,*) Y8OOO(1)
      READ(10,*) WSSS(1)
      READ(10,*) S8OOO(1)
      READ(10,*) SLTOFF(1)
      READ(10,*) TDDD(1)
      CLOSE(10)

      EPSCAV = DCMPLX(ECAV1,ECAV2)
      EPSLY1 = EPSCAV
      EPSDIE = DCMPLX(EDIE1,EDIE2)

C*******************************************************************************
C     UNNORMALIZATION OF READ DATA TO INCHES
C*******************************************************************************

      LAMBDA = 2.997925D10/(OP8FRQ * 1.D9 * 2.54D0)
      LAMCAV = LAMBDA/DSQRT(ECAV1)

      AAA(2)         = AAA(1)       * LAMCAV
      BBB(2)         = BBB(1)       * LAMCAV
      CCC(2)         = CCC(1)       * LAMCAV
      WWW(2)         = WWW(1)       * LAMCAV
      Y8OOO(2)       = Y8OOO(1)     * BBB(2)
      WSSS(2)        = WSSS(1)      * LAMCAV
      S8OOO(2)       = S8OOO(1)     * AAA(2)
      SLTOFF(2)      = SLTOFF(1)    * BBB(2)
      TDDD(2)        = TDDD(1)      * LAMCAV * DSQRT(ECAV1/EDIE1)

      AAA(3)         = AAA(2)       * 2.54D0
      BBB(3)         = BBB(2)       * 2.54D0
      CCC(3)         = CCC(2)       * 2.54D0
      WWW(3)         = WWW(2)       * 2.54D0
      Y8OOO(3)       = Y8OOO(2)     * 2.54D0
      WSSS(3)        = WSSS(2)      * 2.54D0
      S8OOO(3)       = S8OOO(2)     * 2.54D0
      SLTOFF(3)      = SLTOFF(2)    * 2.54D0
      TDDD(3)        = TDDD(2)      * 2.54D0

C*******************************************************************************
C     NORMALIZATION TO HOST WAVELENGTH OR UNNORMALIZATION TO INCHES
C*******************************************************************************

  400 LAMBDA = 2.997925D10 / (OP8FRQ * 1.D9 * 2.54D0)
      LAMCAV = LAMBDA / DSQRT(ECAV1)

C*******************************************************************************
C   THE NEXT PART IS THE MAIN PART OF THE MENU SYSTEM
C*******************************************************************************

C*******************************************************************************
```

```
C         WRITE VARIABLE LIST ON SCREEN
C*******************************************************************************

         WRITE(*,*)

         WRITE(*,749) CONF8F
         WRITE(*,*)
         WRITE(*,750) ECAV1,ECAV2
         WRITE(*,752) EDIE1,EDIE2
         WRITE(*,*)
         WRITE(*,756) NORMAL
         WRITE(*,757) OP8FRQ              , LAMBDA
         WRITE(*,758) AAA(1)              , AAA(2)
         WRITE(*,759) BBB(1)              , BBB(2)
         WRITE(*,760) CCC(1)              , CCC(2)
         WRITE(*,763) WWW(1)              , WWW(2)
         WRITE(*,766) Y8OOO(1)            , Y8OOO(2)
         WRITE(*,770) WSSS(1)             , WSSS(2)
         WRITE(*,772) S8OOO(1)            , S8OOO(2)
         WRITE(*,776) SLTOFF(1)           , SLTOFF(2)
         WRITE(*,775) TDDD(1)             , TDDD(2)
         WRITE(*,*)

  749 FORMAT('Name of geometry configuration file     [NAME] ', A50)
  750 FORMAT('Cavity rel. dielectric constant; REAL [ECR]:',
     & F14.7,2X,'IMAG [ECI]:',F14.7)
  752 FORMAT('Cover rel. dielectric constant;  REAL [EDR]:',
     & F14.7,2X,'IMAG [EDI]:',F14.7)
  756 FORMAT('DIMENSIONS',/,
     &       'ENTRY NORMALIZED TO HOST WAVELENGTH       [NL] ',L1,
     &                        3X,'normalized',4X,'(inches) ',/,
     &                       51X,'----------',4X,'-------- ')
  757 FORMAT('Frequency (GHz) of operation  (lambda_o)  [F]  ',
     & F14.7, 2X,'(',F10.7,')')
  758 FORMAT('cavity dimension along microstrip (x)     [A]  ',
     & F14.7, 2X,'(',F10.7,')')
  759 FORMAT('cavity dimension along slot (y)           [B]  ',
     & F14.7, 2X,'(',F10.7,')')
  760 FORMAT('cavity height dimension (z)               [C]  ',
     & F14.7, 2X,'(',F10.7,')')
  763 FORMAT('width of microstrip line                  [W]  ',
     & F14.7, 2X,'(',F10.7,')')
  766 FORMAT('y center of microstrip (normalized to b)  [Y0] ',
     & F14.7, 2X,'(',F10.7,')')
  770 FORMAT('width of slot                             [WS] ',
     & F14.7, 2X,'(',F10.7,')')
  772 FORMAT('x center of slot (normalized to a)        [S0] ',
     & F14.7, 2X,'(',F10.7,')')
  776 FORMAT('slot offset      (normalized to b)        [OF] ',
     & F14.7, 2X,'(',F10.7,')')
  775 FORMAT('thickness of cover                        [TD] ',
     & F14.7, 2X,'(',F10.7,')')

C*******************************************************************************
C*******************************************************************************
C*******************************************************************************
C     THE NEXT PART READS THE RESPONSES TO THE MENU PROMPT
C     DETECTS WHICH VARIABLE IS TO BE CHANGED
```

```
C     PROMPTS FOR THE CHANGE AND MAKES IT
C     AND UPDATES THE MENU
C*******************************************************************************

      WRITE(*,*)
      WRITE(*,*) 'ENTER VARIABLE NAME [**] OR <RETURN>'

      READ(*,'(A4)',ERR=499) CHECK

C*******  ALTER VARIABLES AS DESIRED

      IF (CHECK.EQ.' ') GOTO 600

      IF (    (CHECK.EQ.'STOP') .OR.(CHECK.EQ.'stop')
     &    .OR.(CHECK.EQ.'QUIT') .OR.(CHECK.EQ.'quit')
     &    .OR.(CHECK.EQ.'EXIT') .OR.(CHECK.EQ.'exit')
     &    .OR.(CHECK.EQ.'ABORT').OR.(CHECK.EQ.'abort') ) STOP

      IF ((CHECK.EQ.'ECR').OR.(CHECK.EQ.'ecr')) THEN
        WRITE(*,*) 'ENTER REAL PART OF CAVITY DIELECTRIC'
        READ(*,*,ERR=499) ECAV1
        EPSCAV = DCMPLX(ECAV1,ECAV2)
        EPSLY1 = EPSCAV
        GOTO 400
      ELSE IF ((CHECK.EQ.'ECI').OR.(CHECK.EQ.'eci')) THEN
        WRITE(*,*) 'ENTER IMAG PART OF CAVITY DIELECTRIC'
        READ(*,*,ERR=499) ECAV2
        EPSCAV = DCMPLX(ECAV1,ECAV2)
        EPSLY1 = EPSCAV
        GOTO 400
      ENDIF

      IF ((CHECK.EQ.'EDR').OR.(CHECK.EQ.'edr')) THEN
        WRITE(*,*) 'ENTER REAL PART OF COVER DIELECTRIC'
        READ(*,*,ERR=499) EDIE1
        EPSDIE=DCMPLX(EDIE1,EDIE2)
        GOTO 400
      ELSE IF ((CHECK.EQ.'EDI').OR.(CHECK.EQ.'edi')) THEN
        WRITE(*,*) 'ENTER IMAG PART OF COVER DIELECTRIC'
        READ(*,*,ERR=499) EDIE2
        EPSDIE=DCMPLX(EDIE1,EDIE2)
        GOTO 400
      ENDIF

      IF (    (CHECK.EQ.'NL').OR.(CHECK.EQ.'CU')
     &    .OR.(CHECK.EQ.'nl').OR.(CHECK.EQ.'cu') ) THEN
          WRITE(*,*) 'SELECT UNITS: '
          WRITE(*,*) '[1] NORMALIZED TO MATERIAL WAVELENGTH '
          WRITE(*,*) '[2] INCHES '
          WRITE(*,*) '[3] CENTIMETERS '
          READ(*,*) UNITS
        GOTO 400
      ENDIF

      IF ( (CHECK.EQ.'F').OR.(CHECK.EQ.'f') ) THEN
        WRITE(*,*) 'ENTER NEW OPERATING FREQUENCY'
        READ(*,*,ERR=499) OP8FRQ
        GOTO 400
```

```
      ENDIF

      IF ( (CHECK.EQ.'A').OR.(CHECK.EQ.'a') ) THEN
        IF (UNITS.EQ.1) THEN
           WRITE(*,*) 'ENTER CAVITY DIMENSION ALONG MICROSTRIP (X): ',
     &                ' (NORMALIZED)'
           READ(*,*,ERR=499) AAA(1)
           AAA(2)          = AAA(1)      * LAMCAV
           AAA(3)          = AAA(2)      * 2.54D0
        ELSE IF (UNITS.EQ.2) THEN
           WRITE(*,*) 'ENTER CAVITY DIMENSION ALONG MICROSTRIP (X): ',
     &                '(INCHES)'
           READ(*,*,ERR=499) AAA(2)
           AAA(1)          = AAA(2)      / LAMCAV
           AAA(3)          = AAA(2)      * 2.54D0
        ELSE IF (UNITS.EQ.3) THEN
           WRITE(*,*) 'ENTER CAVITY DIMENSION ALONG MICROSTRIP (X): ',
     &                ' (CM)'
           READ(*,*,ERR=499) AAA(3)
           AAA(2)          = AAA(3)       / 2.54D0
           AAA(1)          = AAA(2)       / LAMCAV
        ENDIF
        S8000(1)       = S8000(2)     / AAA(2)
        GOTO 400
      ENDIF

      IF ( (CHECK.EQ.'B').OR.(CHECK.EQ.'b') ) THEN
        IF (UNITS.EQ.1) THEN
            WRITE(*,*) 'ENTER CAVITY DIMENSION ALONG SLOT (Y): ',
     &                 ' (NORMALIZED)'
            READ(*,*,ERR=499) BBB(1)
            BBB(2)          = BBB(1)      * LAMCAV
            BBB(3)          = BBB(2)      * 2.54D0
        ELSE IF (UNITS.EQ.2) THEN
            WRITE(*,*) 'ENTER CAVITY DIMENSION ALONG SLOT (Y): (INCHES)'
            READ(*,*,ERR=499) BBB(2)
            BBB(1)          = BBB(2)      / LAMCAV
            BBB(3)          = BBB(2)      * 2.54D0
        ELSE IF (UNITS.EQ.3) THEN
           WRITE(*,*) 'ENTER CAVITY DIMENSION ALONG SLOT (Y): (CM)'
           READ(*,*,ERR=499) BBB(3)
           BBB(2)          = BBB(3)       / 2.54D0
           BBB(1)          = BBB(2)       / LAMCAV
        ENDIF
        Y8000(1)       = Y8000(2)     / BBB(2)
        GOTO 400
      ENDIF

      IF ( (CHECK.EQ.'C').OR.(CHECK.EQ.'c') ) THEN
        IF (UNITS.EQ.1) THEN
           WRITE(*,*) 'ENTER CAVITY HEIGHT DIMENSION (Z): (NORMALIZED)'
           READ(*,*,ERR=499) CCC(1)
           CCC(2)          = CCC(1)      * LAMCAV
           CCC(3)          = CCC(2)      * 2.54D0
        ELSE IF (UNITS.EQ.2) THEN
           WRITE(*,*) 'ENTER CAVITY HEIGHT DIMENSION (Z): (INCHES)'
           READ(*,*,ERR=499) CCC(2)
           CCC(1)          = CCC(2)      / LAMCAV
```

```
         CCC(3)          = CCC(2)      * 2.54D0
       ELSE IF (UNITS.EQ.3) THEN
         WRITE(*,*) 'ENTER CAVITY HEIGHT DIMENSION (Z): (CM)'
         READ(*,*,ERR=499) CCC(3)
         CCC(2)          = CCC(3)      / 2.54D0
         CCC(1)          = CCC(2)      / LAMCAV
       ENDIF
       GOTO 400
     ENDIF

     IF ( (CHECK.EQ.'W').OR.(CHECK.EQ.'w') ) THEN
       IF (UNITS.EQ.1) THEN
         WRITE(*,*) 'ENTER MICROSTRIP WIDTH: (NORMALIZED)'
         READ(*,*,ERR=499) WWW(1)
         WWW(2)          = WWW(1)      * LAMCAV
         WWW(3)          = WWW(2)      * 2.54D0
       ELSE IF (UNITS.EQ.2) THEN
         WRITE(*,*) 'ENTER MICROSTRIP WIDTH: (INCHES)'
         READ(*,*,ERR=499) WWW(2)
         WWW(1)          = WWW(2)      / LAMCAV
         WWW(3)          = WWW(2)      * 2.54D0
       ELSE IF (UNITS.EQ.3) THEN
         WRITE(*,*) 'ENTER MICROSTRIP WIDTH: (CM)'
         READ(*,*,ERR=499) WWW(3)
         WWW(2)          = WWW(3)      / 2.54D0
         WWW(1)          = WWW(2)      / LAMCAV
       ENDIF
       GOTO 400
     ENDIF

     IF ( (CHECK.EQ.'Y0').OR.(CHECK.EQ.'y0') ) THEN
       IF (UNITS.EQ.1) THEN
         WRITE(*,*) 'ENTER MICROSTRIP Y CENTER (NORMALIZED TO B): '
         READ(*,*,ERR=499) Y8OOO(1)
         Y8OOO(2) = Y8OOO(1) * BBB(2)
         Y8OOO(3) = Y8OOO(2) * 2.54D0
       ELSE IF (UNITS.EQ.2) THEN
         WRITE(*,*) 'ENTER MICROSTRIP Y CENTER  (INCHES)'
         READ(*,*,ERR=499) Y8OOO(2)
         Y8OOO(1) = Y8OOO(2) / BBB(2)
         Y8OOO(3) = Y8OOO(2) * 2.54D0
       ELSE IF (UNITS.EQ.3) THEN
         WRITE(*,*) 'ENTER MICROSTRIP Y CENTER  (CM)'
         READ(*,*,ERR=499) Y8OOO(3)
         Y8OOO(2) = Y8OOO(3) / 2.54D0
         Y8OOO(1) = Y8OOO(2) / BBB(2)
       ENDIF
       GOTO 400
     ENDIF

     IF ( (CHECK.EQ.'WS').OR.(CHECK.EQ.'ws') ) THEN
       IF (UNITS.EQ.1) THEN
         WRITE(*,*) 'ENTER SLOT WIDTH (NORMALIZED)'
         READ(*,*,ERR=499) WSSS(1)
         WSSS(2)         = WSSS(1)   * LAMCAV
         WSSS(3)         = WSSS(2)   * 2.54D0
       ELSE IF (UNITS.EQ.2) THEN
         WRITE(*,*) 'ENTER SLOT WIDTH (INCHES)'
```

```
          READ(*,*,ERR=499) WSSS(2)
          WSSS(1)          = WSSS(2)  / LAMCAV
          WSSS(3)          = WSSS(2)  * 2.54D0
        ELSE IF (UNITS.EQ.3) THEN
          WRITE(*,*) 'ENTER SLOT WIDTH (CM)'
          READ(*,*,ERR=499) WSSS(3)
          WSSS(2)          = WSSS(3)  / 2.54D0
          WSSS(1)          = WSSS(2)  / LAMCAV
        ENDIF
        GOTO 400
      ENDIF

      IF ( (CHECK.EQ.'S0').OR.(CHECK.EQ.'s0') ) THEN
        IF (UNITS.EQ.1) THEN
          WRITE(*,*) 'ENTER SLOT X CENTER (NORMALIZED TO A)'
          READ(*,*,ERR=499) S8000(1)
          S8000(2)         = S8000(1)    * AAA(2)
          S8000(3)         = S8000(2)    * 2.54D0
       ELSE IF (UNITS.EQ.2) THEN
          WRITE(*,*) 'ENTER SLOT X CENTER (INCHES)'
          READ(*,*,ERR=499) S8000(2)
          S8000(1)         = S8000(2)    / AAA(2)
          S8000(3)         = S8000(2)    * 2.54D0
        ELSE IF (UNITS.EQ.3) THEN
          WRITE(*,*) 'ENTER SLOT X CENTER (CM)'
          READ(*,*,ERR=499) S8000(3)
          S8000(2)         = S8000(3)    / 2.54D0
          S8000(1)         = S8000(2)    / AAA(2)
        ENDIF
        GOTO 400
      ENDIF

      IF ((CHECK.EQ.'OF').OR.(CHECK.EQ.'of')) THEN
        WRITE(*,*) 'Slot offset is defined as the distance '
        WRITE(*,*) 'from the nearest end of the slot to the '
        WRITE(*,*) 'center of the microstrip line.'
        WRITE(*,*) 'Defined as positive is center is under slot.'
        WRITE(*,*)
        IF (UNITS.EQ.1) THEN
          WRITE(*,*) 'ENTER SLOT OFFSET (NORMALIZED TO B) '
          READ(*,*,ERR=499) SLTOFF(1)
          SLTOFF(2) = SLTOFF(1) * BBB(2)
          SLTOFF(3) = SLTOFF(2) * 2.54D0
        ELSE IF (UNITS.EQ.2) THEN
          WRITE(*,*) 'ENTER SLOT OFFSET (INCHES)'
          READ(*,*,ERR=499) SLTOFF(2)
          SLTOFF(1) = SLTOFF(2) / BBB(2)
          SLTOFF(3) = SLTOFF(2) * 2.54D0
        ELSE IF (UNITS.EQ.3) THEN
          WRITE(*,*) 'ENTER SLOT OFFSET (CM)'
          READ(*,*,ERR=499) SLTOFF(3)
          SLTOFF(2) = SLTOFF(3) / 2.54D0
          SLTOFF(1) = SLTOFF(2) / BBB(2)
        ENDIF
        GOTO 400
      ENDIF

      IF ( (CHECK.EQ.'TD').OR.(CHECK.EQ.'td') ) THEN
```

```
         IF (UNITS.EQ.1) THEN
            WRITE(*,*) 'ENTER COVER THICKNESS (NORMALIZED)'
            READ(*,*,ERR=499) TDDD(1)
            TDDD(2)          = TDDD(1)  * LAMCAV * DSQRT(ECAV1/EDIE1)
            TDDD(3)          = TDDD(2)  * 2.54D0
         ELSE IF (UNITS.EQ.2) THEN
            WRITE(*,*) 'ENTER COVER THICKNESS (INCHES)'
            READ(*,*,ERR=499) TDDD(2)
            TDDD(1)          = TDDD(2)  / (LAMCAV * DSQRT(ECAV1/EDIE1))
            TDDD(3)          = TDDD(2)  * 2.54D0
         ELSE IF (UNITS.EQ.3) THEN
            WRITE(*,*) 'ENTER COVER THICKNESS (CM)'
            READ(*,*,ERR=499) TDDD(3)
            TDDD(2)          = TDDD(3)  / 2.54D0
            TDDD(1)          = TDDD(2)  / (LAMCAV * DSQRT(ECAV1/EDIE1))
         ENDIF
        GOTO 400
      ENDIF

  499 WRITE(*,'( )')
      WRITE(*,*) '*********************************************'
      WRITE(*,*) '      ENTRY FORMAT ERROR -- TRY AGAIN         '
      WRITE(*,*) '*********************************************'
      WRITE(*,'( )')
      GOTO 400


C*****************************************************************************
C     THE NEXT PART IS THE LAUNCHING POINT FOR THE REST OF THE PROGRAM
C     IT IS ONLY REACHED WHEN FINISHED WITH THE MENU PART
C*****************************************************************************

C*****************************************************************************
C               RENORMALIZE TO FREE SPACE WAVELENGTH FOR PROGRAM
C*****************************************************************************

  600 EPS8O = 8.854185D-12 * (2.54D0 * LAMBDA /100.D0 )
      MU8O = 4.0D-7 * PI   * (2.54D0 * LAMBDA /100.D0 )

      A = AAA(2) /LAMBDA
      B = BBB(2) /LAMBDA
      C = CCC(2) /LAMBDA

      H = C / 2.D0

      L  = A
      TD = TDDD(2) /LAMBDA
      W  = WWW(2)  /LAMBDA
      WS = WSSS(2) /LAMBDA

      FREQ   = OP8FRQ * 1.0D9

      K8O = 2.D0 * PI
      K8C = K8O  * CDSQRT(EPSCAV)
      K8B = DREAL(K8C)

      K8CS2  = K8C * K8C
      COEF11 = - PI * FREQ * MU8O * 8.D0/( A * B )
```

```
      COEF12 = - 4.D0 /(A*B)
      COEF33 = - PI * FREQ * EPS8O * 8.D0 / ( A * B )

      X8O = A / 2.D0
      Y8O = B * Y8OOO(1)

      NSECM = NINT( AAA(1) * 35.D0 )
      L8U = L/FLOAT(NSECM+1)

C CALCULATE A FULL SUBSECTION ON EACH END OF THE STRIP AS WELL WHICH CAN BE LATER
C REMOVED OR USED AS A HALF SINUSOID SUBSECTION FOR ANALYSIS AS A ONE OR TWO PORT
C NETWORK (IN ANOTHER PROGRAM WILL DIVIDE BY IMMITTANCE ELEMENT BY TWO FOR HALF-
C SINE BASIS FUNCTION ).

      NSECM = NSECM + 2
      DO 1100 I=1,NSECM
         STRIP(I) = FLOAT(I-1) * L8U - L/2.D0
 1100 CONTINUE

      S8O = A * S8OOO(1)
      SL8OFF = SLTOFF(1) * B

      POINT = .TRUE.
      EPSLY1 = EPSCAV

C*******************************************************************************
C      KATEHI'S PROGRAM PARAMETERS

        TT    = 1.0D-5
        ER    = DREAL(EPSDIE)
        EER   = ER
        HH    = TD * DSQRT(EER)
        WW    = WS     * DSQRT(EER)

        AA    = 100.0D0
        AK0   = 2.0D0*PI / DSQRT(EER)
        AK    = AK0      * DSQRT(ER)
        AKK   = 2.0D0*PI

        OFFSET(7)= 0.0D0
        NS2   = 0
        NOFF  = 1

        INS   = 1
        INS1S2= 2

C*******************************************************************************
C*******  WRITE VARIABLES TO NEW FILE OR OLD FILE


      CALL TRIM(CFG8F,50,I1,I2)
      CFG8F = CFG8F(I1:I2)
      INQUIRE(FILE=CFG8F(I1:I2),EXIST=EXISTS)
      IF ( EXISTS ) THEN
         WRITE(*,*) 'FILE ALREADY EXISTS --- '
         WRITE(*,*) 'Rename [R] to preserve existing ''?*.cfg'' file.'
         WRITE(*,*) '<RETURN> : runs new configuration ',
     &              'but doesn''t save in ''?*.cfg'' file.'
```

```
         WRITE(*,*) '[OVERRIDE] writes over existing ''?*.cfg'' file.'
         READ(*,'(A8)') ANS
         IF (ANS.NE.' ') THEN
            IF ((ANS.EQ.'OVERRIDE').OR.(ANS.EQ.'override')) THEN
               GOTO 599
            ENDIF
         ENDIF
      ELSE
  599    OPEN(10,FILE=CFG8F)
         CALL WROUT(10,STRIP,SLOT,SLTOFF,
     &              A,B,C,Y8O,W,S8O,WS,TD,FREQ,EPSCAV,EPSDIE,CONF8F)
         CLOSE(10)
      ENDIF

C*******************************************************************************

      RETURN
      END

C*******************************************************************************
C*******************************************************************************
C*******************************************************************************
      SUBROUTINE TRIM(CHAR,LEN,I1,I2)
C*************************************************************************C
C                                                                         C
C   FINDS POSITIONS OF THE FIRST AND LAST NON-BLANK CHARACTERS            C
C   IN A STRING                                                           C
C                                                                         C
C*************************************************************************C
      INTEGER LEN,I1,I2,I
      CHARACTER*(*) CHAR
      DO 10 I=1,LEN
        IF (CHAR(I:I).NE.' ') THEN
          I1=I
          GOTO 15
        ENDIF
  10  CONTINUE
  15  DO 20 I=LEN,1,-1
        IF (CHAR(I:I).NE.' ') THEN
          I2=I
          GOTO 25
        ENDIF
  20  CONTINUE
  25  RETURN
      END

C*******************************************************************************
C*******************************************************************************
C*******************************************************************************
C     SUBROUTINE WROUT WRITES ALL OF THE PARAMETERS FOR THE PROBLEM IN
C     THE FILE OPENED AS FILE = IFILE IN THE CALLING PROGRAM.
C
C     WHEN WRITEOUT IS CALLED VARIABLES ARE ASSUMED NORMALIZED TO FREE SPACE
C     WAVELENGTH. WROUT RENORMALIZES TO WAVELENGTHS IN THE HOST MEDIA BEFORE
C     WRITING OUT TO THE DATA FILE.
C*******************************************************************************

      SUBROUTINE WROUT(IFILE,STRIP,SLOT,SLTOFF,
```

```
     &                A,B,C,Y8O,W,S8O,WS,TD,FREQ,EPSCAV,EPSDIE,CONF8F)

C*******************************************************************************

      INTEGER IFILE

      REAL*8 FREQ,LAMBDA,LAM8CM,ECAV1,ECAV2,EDIE1,EDIE2
      REAL*8 A,B,C,Y8O,W,STRIP(1),S8O,WS,SLOT(1),SLTOFF,TD

      COMPLEX*16 EPSCAV,EPSDIE

      CHARACTER*50 CONF8F

C*******************************************************************************
      LAMBDA = 2.997925D10/(FREQ * 2.54D0)
      LAM8CM = LAMBDA * 2.54D0

      ECAV1 = DREAL(EPSCAV)
      ECAV2 = DIMAG(EPSCAV)
      EDIE1 = DREAL(EPSDIE)
      EDIE2 = DIMAG(EPSDIE)

      WRITE(IFILE,*) CONF8F,'  Configuration file name'
      WRITE(IFILE,*) FREQ/1.D9,
     &'           Frequency (GHz) ------> lambda_o = ',LAMBDA,' inches',
     &                                                 LAM8CM, ' cm'
      WRITE(IFILE,*) ECAV1,ECAV2,
     &'           Cavity REAL and IMAG Dielectric Constant'
      WRITE(IFILE,*) EDIE1,EDIE2,
     &'           Cover REAL and IMAG Dielectric Constant'
      WRITE(IFILE,*) A * DSQRT(ECAV1),
     &'           Cavity x dimension ------> ',    A * LAMBDA,' inches',
     &                                             A * LAM8CM, ' cm'
      WRITE(IFILE,*) B * DSQRT(ECAV1),
     &'           Cavity y dimension ------> ',    B * LAMBDA,' inches',
     &                                             B * LAM8CM, ' cm'
      WRITE(IFILE,*) C * DSQRT(ECAV1),
     &'           Cavity z dimension ------> ',    C * LAMBDA,' inches',
     &                                             C * LAM8CM, ' cm'
      WRITE(IFILE,*) W * DSQRT(ECAV1),
     &'           Microstrip line width ------> ', W * LAMBDA,' inches',
     &                                             W * LAM8CM, ' cm'
      WRITE(IFILE,*) Y8O / B,
     &'           Microstrip y offset   ------> ',Y8O *LAMBDA,' inches',
     &                                            Y8O *LAM8CM, ' cm'
      WRITE(IFILE,*) WS * DSQRT(ECAV1),
     &'           Slot width ------> ',           WS * LAMBDA,' inches',
     &                                            WS * LAM8CM, ' cm'
      WRITE(IFILE,*) S8O / A,
     &'           Slot x center ------> ',       S8O * LAMBDA,' inches',
     &                                           S8O * LAM8CM, ' cm'
      WRITE(IFILE,*) SLTOFF,
     &'           Slot offset -----> ',   SLTOFF * B * LAMBDA,' inches',
     &                                    SLTOFF * B * LAM8CM, ' cm'
      WRITE(IFILE,*) TD * DSQRT(EDIE1),
     &'           Cover thickness ------> ',      TD * LAMBDA,' inches',
     &                                            TD * LAM8CM, ' cm'
```

```
      RETURN
      END


C*********************************************************************C
C*********************************************************************C
C*********************************************************************C
C                                                                     C
      SUBROUTINE HANKZ1(R,N,HZERO,HONE)
C                                                                     C
C*********************************************************************C
C                                                                     C
C     CALLED BY MAIN PROGRAMTO COMPUTE HANKEL                         C
C     FUNCTIONS OF THE FIRST KIND FOR ORDERS ONE AND ZERO.  THE       C
C     ARGUMENT IS VARIABLE R AND MUST BE POSITIVE.                    C
C                                                                     C
C.....HANKEL FUNCTIONS ARE OF FIRST KIND--  J+IY
C.....   N=0 RETURNS HZERO (H-ZERO)
C.....   N=1 RETURNS HONE  (H-ONE)
C.....   N=2 RETURNS HZERO AND HONE
C.....SUBROUTINE REQUIRES R>0
C.....SUBROUTINE ADAM MUST BE SUPPLIED BY USER
      DIMENSION A(7),B(7),C(7),D(7),E(7),F(7),G(7),H(7)
      REAL*8 A,B,C,D,E,F,G,H,N,R,T,X,Y,BJ,BY,FOOL
      COMPLEX*16 HZERO,HONE
      DATA A,B,C,D,E,F,G,H/1.0D0,-2.2499997D0,1.2656208D0,-0.3163866D0,
     &0.0444479D0,-0.0039444D0,0.00021D0,0.36746691D0,0.60559366D0,
     &-0.74350384D0,0.25300117D0,-0.04261214D0,0.00427916D0,
     &-0.00024846D0,0.5D0,-0.56249985D0,0.21093573D0,-0.03954289D0,
     &0.00443319D0,-0.00031761D0,0.00001109D0,-0.6366198D0,0.2212091D0,
     &2.1682709D0,-1.3164827D0,0.3123951D0,-0.0400976D0,0.0027873D0,
     &0.79788456D0,-0.00000077D0,-0.0055274D0,-0.00009512D0,
     &0.00137237D0,-0.00072805D0,0.00014476D0,-0.78539816D0,
     &-0.04166397D0,-0.00003954D0,0.00262573D0,-0.00054125D0,
     &-0.00029333D0,0.00013558D0,0.79788456D0,0.00000156D0,0.01659667D0,
     &0.00017105D0,-0.00249511D0,0.00113653D0,-0.00020033D0,
     &-2.35619449D0,0.12499612D0,0.0000565D0,-0.00637879D0,0.00074348D0,
     &0.00079824D0,-0.00029166/
      IF (R.LE.0.0D0) GO TO 50
      IF (N.LT.0.OR.N.GT.2) GO TO 50
      IF (R.GT.3.0D0) GO TO 20
      X=R*R/9.0D0
      IF (N.EQ.1) GO TO 10
      CALL ADAM(A,X,BJ)
      CALL ADAM(B,X,Y)
      BY=0.6366198D0*LOG(0.5D0*R)*BJ+Y
      HZERO=DCMPLX(BJ,BY)
      IF (N.EQ.0) RETURN
  10  CALL ADAM(C,X,Y)
      BJ=R*Y
      CALL ADAM(D,X,Y)
      BY=0.6366198D0*LOG(0.5D0*R)*BJ+Y/R
      HONE=DCMPLX(BJ,BY)
      RETURN
  20  X=3.0D0/R
      IF (N.EQ.1) GO TO 30
      CALL ADAM(E,X,Y)
      FOOL=Y/DSQRT(R)
```

```
      CALL ADAM(F,X,Y)
      T=R+Y
      BJ=FOOL*COS(T)
      BY=FOOL*SIN(T)
      HZERO=DCMPLX(BJ,BY)
      IF (N.EQ.0) RETURN
   30 CALL ADAM(G,X,Y)
      FOOL=Y/DSQRT(R)
      CALL ADAM(H,X,Y)
      T=R+Y
      BJ=FOOL*COS(T)
      BY=FOOL*SIN(T)
      HONE=DCMPLX(BJ,BY)
      RETURN
   50 WRITE(6,90) N,R
   90 FORMAT(32H0SICK DATA IN HANKZ1  *QUIT*  N=,I2,2X,2HR=,E11.3)
      CALL SYSTEM
      END
C*********************************************************************C
C*********************************************************************C
C                                                                     C
C*********************************************************************C
C                                                                     C
      SUBROUTINE ADAM(C,X,Y)
C                                                                     C
C*********************************************************************C
C                                                                     C
C     CALLED BY SUBROUTINE HANKZ1 TO COMPUTE THE VALUE OF A 7TH     C
C     ORDER POLYNOMIAL WHOSE ARGUMENT IS X AND COEFFICIENTS ARE     C
C     CONTAINED IN VECTOR C.                                        C
C                                                                     C
      DIMENSION C(7)
      REAL*8 X,Y,C
      INTEGER I
C
      Y=X*C(7)
C
         DO 10 I=1,5
         Y=X*(C(7-I)+Y)
   10    CONTINUE
C
      Y=Y+C(1)
      RETURN
      END

C*******************************************************************************
C*******************************************************************************
C*******************************************************************************
      FUNCTION CTAN(Z)

C*******************************************************************************
C     CALCULATES THE TANGENT OF A COMPLEX ARGUEMENT
C*******************************************************************************

      COMPLEX*16 Z,JCMPLX,CTAN
      REAL*8 RANGE,SGN,YMAG

      PARAMETER(JCMPLX=(0.D0,1.D0))
```

```
      RANGE=30.0D0
      SGN=1.0D0

      YMAG=ABS(DIMAG(Z))
      IF (YMAG.GT.RANGE) THEN
         SGN=SIGN(SGN,DIMAG(Z))
         CTAN=SGN*JCMPLX
      ELSE
         CTAN=CDSIN(Z)/CDCOS(Z)
      ENDIF

      RETURN
      END

C*******************************************************************************
C*******************************************************************************
C*******************************************************************************

      FUNCTION SINC(X)

      REAL*8 X,SINC

C*******************************************************************************
C     CALCULATES THE SINC = SIN(X)/(X) FUNCTION

      IF (X.NE.0.D0) THEN
         SINC = SIN(X)/X
      ELSE
         SINC = 1.0D0
      ENDIF

      RETURN
      END


C*******************************************************************************
C*******************************************************************************
C*******************************************************************************
C
C        MATRIX11 CALCULATES THE ELEMENTS OF THE IMPEDANCE SUBMATRIX Z11
C
C        COMPUTATION OF THE ELECTRIC FIELD COMPONENTS ON THE STRIP
C        DUE TO CURRENTS ON THE STRIP.
C*******************************************************************************

      SUBROUTINE MTRX11(K8M,K8N,STRIP,IEOPL,IEOMN,ZDIM,
     &                  A,B,C,H,L,L8U,X8O,Y8O,W,NSECM,
     &                  EPSCAV,EPSLY1,K8C,K8B,K8CS2,COEF11)

C********* parameter declarations

      INTEGER ZDIM

C*********  ELECTRICAL PARAMETER DECLARATIONS
      COMPLEX*16 EPSCAV,EPSLY1

      COMPLEX*16 K8C,K8G1,K8GC,K8CS2
```

```
      REAL*8 K8M,K8N,K8MS2,K8NS2,K8MNS2,K8B

C*********  GEOMETRY DECLARATIONS
      REAL*8 A,B,C,H
      REAL*8 X8O,Y8O,L,L8U,W,STRIP(1)
      INTEGER NSECM
      REAL*8 XPL,XMN

C*********  GREEN'S FUNCTION AND IMMITTANCE MATRIX DECLARATIONS
      COMPLEX*16 IEOPL(1),IEOMN(1)
      COMPLEX*16 N8EPP,N8MPP,ETA8EP,ETA8MP,
     &         GXX,COEF,CTNCAV,CTAN,CTNGT
      COMPLEX*16 N8MPPP
      REAL*8 RCOEF,COEF11
      REAL*8 COSI(500),COSIS(1000),SINMPL,TCOUPL

C*********  LOOP DECLARATIONS
      INTEGER I

C*********  ADDED DECLARATIONS FOR RECTI-LINEAR CASE
      COMPLEX*16 HZERO,HONE
      REAL*8 COSIPL

C*****************************************************************************
C  DETERMINE PROPAGATION CONSTANTS AND IMPEDANCE FACTORS
C
C     K8G1       -PROPAGATION CONSTANT IN LAYER1
C     K8GC       -PROPAGATION CONSTANT IN CAVITY
C     ETA8EP     -MODIFIED IMPEDANCE BOUNDARY CONDITION FOR LSE MODES
C     ETA8MP     -MODIFIED IMPEDANCE BOUNDARY CONDITION FOR LSM MODES
C     EPSCAV     -RELATIVE DIELECTRIC CONSTANT OF CAVITY ABOVE THE MICROSTIP
C     EPSLY1     -RELATIVE DIELECTRIC CONSTANT OF DIELECTRIC LAYER 1
C     N8EPP      -FACTOR SIN K8GC(C-H)*ETA8EP/D8E IN GREEN'S FUNCTION COMPONENT
C     N8MPP      -FACTOR SIN K8GC(C-H)*ETA8MP/D8M IN GREEN'S FUNCTION COMPOPENT
C*****************************************************************************

      K8MS2 = K8M * K8M
      K8NS2 = K8N * K8N
      K8MNS2 = K8MS2 + K8NS2

      K8G1= K8CS2 * EPSLY1/EPSCAV - K8MNS2
      K8GC= K8CS2 - K8MNS2

      K8G1=CDSQRT(K8G1)
      K8GC=CDSQRT(K8GC)

      CTNGT = CTAN(K8G1*H)
      CTNCAV = CTAN( K8GC*(C-H) )

      ETA8EP=-CTNGT/K8G1
      ETA8MP=-K8G1*CTNGT*(EPSCAV/EPSLY1)

      N8EPP=CTNCAV/(K8GC - CTNCAV/ETA8EP)
      N8MPP=CTNCAV/(1.D0/K8GC - CTNCAV/ETA8MP)

C*****************************************************************************
C             PERFORM THE M-O-M INTEGRATIONS OVER THE CURRENT SUBSECTION      *
C*****************************************************************************
```

```
C     SINC FUNCTIONS

      IF ( (K8M+K8B).EQ.0.D0 ) THEN
         IF ( (K8M-K8B).EQ.0.D0) THEN
             TCOUPL = 1.D0
         ELSE
             TCOUPL = SIN((K8M-K8B)*L8U/2.D0)/((K8M-K8B)*L8U/2.D0)
         ENDIF
      ELSE
         IF ( (K8M-K8B).EQ.0.D0) THEN
             TCOUPL = SIN((K8M+K8B)*L8U/2.D0)/((K8M+K8B)*L8U/2.D0)
         ELSE
             TCOUPL = SIN((K8M+K8B)*L8U/2.D0)*SIN((K8M-K8B)*L8U/2.D0)
     &                /( ((K8M+K8B)*L8U/2.D0)*((K8M-K8B)*L8U/2.D0) )
         ENDIF
      ENDIF

      TCOUPL = (L8U*L8U*K8B) * TCOUPL / SIN(K8B*L8U)

C     BESSEL FUNCTIONS

      IF( (K8N*W/2.D0).GT.0.D0) THEN
         CALL HANKZ1( (K8N*W/2.D0),0,HZERO,HONE)
      ELSE
         HZERO=DCMPLX(1.D0,0.D0)
      ENDIF
      COSIPL=DREAL(HZERO)

      SINMPL = SIN(K8N*Y8O) * COSIPL

C*************************************************************************
      RCOEF = TCOUPL**2 * SINMPL**2

      DO 1001 I = 1, 2 * NSECM - 1
         XPL = -L+FLOAT(I-1)*L8U
         COSIS(I) = RCOEF *COS(K8M *(XPL + 2.D0*X8O))
 1001 CONTINUE
      DO 1002 I = 1, NSECM
         XMN = FLOAT(I-1)*L8U
         COSI(I) = RCOEF *COS(K8M*XMN)
 1002 CONTINUE

C*************************************************************************
C COEFICIENTS OF GREEN'S FUNCTION IN CAVITY FIXED COORDINATES

      IF ( (K8M.EQ.0.D0).OR.(K8N.EQ.0.D0) ) THEN
         RCOEF = COEF11/(2.D0*K8MNS2)
      ELSE
         RCOEF = COEF11/K8MNS2
      ENDIF

      COEF = DCMPLX(0.D0,RCOEF)
      N8MPPP = N8MPP/K8CS2

      GXX =  (K8NS2 * N8EPP + K8MS2 * N8MPPP )

      DO 1003 I = 1, 2 * NSECM - 1
```

```
         IEOPL(I) = IEOPL(I) + COEF * GXX * COSIS(I)
 1003 CONTINUE
      DO 1004 I = 1, NSECM
         IEOMN(I) = IEOMN(I) + COEF * GXX * COSI(I)
 1004 CONTINUE

      RETURN
      END

C*******************************************************************************
C*******************************************************************************
C*******************************************************************************
C
C        MATRIX12 CALCULATES THE ELEMENTS OF THE ADMITTANCE SUBMATRIX Y12
C
C        COMPUTATION OF THE ELECTRIC FIELD COMPONENTS ON THE STRIP
C        DUE TO MAGNETIC CURRENTS ON THE SLOT AND THE COMPLEMENT BY SYMMETRY.
C                          ----- Y12 SUBMATRIX
C                          ----- Z31 SUBMATRIX BY SYMMETRY
C                          ----- FOR POINT MATCH COMPUTE Y13 SEPERATELY
C*******************************************************************************

      SUBROUTINE MTRX12(K8M,K8N,DONE,STRIP,SLOT,ZMATRX,ZDIM,
     &                  A,B,C,H,L,L8U,X8O,Y8O,W,NSECM,
     &                  LS,L8S,S8O,T8O,WS,NSECS,
     &                  EPSCAV,EPSLY1,K8C,K8B,K8CS2,COEF12)

C********* parameter declarations

      INTEGER ZDIM

C*********  ELECTRICAL PARAMETER DECLARATIONS
      COMPLEX*16 EPSCAV,EPSLY1

      COMPLEX*16 K8C,K8G1,K8GC,K8GCH,K8CS2
      REAL*8 K8M,K8N,K8MS2,K8NS2,K8MNS2,K8B

C*********  GEOMETRY DECLARATIONS
      REAL*8 A,B,C,H
      REAL*8 X8O,Y8O,L,L8U,W,STRIP(1)
      REAL*8 S8O,T8O,LS,L8S,WS,SLOT(1)
      INTEGER NSECM,NSECS

C*********  GREEN'S FUNCTION AND IMMITTANCE MATRIX DECLARATIONS

      COMPLEX*16 ZMATRX(ZDIM,ZDIM)
      COMPLEX*16 N8EPP,N8MPP,ETA8EP,ETA8MP,
     &        GXY,GUS,
     &        COEF,CSNE,CTNCAV,CTAN,CTNGT
      REAL*8 RCOEF,COEF12
      REAL*8 COSI(500),COSIS(500)
      REAL*8 SINMPL,COSMPL,TCOUPL,TSIUPL

C*********  LOOP DECLARATIONS
      INTEGER I,J
      LOGICAL DONE

C*********  ADDED DECLARATIONS FOR RECTI-LINEAR CASE
```

```
      COMPLEX*16 HZERO,HONE
      REAL*8 COSIPL

C*************************************************************************
C  DETERMINE PROPAGATION CONSTANTS AND IMPEDANCE FACTORS
C
C     K8G1       -PROPAGATION CONSTANT IN LAYER1
C     K8GC       -PROPAGATION CONSTANT IN CAVITY
C     ETA8EP     -MODIFIED IMPEDANCE BOUNDARY CONDITION FOR LSE MODES
C     ETA8MP     -MODIFIED IMPEDANCE BOUNDARY CONDITION FOR LSM MODES
C     EPSCAV     -RELATIVE DIELECTRIC CONSTANT OF CAVITY ABOVE THE MICROSTIP
C     EPSLY1     -RELATIVE DIELECTRIC CONSTANT OF DIELECTRIC LAYER 1
C     N8EPP      -FACTOR SIN K8GC(C-H)*ETA8EP/D8E IN GREEN'S FUNCTION COMPONENT
C     N8MPP      -FACTOR SIN K8GC(C-H)*ETA8MP/D8M IN GREEN'S FUNCTION COMPOPENT
C*************************************************************************

      K8MS2 = K8M * K8M
      K8NS2 = K8N * K8N
      K8MNS2 = K8MS2 + K8NS2

      K8G1= K8CS2 * EPSLY1/EPSCAV - K8MNS2
      K8GC= K8CS2 - K8MNS2

      K8G1=CDSQRT(K8G1)
      K8GC=CDSQRT(K8GC)

      CTNGT = CTAN(K8G1*H)

      K8GCH=K8GC*(C-H)
      CTNCAV = CTAN(K8GCH)
      IF (DIMAG(K8GCH).LT.30.D0) THEN
          CSNE = CDSIN(K8GCH)
      ELSE
          DONE = .TRUE.
          GOTO 1000
      ENDIF

      IF (DONE) GOTO 1000
      ETA8EP = - K8GC * CTNGT/K8G1
      ETA8MP = - (K8G1*CTNGT/K8GC)*(EPSCAV/EPSLY1)

      N8EPP = CTNCAV/(1.D0 - CTNCAV/ETA8EP)
      N8MPP = CTNCAV/(1.D0 - CTNCAV/ETA8MP)

C*************************************************************************
C             PERFORM THE M-O-M INTEGRATIONS OVER THE CURRENT SUBSECTION      *
C*************************************************************************

C     SINC FUNCTIONS

      IF ( (K8M+K8B).EQ.0.D0 ) THEN
         IF ( (K8M-K8B).EQ.0.D0) THEN
             TCOUPL = 1.D0
         ELSE
             TCOUPL = SIN((K8M-K8B)*L8U/2.D0)/((K8M-K8B)*L8U/2.D0)
         ENDIF
      ELSE
         IF ( (K8M-K8B).EQ.0.D0) THEN
```

```
            TCOUPL = SIN((K8M+K8B)*L8U/2.D0)/((K8M+K8B)*L8U/2.D0)
         ELSE
            TCOUPL = SIN((K8M+K8B)*L8U/2.D0)*SIN((K8M-K8B)*L8U/2.D0)
     &              /( ((K8M+K8B)*L8U/2.D0)*((K8M-K8B)*L8U/2.D0) )
         ENDIF
       ENDIF

       TCOUPL = (L8U*L8U*K8B) * TCOUPL / SIN(K8B*L8U)

C      BESSEL FUNCTIONS

       IF( (K8N*W/2.D0).GT.0.D0) THEN
          CALL HANKZ1( (K8N*W/2.D0),0,HZERO,HONE)
       ELSE
          HZERO=DCMPLX(1.D0,0.D0)
       ENDIF
       COSIPL=DREAL(HZERO)

       SINMPL = SIN(K8N*Y8O) * COSIPL

C*****************************************************************************

       DO 1001 I = 1,NSECM
          COSI(I)  = TCOUPL*COS(K8M*(STRIP(I)+X8O)) * SINMPL
 1001  CONTINUE

C*****************************************************************************

C      SINC FUNCTIONS

       IF ( (K8N+K8B).EQ.0.D0 ) THEN
          IF ( (K8N-K8B).EQ.0.D0) THEN
             TSIUPL = 1.D0
          ELSE
             TSIUPL = SIN((K8N-K8B)*L8S/2.D0)/((K8N-K8B)*L8S/2.D0)
          ENDIF
       ELSE
          IF ( (K8N-K8B).EQ.0.) THEN
             TSIUPL = SIN((K8N+K8B)*L8S/2.D0)/((K8N+K8B)*L8S/2.D0)
          ELSE
             TSIUPL = SIN((K8N+K8B)*L8S/2.D0)*SIN((K8N-K8B)*L8S/2.D0)
     &              /( ((K8N+K8B)*L8S/2.D0)*((K8N-K8B)*L8S/2.D0) )
          ENDIF
       ENDIF

       TSIUPL = (L8S*L8S*K8B) * TSIUPL / SIN(K8B*L8S)

C      BESSEL FUNCTIONS

       IF( (K8M*WS/2.D0).GT.0.D0) THEN
          CALL HANKZ1( (K8M*WS/2.D0),0,HZERO,HONE)
       ELSE
          HZERO=DCMPLX(1.D0,0.D0)
       ENDIF
       COSIPL=DREAL(HZERO)

       COSMPL = COS(K8M*S8O) * COSIPL
```

```
C*******************************************************************************

      DO 1002 I = 1,NSECS
         COSIS(I) = TSIUPL *SIN(K8N *(SLOT(I) + T8O)) * COSMPL
 1002 CONTINUE

C*******************************************************************************
C COEFICIENTS OF GREEN'S FUNCTION IN CAVITY FIXED COORDINATES
C  CORRECT FOR SIN K8GC(C-H) FACTOR

      IF ( (K8M.EQ.0.D0).OR.(K8N.EQ.0.D0) ) THEN
         RCOEF = COEF12/(2.D0*K8MNS2)
      ELSE
         RCOEF = COEF12/K8MNS2
      ENDIF

      COEF = RCOEF/CSNE

      GXY = - ( K8NS2 * N8EPP + K8MS2 * N8MPP )

      DO 1003 I = 1, NSECM
         DO 1004 J = 1, NSECS
            GUS =  GXY * COSI(I) * COSIS(J)
            ZMATRX(I,J+NSECM) = ZMATRX(I,J+NSECM) + COEF * GUS
 1004    CONTINUE
 1003 CONTINUE

 1000 RETURN
      END

C*******************************************************************************
C*******************************************************************************
C*******************************************************************************
C
C        MATRIX21 CALCULATES THE ELEMENTS OF THE ADMITTANCE SUBMATRIX Z31
C
C        COMPUTATION OF THE MAGNETIC FIELD COMPONENTS ON THE SLOT
C        DUE TO ELECTRIC CURRENTS ON THE STRIP
C                           ----- NEEDED ONLY FOR THE POINT MATCH CASE
C*******************************************************************************

      SUBROUTINE MTRX21(K8M,K8N,DONE,STRIP,SLOT,ZMATRX,ZDIM,
     &                  A,B,C,H,L,L8U,X8O,Y8O,W,NSECM,
     &                  LS,L8S,S8O,T8O,WS,NSECS,
     &                  EPSCAV,EPSLY1,K8C,K8B,K8CS2,COEF12,POINT)

C********* parameter declarations

      INTEGER ZDIM

C*********  ELECTRICAL PARAMETER DECLARATIONS
      COMPLEX*16 EPSCAV,EPSLY1

      COMPLEX*16 K8C,K8G1,K8GC,K8GCH,K8CS2
      REAL*8 K8M,K8N,K8MS2,K8NS2,K8MNS2,K8B

C*********  GEOMETRY DECLARATIONS
      REAL*8 A,B,C,H
```

```
      REAL*8 X8O,Y8O,L,L8U,W,STRIP(1)
      REAL*8 S8O,T8O,LS,L8S,WS,SLOT(1)
      INTEGER NSECM,NSECS

C*********  GREEN'S FUNCTION AND IMMITTANCE MATRIX DECLARATIONS

      COMPLEX*16 ZMATRX(ZDIM,ZDIM)
      COMPLEX*16 N8EPP,N8MPP,ETA8EP,ETA8MP,
     &         GXY,GUS,
     &         COEF,CSNE,CTNCAV,CTAN,CTNGT
      REAL*8 RCOEF,COEF12
      REAL*8 COSI(500),COSIS(500)
      REAL*8 SINMPL,COSMPL,TCOUPL,TSIUPL

C*********  LOOP DECLARATIONS
      INTEGER I,J
      LOGICAL DONE
      LOGICAL POINT

C*********  ADDED DECLARATIONS FOR RECTI-LINEAR CASE
      COMPLEX*16 HZERO,HONE
      REAL*8 COSIPL

C*****************************************************************************
C  DETERMINE PROPAGATION CONSTANTS AND IMPEDANCE FACTORS
C
C     K8G1      -PROPAGATION CONSTANT IN LAYER1
C     K8GC      -PROPAGATION CONSTANT IN CAVITY
C     ETA8EP    -MODIFIED IMPEDANCE BOUNDARY CONDITION FOR LSE MODES
C     ETA8MP    -MODIFIED IMPEDANCE BOUNDARY CONDITION FOR LSM MODES
C     EPSCAV    -RELATIVE DIELECTRIC CONSTANT OF CAVITY ABOVE THE MICROSTIP
C     EPSLY1    -RELATIVE DIELECTRIC CONSTANT OF DIELECTRIC LAYER 1
C     N8EPP     -FACTOR SIN K8GC(C-H)*ETA8EP/D8E IN GREEN'S FUNCTION COMPONENT
C     N8MPP     -FACTOR SIN K8GC(C-H)*ETA8MP/D8M IN GREEN'S FUNCTION COMPOPENT
C*****************************************************************************

      K8MS2 = K8M * K8M
      K8NS2 = K8N * K8N
      K8MNS2 = K8MS2 + K8NS2

      K8G1= K8CS2 * EPSLY1/EPSCAV - K8MNS2
      K8GC= K8CS2 - K8MNS2

      K8G1=CDSQRT(K8G1)
      K8GC=CDSQRT(K8GC)

      CTNGT = CTAN(K8G1*H)

      K8GCH=K8GC*(C-H)
      CTNCAV = CTAN(K8GCH)
      IF (DIMAG(K8GCH).LT.30.D0) THEN
          CSNE = CDSIN(K8GCH)
      ELSE
          DONE = .TRUE.
          GOTO 1000
      ENDIF

      IF (DONE) GOTO 1000
```

```
      ETA8EP = - K8GC * CTNGT/K8G1
      ETA8MP = - (K8G1*CTNGT/K8GC)*(EPSCAV/EPSLY1)

      N8EPP = CTNCAV/(1.D0 - CTNCAV/ETA8EP)
      N8MPP = CTNCAV/(1.D0 - CTNCAV/ETA8MP)

C*****************************************************************************
C             PERFORM THE M-O-M INTEGRATIONS OVER THE CURRENT SUBSECTION       *
C*****************************************************************************

C     SINC FUNCTIONS

      IF ( (K8M+K8B).EQ.0.D0 ) THEN
         IF ( (K8M-K8B).EQ.0.D0) THEN
             TCOUPL = 1.D0
         ELSE
             TCOUPL = SIN((K8M-K8B)*L8U/2.D0)/((K8M-K8B)*L8U/2.D0)
         ENDIF
      ELSE
         IF ( (K8M-K8B).EQ.0.D0) THEN
             TCOUPL = SIN((K8M+K8B)*L8U/2.D0)/((K8M+K8B)*L8U/2.D0)
         ELSE
             TCOUPL = SIN((K8M+K8B)*L8U/2.D0)*SIN((K8M-K8B)*L8U/2.D0)
     &                /( ((K8M+K8B)*L8U/2.D0)*((K8M-K8B)*L8U/2.D0) )
         ENDIF
      ENDIF

      TCOUPL = (L8U*L8U*K8B) * TCOUPL / SIN(K8B*L8U)

C     BESSEL FUNCTIONS

      IF( (K8N*W/2.D0).GT.0.D0) THEN
         CALL HANKZ1( (K8N*W/2.D0),0,HZERO,HONE)
      ELSE
         HZERO=DCMPLX(1.D0,0.D0)
      ENDIF
      COSIPL=DREAL(HZERO)

      SINMPL = SIN(K8N*Y8O) * COSIPL

C*****************************************************************************

      DO 1001 I = 1,NSECM
         COSI(I)  = TCOUPL*COS(K8M *(STRIP(I)+X8O)) *SINMPL
 1001 CONTINUE

C*****************************************************************************

C     SINC FUNCTIONS

         IF ( (K8N+K8B).EQ.0.D0 ) THEN
            IF ( (K8N-K8B).EQ.0.D0) THEN
               TSIUPL = 1.D0
            ELSE
               TSIUPL = SIN((K8N-K8B)*L8S/2.D0)/((K8N-K8B)*L8S/2.D0)
            ENDIF
         ELSE
            IF ( (K8N-K8B).EQ.0.) THEN
```

```
               TSIUPL = SIN((K8N+K8B)*L8S/2.D0)/((K8N+K8B)*L8S/2.D0)
            ELSE
               TSIUPL = SIN((K8N+K8B)*L8S/2.D0)*SIN((K8N-K8B)*L8S/2.D0)
     &                  /( ((K8N+K8B)*L8S/2.D0)*((K8N-K8B)*L8S/2.D0) )
            ENDIF
         ENDIF

         TSIUPL = (L8S*L8S*K8B) * TSIUPL / SIN(K8B*L8S)

C     BESSEL FUNCTIONS

      IF( (K8M*WS/2.D0).GT.0.D0) THEN
         CALL HANKZ1( (K8M*WS/2.D0),0,HZERO,HONE)
      ELSE
         HZERO=DCMPLX(1.D0,0.D0)
      ENDIF
      COSIPL=DREAL(HZERO)

      COSMPL = COS(K8M*S8O)
      IF (.NOT.POINT) COSMPL = COSMPL * COSIPL

C*******************************************************************************

      DO 1002 I = 1,NSECS
         COSIS(I) = TSIUPL *SIN(K8N *(SLOT(I) + T8O)) *COSMPL
 1002 CONTINUE

C*******************************************************************************
C COEFICIENTS OF GREEN'S FUNCTION IN CAVITY FIXED COORDINATES
C  CORRECT FOR SIN K8GC(C-H) FACTOR

      IF ( (K8M.EQ.0.D0).OR.(K8N.EQ.0.D0) ) THEN
         RCOEF = COEF12/(2.D0*K8MNS2)
      ELSE
         RCOEF = COEF12/K8MNS2
      ENDIF

      COEF = RCOEF/CSNE

      GXY = - ( K8NS2 * N8EPP + K8MS2 * N8MPP )

      DO 1003 I = 1, NSECS
         DO 1004 J = 1, NSECM
            GUS =  GXY * COSI(J) * COSIS(I)
            ZMATRX(I+NSECM,J) = ZMATRX(I+NSECM,J) + COEF * GUS
 1004    CONTINUE
 1003 CONTINUE

 1000 RETURN
      END

C*******************************************************************************
C*******************************************************************************
C*******************************************************************************
C
C        MATRIX33 CALCULATES THE ELEMENTS OF THE ADMITTANCE SUBMATRIX Z33
C
C        COMPUTATION OF THE MAGNETIC FIELD COMPONENTS ON THE SLOT
```

```
C        DUE TO MAGNETIC CURRENTS ON THE SLOT.
C*************************************************************************

      SUBROUTINE MTRX33(K8M,K8N,SLOT,IEOPL,IEOMN,ZDIM,
     &                  A,B,C,H,LS,L8S,S8O,T8O,WS,NSECS,
     &                  EPSCAV,EPSLY1,K8C,K8B,K8CS2,COEF33,POINT)

C********* parameter declarations

      INTEGER ZDIM

C*********  ELECTRICAL PARAMETER DECLARATIONS
      COMPLEX*16 EPSCAV,EPSLY1

      COMPLEX*16 K8C,K8G1,K8GC,KGKCS2,K8CS2
      REAL*8 K8M,K8N,K8MS2,K8NS2,K8MNS2,K8B

C*********  GEOMETRY DECLARATIONS
      REAL*8 A,B,C,H,S8O,T8O,LS,L8S,WS,SLOT(1)
      INTEGER NSECS
      REAL*8 XPL,XMN

C*********  GREEN'S FUNCTION AND IMMITTANCE MATRIX DECLARATIONS

      COMPLEX*16 IEOPL(2*ZDIM),IEOMN(ZDIM)
      COMPLEX*16 NEDE,NMDM,ETA8EP,ETA8MP,
     &        GYY,COEF,CTNCAV,CTAN,CTNGT
      COMPLEX*16 NEDEP
      REAL*8 RCOEF,COEF33
      REAL*8 COSI(500),COSIS(1000)
      REAL*8 COSMPL,TSIUPL

C*********  LOOP DECLARATIONS
      INTEGER I
      LOGICAL POINT

C*********  ADDED DECLARATIONS FOR RECTI-LINEAR CASE
      COMPLEX*16 HZERO,HONE
      REAL*8 COSIPL

C*************************************************************************
C  DETERMINE PROPAGATION CONSTANTS AND IMPEDANCE FACTORS
C
C     K8G1      -PROPAGATION CONSTANT IN LAYER1
C     K8GC      -PROPAGATION CONSTANT IN CAVITY
C     ETA8EP    -MODIFIED IMPEDANCE BOUNDARY CONDITION FOR LSE MODES
C     ETA8MP    -MODIFIED IMPEDANCE BOUNDARY CONDITION FOR LSM MODES
C     EPSCAV    -RELATIVE DIELECTRIC CONSTANT OF CAVITY ABOVE THE MICROSTIP
C     EPSLY1    -RELATIVE DIELECTRIC CONSTANT OF DIELECTRIC LAYER 1
C     N8EPP     -FACTOR SIN K8GC(C-H)*ETA8EP/D8E IN GREEN'S FUNCTION COMPONENT
C     N8MPP     -FACTOR SIN K8GC(C-H)*ETA8MP/D8M IN GREEN'S FUNCTION COMPOPENT
C*************************************************************************

      K8MS2 = K8M * K8M
      K8NS2 = K8N * K8N
      K8MNS2 = K8MS2 + K8NS2

      K8G1= K8CS2 * EPSLY1/EPSCAV - K8MNS2
```

```
      K8GC= K8CS2 - K8MNS2

      K8G1=CDSQRT(K8G1)
      K8GC=CDSQRT(K8GC)

      KGKCS2 = (K8GC/K8C)**2

      CTNGT = CTAN(K8G1*H)
      CTNCAV = CTAN(K8GC*(C-H))

      ETA8EP=-K8GC*CTNGT/K8G1
      ETA8MP=-(K8G1*CTNGT/K8GC)*(EPSCAV/EPSLY1)

      NEDE=(ETA8EP*CTNCAV+1.0D0)/(ETA8EP-CTNCAV)
      NMDM=(ETA8MP*CTNCAV+1.0D0)/(ETA8MP-CTNCAV)

C*******************************************************************************
C             PERFORM THE M-O-M INTEGRATIONS OVER THE CURRENT SUBSECTION         *
C*******************************************************************************

C     SINC FUNCTIONS

         IF ( (K8N+K8B).EQ.0.D0 ) THEN
            IF ( (K8N-K8B).EQ.0.D0) THEN
                TSIUPL = 1.0D0
            ELSE
                TSIUPL = SIN((K8N-K8B)*L8S/2.D0)/((K8N-K8B)*L8S/2.D0)
            ENDIF
         ELSE
            IF ( (K8N-K8B).EQ.0.D0) THEN
                TSIUPL = SIN((K8N+K8B)*L8S/2.D0)/((K8N+K8B)*L8S/2.D0)
            ELSE
                TSIUPL = SIN((K8N+K8B)*L8S/2.D0)*SIN((K8N-K8B)*L8S/2.D0)
     &                   /( ((K8N+K8B)*L8S/2.D0)*((K8N-K8B)*L8S/2.D0) )
            ENDIF
         ENDIF

         TSIUPL = (L8S*L8S*K8B) * TSIUPL / SIN(K8B*L8S)

C     BESSEL FUNCTIONS

      IF( (K8M*WS/2.0D0).GT.0.0D0) THEN
         CALL HANKZ1( (K8M*WS/2.0D0),0,HZERO,HONE)
      ELSE
         HZERO=DCMPLX(1.0D0,0.0D0)
      ENDIF
      COSIPL=DREAL(HZERO)

      COSMPL = COS(K8M*S8O)**2

C*******************************************************************************

      IF (.NOT.POINT) THEN
         RCOEF = TSIUPL**2 * COSMPL * COSIPL**2
C                                      Galerkin's method has J_o to second order
      ELSE
         RCOEF = TSIUPL**2 * COSMPL * COSIPL
C                                            Point match has J_o to first order
```

```
      ENDIF

      DO 1001 I = 1, 2 * NSECS - 1
         XPL = 2.D0*(-LS/2.+L8S)+FLOAT(I-1)*L8S
         COSIS(I) = RCOEF *COS(K8N *(XPL + 2.D0*T8O))
 1001 CONTINUE
      DO 1002 I = 1, NSECS
         XMN = FLOAT(I-1)*L8S
         COSI(I) = RCOEF *COS(K8N*XMN)
 1002 CONTINUE

C*************************************************************************
C COEFICIENTS OF GREEN'S FUNCTION IN CAVITY FIXED COORDINATES

      IF ( (K8M.EQ.0.).OR.(K8N.EQ.0.) ) THEN
         RCOEF = COEF33/(2.0*K8MNS2)
      ELSE
         RCOEF = COEF33/K8MNS2
      ENDIF

      COEF = DCMPLX(0.D0,RCOEF)
      COEF = COEF * EPSCAV / K8GC

      NEDEP = NEDE * KGKCS2

      GYY =   K8MS2 * NMDM + NEDEP * K8NS2

      DO 1003 I = 1, 2 * NSECS - 1
         IEOPL(I) = IEOPL(I) + COEF * GYY * COSIS(I)
 1003 CONTINUE
      DO 1004 I = 1, NSECS
         IEOMN(I) = IEOMN(I) + COEF * GYY * COSI(I)
 1004 CONTINUE

      RETURN
      END


C*************************************************************************
C*************************************************************************
C*************************************************************************
      SUBROUTINE CSPLNE(COORD,FIELD,N,CSPLIN)

      INTEGER I,N,NMAX
      PARAMETER(NMAX=500)

      REAL*8 COORD(N)
      REAL*8 LOSLOP,HISLOP
      REAL*8 RFIELD(NMAX),IFIELD(NMAX)
      REAL*8 RSPLNE(NMAX),ISPLNE(NMAX)

      COMPLEX*16 FIELD(N),CSPLIN(N)

C*************************************************************************

      DO 10 I = 1, N
         RFIELD(I) = DREAL(FIELD(I))
         IFIELD(I) = DIMAG(FIELD(I))
```

```
   10 CONTINUE

      LOSLOP  = ( DREAL(FIELD(2)) - DREAL(FIELD(1)) )
     &                  /( COORD(2) - COORD(1)  )
      HISLOP = ( DREAL(FIELD(N)) - DREAL(FIELD(N-1)) )
     &                  /( COORD(N) - COORD(N-1)  )

      CALL SPLINE(COORD,RFIELD,N,LOSLOP,HISLOP,RSPLNE)

      LOSLOP  = ( DIMAG(FIELD(2)) - DIMAG(FIELD(1)) )
     &                  /( COORD(2) - COORD(1)  )
      HISLOP = ( DIMAG(FIELD(N)) - DIMAG(FIELD(N-1)) )
     &                  /( COORD(N) - COORD(N-1)  )

      CALL SPLINE(COORD,IFIELD,N,LOSLOP,HISLOP,ISPLNE)

      DO 11 I = 1, N
         CSPLIN(I) = DCMPLX(RSPLNE(I),ISPLNE(I))
   11 CONTINUE

      RETURN
      END

C*******************************************************************************
C*******************************************************************************
C*******************************************************************************
      SUBROUTINE CSPLNT( COORD,FIELD,SPLINE,N,X,F)

      INTEGER N,I,NMAX
      PARAMETER(NMAX=500)

      REAL*8 COORD(N),X
      REAL*8 RSPLNE(NMAX),ISPLNE(NMAX)
      REAL*8 RFIELD(NMAX),IFIELD(NMAX)
      REAL*8 FIELDR,FIELDI

      COMPLEX*16 FIELD(N),SPLINE(N),F

C*******************************************************************************

      DO 10 I = 1, N
         RFIELD(I) = DREAL(FIELD(I))
         RSPLNE(I) = DREAL(SPLINE(I))
         IFIELD(I) = DIMAG(FIELD(I))
         ISPLNE(I) = DIMAG(SPLINE(I))
   10 CONTINUE

      CALL SPLINT(COORD,RFIELD,RSPLNE,N,X,FIELDR)
      CALL SPLINT(COORD,IFIELD,ISPLNE,N,X,FIELDI)

      F = DCMPLX(FIELDR,FIELDI)

      RETURN
      END



C*******************************************************************************
```

```
C*******************************************************************************
C*******************************************************************************
      SUBROUTINE SPLINE(X,Y,N,YP1,YPN,Y2)

      INTEGER N,I,K,NMAX
      PARAMETER (NMAX=500)

      REAL*8 YP1,YPN,SIG,P,QN,UN
      REAL*8 X(N),Y(N),Y2(N),U(NMAX)

C*******************************************************************************
      IF (YP1.GT..99D30) THEN
        Y2(1)=0.D0
        U(1)=0.D0
      ELSE
        Y2(1)=-0.5D0
        U(1)=(3.D0/(X(2)-X(1)))*((Y(2)-Y(1))/(X(2)-X(1))-YP1)
      ENDIF
      DO 11 I=2,N-1
        SIG=(X(I)-X(I-1))/(X(I+1)-X(I-1))
        P=SIG*Y2(I-1)+2.D0
        Y2(I)=(SIG-1.D0)/P
        U(I)=(6.D0*((Y(I+1)-Y(I))/(X(I+1)-X(I))-(Y(I)-Y(I-1))
     *       /(X(I)-X(I-1)))/(X(I+1)-X(I-1))-SIG*U(I-1))/P
11    CONTINUE
      IF (YPN.GT..99D30) THEN
        QN=0.D0
        UN=0.D0
      ELSE
        QN=0.5D0
        UN=(3.D0/(X(N)-X(N-1)))*(YPN-(Y(N)-Y(N-1))/(X(N)-X(N-1)))
      ENDIF
      Y2(N)=(UN-QN*U(N-1))/(QN*Y2(N-1)+1.D0)
      DO 12 K=N-1,1,-1
        Y2(K)=Y2(K)*Y2(K+1)+U(K)
12    CONTINUE
      RETURN
      END

C*******************************************************************************
C*******************************************************************************
C*******************************************************************************
      SUBROUTINE SPLINT(XA,YA,Y2A,N,X,Y)

      INTEGER N,KLO,KHI,K
      REAL*8 X,Y,H,A,B
      REAL*8 XA(N),YA(N),Y2A(N)

C*******************************************************************************

      KLO=1
      KHI=N
1     IF (KHI-KLO.GT.1) THEN
        K=(KHI+KLO)/2
        IF(XA(K).GT.X)THEN
          KHI=K
        ELSE
          KLO=K
```

```
        ENDIF
      GOTO 1
      ENDIF
      H=XA(KHI)-XA(KLO)
      IF (H.EQ.0.D0) PAUSE 'Bad XA input.'
      A=(XA(KHI)-X)/H
      B=(X-XA(KLO))/H
      Y=A*YA(KLO)+B*YA(KHI)+
     *      ((A**3-A)*Y2A(KLO)+(B**3-B)*Y2A(KHI))*(H**2)/6.D0
      RETURN
      END



C*******************************************************************************
C*******************************************************************************
C*******************************************************************************
C
C      SUBROUTINE SCATT CALCULATES THE INCIDENT TEM H-FIELD ON THE SLOT
C      FOR THE CALCULATION OF THE SCATTERING COEFFICIENTS USING THE REACTION
C      THEOREM
C
C      THIS VERSION IS FOR STRIPLINE ONLY
C
C*******************************************************************************

      SUBROUTINE SCATT(ZREACT,SPLPOS,HFIELD,HDIM,
     &                 A,C,X8O,W,K8Z,FREQ,MU8O,MNSTOP)

C*******************************************************************************
C     CHANGED COORDINATE SYSTEM NOTATION FOR WAVEGUIDE CALCULATION
C       - x/y and y/z axes are interchanged for the waveguide calculations
C       - x now is the transverse to strip direction; a is waveguide width
C         X8O is the strip center ; c is waveguide height

      INTEGER HDIM
      REAL*8 PI
      PARAMETER(PI=3.14159265358979324D0)

C*********  GEOMETRY DECLARATIONS

      REAL*8 A,C,X8O,W
      REAL*8 FIELDX,FIELDY,SPLPOS(1)

C*********  ELECTRICAL PARAMETER DECLARATIONS

      REAL*8 FREQ,MU8O,K8X
      COMPLEX*16 K8Y,K8Z

      REAL*8 RCOEF
      COMPLEX*16 COEF,ZREACT

C********* FIELD DECLARATIONS

      COMPLEX*16 HYZ,GXZ,HFIELD(1)

C*********  FUNCTIONS
```

```
      REAL*8 J8O
      COMPLEX*16 HZERO,HONE,COSCOS

C*********  LOOP DECLARATIONS
      INTEGER I,M,MNSTOP

C*******************************************************************************
C*******************************************************************************

      WRITE(*,*) 'CALCULATING THE REACTION ON CROSS SECTION'

      CALL REACT(A,C,X8O,W,K8Z,ZREACT,FREQ,MU8O,MNSTOP)

      ZREACT = 2.D0 * ZREACT

      WRITE(*,*)
      WRITE(*,100) DREAL(ZREACT)
  100 FORMAT(1X,'Reaction : ',F7.4,
     &          ' = Characteristic impedance for lossless case')

C     CALCULATE H-FIELD COMPONENT ALONG SLOT

      FIELDY = C
      DO 10 I = 0, HDIM - 1
         FIELDX = FLOAT(I) * A/FLOAT(HDIM-1)
         HYZ    = DCMPLX(0.D0,0.D0)

         DO 11 M = 1, MNSTOP
            K8X = M * PI / A
            K8Y = DCMPLX(0.D0,K8X)

            IF ( (K8X*W/2.D0).GT.0.D0) THEN
               CALL HANKZ1( (K8X*W/2.D0),0,HZERO,HONE)
            ELSE
               HZERO=DCMPLX(1.D0,0.D0)
            ENDIF
            J8O = REAL(HZERO)

            RCOEF = - 2.D0 * SIN(K8X * FIELDX) / A
            COEF  = COSCOS(K8Y*(C-FIELDY),K8Y*C/2.D0)
            GXZ   = 0.5D0 * RCOEF * COEF

            HYZ = HYZ + J8O * DSIN(K8X * X8O) * GXZ
   11    CONTINUE

         SPLPOS(I) = FIELDX
         HFIELD(I) = HYZ
   10 CONTINUE

      RETURN
      END



C*******************************************************************************
C*******************************************************************************
C*******************************************************************************
C
```

```
C        REACT CALCULATES THE REACTION ON THE CROSS SECTION OF THE WAVEGUIDE
C        FOR THE SELECTED K8Z EIGENVALUE.
C
C*******************************************************************************

      SUBROUTINE REACT(A,C,X8O,W,K8Z,ZREACT,FREQ,MU8O,MNSTOP)

      REAL*8 PI
      PARAMETER(PI=3.14159265358979324D0)

C*********  GEOMETRY DECLARATIONS

      REAL*8 A,C,X8O,W

C*********  ELECTRICAL PARAMETER DECLARATIONS

      REAL*8 MU8O,FREQ,K8X
      COMPLEX*16 K8Y,K8Z

C*********  DECLARATIONS FOR REACTION

      REAL*8 P1,P2,RCOEF,LENGTH,DIV1
      COMPLEX*16 DIV
      COMPLEX*16 ECOEF1,ECOEF2,ECOEF,HCOEF1,HCOEF2,HCOEF
      COMPLEX*16 ZREACT
      COMPLEX*16 REACTE,REACE1,REACE2,REACTM,REACM1,REACM2

C*********  LOOP DECLARATIONS

      INTEGER M,MNSTOP

C*********  FUNCTIONS

      REAL*8 J8O
      COMPLEX*16 CX8COS,CTAN,HZERO,HONE

C*******************************************************************************
C*******************************************************************************

      ZREACT   = DCMPLX(0.D0,0.D0)

      DO 10 M = 1, MNSTOP

         K8X = M * PI / A
         K8Y = DCMPLX(0.D0,K8X)

         IF ( (K8X*W/2.D0).GT.0.D0) THEN
            CALL HANKZ1( (K8X*W/2.D0),0,HZERO,HONE)
         ELSE
            HZERO = DCMPLX(1.D0,0.D0)
         ENDIF
         J8O = DREAL(HZERO)

         LENGTH = .5D0 * C

         DIV = CX8COS(K8Y * LENGTH)
         DIV1 = CDABS(DIV)
         DIV1 = DIV1**2
```

```
          RCOEF = DSIN(K8X * X8O)**2 * J8O**2
          RCOEF = RCOEF * 4.D0 * PI * FREQ * MU8O / A

C*************************************************************************
C      ( EX X HY ) MODES:
C*************************************************************************

          HCOEF  = DCMPLX(0.D0,.5D0)

C******* FOR THE LSE X LSE TERMS

             ECOEF1 = -DCMPLX(0.D0,.5D0) *K8Z /((K8X**2 + K8Z**2) )

             IF (DIV1.GE.1.D15) THEN
                P1 = 0.D0
             ELSE
                P1 = - .5D0 * LENGTH / DIV1
             ENDIF

             P2 = .5D0 * CTAN(K8Y * LENGTH)/K8Y

             REACE1 = P1 + P2
             REACTE = RCOEF * HCOEF * ECOEF1 * REACE1

C******* FOR THE LSM X LSE TERMS

             ECOEF2 = - .25D0*K8X/((K8X**2+K8Z**2)*K8Z)
             REACE2 = CTAN(K8Y * LENGTH)

             REACTE = REACTE + RCOEF * HCOEF * ECOEF2 * REACE2

C*************************************************************************
C       - ( EY X HX ) MODES:
C*************************************************************************

             ECOEF = 1.D0 / (8.D0 * K8Z * (K8X**2 + K8Z**2))

C******* FOR THE LSM X LSM TERMS

             HCOEF1 = (K8Z)**2

             IF (DIV1.GE.1.D15) THEN
                P1 = 0.D0
             ELSE
                P1 = LENGTH / DIV1
             ENDIF
             P2 = CTAN(K8Y * LENGTH)/K8Y

             REACM1 = P1 + P2
             REACTM = RCOEF * ECOEF * HCOEF1 * REACM1

C******* FOR THE LSM X LSE TERMS

             HCOEF2 = DCMPLX(0.D0,1.D0) * K8X
             REACM2 = CTAN(K8Y * LENGTH)

             REACTM = REACTM - RCOEF * ECOEF * HCOEF2 * REACM2
```

```
C*******************************************************************************

            ZREACT    = ZREACT    + REACTE + REACTM

   10 CONTINUE

      RETURN
      END



C*******************************************************************************
C*******************************************************************************
C*******************************************************************************

      FUNCTION CX8COS(X)

C     CALCULATES CCOS(X) FUNCTION WITH ASYMPTOTIC FORMS

      REAL*8 LIMIT1,LIMIT2,LIMIT3
      PARAMETER(LIMIT1 = 30.D0 )
      PARAMETER(LIMIT2 = 43.D0 )
      PARAMETER(LIMIT3 = 1.D19 )

      COMPLEX*16 CX8COS
      COMPLEX*16 X,JCMPLX
      PARAMETER(JCMPLX=(0.D0,1.D0))

      IF (DABS(DIMAG(X)).LT.LIMIT2) THEN
         IF (DIMAG(X).GT.LIMIT1) THEN
            CX8COS = DCOS(DREAL(X)) * DEXP(DIMAG(X))
     &                  - JCMPLX * DSIN(DREAL(X)) * DEXP(DIMAG(X))
         ELSE IF (DIMAG(X).LT.-LIMIT1) THEN
            CX8COS = DCOS(DREAL(X)) * DEXP(-DIMAG(X))
     &                  + JCMPLX * DSIN(DREAL(X)) * DEXP(-DIMAG(X))
         ELSE
            CX8COS = CDCOS(X)
         ENDIF
      ELSE
         CX8COS = LIMIT3
      ENDIF

      RETURN
      END



C*******************************************************************************
C*******************************************************************************
C*******************************************************************************

      FUNCTION COSCOS(X,Y)

C     CALCULATES CCOS(X)/CCOS(Y) FUNCTION WITH ASYMPTOTIC FORMS

      REAL*8 LIMIT1,LIMIT2
      PARAMETER(LIMIT1 = 25.D0 )
```

```
      PARAMETER(LIMIT2 = 45.D0 )

      COMPLEX*16 COSCOS
      COMPLEX*16 X,Y,CTAN
      REAL*8 CSHCSH,Z

      COMPLEX*16 JCMPLX
      PARAMETER(JCMPLX=(0.D0,1.D0))

      IF (DABS(DIMAG(X)).LT.LIMIT1) THEN

         IF (DABS(DIMAG(Y)).LT.LIMIT1) THEN
            CSHCSH = DCOSH(DIMAG(X))/DCOSH(DIMAG(Y))
         ELSE
            IF(DABS(DIMAG(Y)).LT.LIMIT2) THEN
               CSHCSH = 2.D0 * DCOSH(DIMAG(X))/DEXP(DABS(DIMAG(Y)))
            ELSE
               Z = LOG(DCOSH(DIMAG(X))) - DABS(DIMAG(Y)) + LOG(2.)
               IF (Z.LE.LIMIT2) THEN
                  CSHCSH = DEXP(Z)
               ELSE
                  CSHCSH = 0.D0
               ENDIF
            ENDIF
         ENDIF
      ELSE IF (DABS(DIMAG(X)).LT.LIMIT2) THEN
         IF (DABS(DIMAG(Y)).LT.LIMIT1) THEN
            CSHCSH = DEXP(DABS(DIMAG(X)))/(2.D0 * DCOSH(DIMAG(Y)) )
         ELSE
            Z = DABS(DIMAG(X))-DABS(DIMAG(Y))
            IF (Z.LE.LIMIT2) THEN
              CSHCSH = DEXP(Z)
            ELSE
              CSHCSH = 0.D0
            ENDIF
         ENDIF
      ELSE
         Z = DABS(DIMAG(X))-DABS(DIMAG(Y))
         IF (Z.LE.LIMIT2) THEN
            CSHCSH = DEXP(Z)
         ELSE
            CSHCSH = 0.D0
         ENDIF
      ENDIF

      COSCOS = CSHCSH * (DCOS(DREAL(X))
     &                  -DSIN(DREAL(X))*CTAN(DCMPLX(0.D0,DIMAG(X))))
      COSCOS = COSCOS / (DCOS(DREAL(Y))
     &                  -DSIN(DREAL(Y))*CTAN(DCMPLX(0.D0,DIMAG(Y))))

      RETURN
      END


C*********************************************************************
C*********************************************************************
C*********************************************************************
C     ..................  YIJ_DIEL_K0.FTN   ..............................
```

```
C
C     In this program AKK=AK0
C
C*********************************************************************


      SUBROUTINE YIJDIE(YSD,DLX,W,EER,NS1,NS2,NOFF,INS,INS1S2,
     &                  ER,H,T,A,AK0,AK,AKK,OFFSET)

C*********************************************************************

      INTEGER SLTDIM
      PARAMETER(SLTDIM=130)

      REAL*8 PI
      PARAMETER(PI=3.14159265358979324D0)

      COMPLEX*16 CI
      PARAMETER(CI=(0.D0,1.D0))

      INTEGER NS,NS1S2,NMAX(7),NS1,NS2,NSS2,NOFF,INS,INS1S2
      INTEGER NK0,NK0K,NDP,NSP,NTP,NPOINT,NSER,MPOINT,MA
      INTEGER I,NTM,NTE,IFIRST,K,MS,MS1S2

      REAL*8 A,E1,E2,FA
      REAL*8 CONST,GSK,ADL,TEP,TMP,TPO,CVON,YSIN,YCOS,WREAL
      REAL*8 ER,H,T,DLX,W,EER,AK0,AK,AX,AKK,OFFLIM

      REAL*8 CND(20),CNS(40),CNT(40),COAL(20),BCOAL(10),OFFSET(7)
      REAL*8 POINT(20),BPOINT(10),XND(20,2),XNS(40),XNT(40,3)
      REAL*8 AM(41),BM(151),DM(41),CN(51),SERA(5),SERS(5)
      REAL*8 POLES(40),POLTE(20),POLTM(20),VZXE(20),VXXM(20),VZXM(20)

      REAL*8 FSD(SLTDIM),GS(SLTDIM)
      COMPLEX*16 YSD(SLTDIM)

      REAL*8 S1(4,SLTDIM,7),D1(4,SLTDIM,7),D2(4,SLTDIM,7)
      REAL*8 T1(4,SLTDIM,7),T2(4,SLTDIM,7),T3(4,SLTDIM,7),T4(4,SLTDIM,7)
      REAL*8 S(10,2),RCOE(20,SLTDIM,7,10),DIST(SLTDIM,7,10)
      REAL*8 SSJ0(SLTDIM,7),SAJ0(SLTDIM,7)

C*********************************************************************

      DO 999 I = 1, NS1
         YSD(I) = DCMPLX(0.D0,0.D0)
         GS(I)  = DCMPLX(0.D0,0.D0)
         FSD(I) = DCMPLX(0.D0,0.D0)
  999 CONTINUE

      WREAL=W

*      W=WREAL*(1.D0+2.D0*WDELTA/WREAL)

C     Subroutine POLES evaluates the poles of the Green's function
C     and orders them according to their magnitude
C
C     IFIRST=0  :dominant mode is a TM wave
C            1  :dominant mode is a TE wave
```

```
C            2  :only one TM wave
C

      CALL SPOLES(ER,H,A,EER,POLTM,POLTE,POLES,NTM,NTE,IFIRST)

C
C     This subroutines gives data for the numerical integration
C

      CALL DATSLT(NK0,NK0K,NDP,NSP,NTP,NPOINT,NSER,MPOINT,MA,
     &            A,E1,E2,CND,CNS,CNT,COAL,BCOAL,FA,ER,
     &            POINT,BPOINT,XND,XNS,XNT)


      NS=NS1
      IF (NS1.LT.NS2) NS=NS2
      MS=NS
      IF (NOFF.EQ.1) GO TO 50
          NS1S2=NS2+NSS2-1
          MS1S2=NS1S2
          IF (NS1S2.GT.200) NS1S2=200

   50 CONTINUE
      IF(NMAX(INS).LE.(NS+2)) NMAX(INS)=NS+2
      IF (NOFF.EQ.1) GO TO 51
          IF(NMAX(INS1S2).LE.(NS1S2+2)) NMAX(INS1S2)=NS1S2+2

   51 ADL=AKK*DLX
      YSIN=DSIN(ADL)
      YCOS=DCOS(ADL)

C     For the normalization of the current along the y axis

                CVON=W*PI/2.D0

C     Computation of lamda-integration limits between 0 and A

      CALL LIMIT(NK0,MA,NTM,NTE,NK0K,IFIRST,A,EER,AK0,AK,ER,H,T,
     &           CN,BM,AM,DM,POLTM,POLTE,POLES,VXXM,VZXM,VZXE)


C     Evaluation of the Green's function at different points
C     in the interval [0,A]. The Bessel function has been excluded

      CALL GGREEN(ER,H,T,E1,E2,FA,COAL,BCOAL,
     &            CN,AM,BM,DM,MA,MPOINT,NPOINT,POINT,NK0,NTM,NTE,
     &            NK0K,IFIRST,INS,NOFF,NSER,NMAX,DLX,W,AK,AKK,AK0,
     &            OFFSET,OFFLIM,BPOINT,DIST,RCOE,AX,SERS,SERA,
     &            S,WREAL,POLTM,POLTE,NS,
     &            VXXM,VZXM,VZXE,YCOS,YSD,FSD,SSJ0,SAJ0,SLTDIM)


C     Evaluation of the tail contribution (from a to infinity)

      CALL TAIL(NS,NS1S2,NOFF,NDP,NTP,NSP,NMAX,INS,INS1S2,
     &          S1,T1,T2,T3,T4,D1,D2,GS,AX,WREAL,
     &          ER,H,T,DLX,W,E1,E2,AK0,AK,AKK,FA,OFFSET,
     &          XNS,XND,XNT,CNS,CND,CNT,SLTDIM)
```

```
      CONST=-(1.D0/CVON)*DSQRT(EER)/(480.D0*(PI**3)*YSIN*YSIN)

C      WRITE (6,99) CONST
C 99   FORMAT(10X,'CONST=',E14.7//)
C      WRITE(6,9)
C  9   FORMAT(///10X,'Contribution to admittance from the dielectric'///)
C      WRITE(6,10) MS
C  10  FORMAT(11X,I4)

      DO 11 K=1,MS
            YSD(K)=YSD(K)*CONST
            GSK=(GS(K)-FSD(K))*CONST
C           WRITE (6,30) K,YSD(K),GS(K),FSD(K)
C  30        FORMAT(1X,'K=',I4,2X,'YS=',E14.7,2X,E14.7,2X,
C     *      'GS=',E14.7,2X,'FSD=',E14.7)
            YSD(K)=(YSD(K)+GSK)*CI
  11  CONTINUE
C      DO 20 K=1,MS
C            WRITE (6,12) YSD(K)
C  12        FORMAT(10X,E14.7,1X,E14.7)
C  20  CONTINUE

 1000 CONTINUE
      RETURN
      END

C*********************************************************************
C*********************************************************************
C*********************************************************************
C     The name of this file is ..... POLES.FTN .............
C*********************************************************************
      SUBROUTINE SPOLES(ER,H,A,EER,TMP,TEP,TPO,NM,NE,IFIRST)

      REAL*8 PI
      PARAMETER(PI=3.14159265358979324D0)

      INTEGER NM,NE,IFIRST
      INTEGER I8I,IIF,IIP,NNK,IIW,IQW,I,IK,NK,I8MAX
      INTEGER LOR(40)

      REAL*8 ER,H,A,EER,X0,AK,AN,DP
      REAL*8 AK0,AER,XS0,XS1,ERR,XRA,XSA,QWR,Y,TEPMIN,TMPMIN
      REAL*8 TMP(1),TEP(1),TPO(1),XR(1),XS(1)

C.....................................................................
C
C     ER    :....Dielectric constant
C
C     H     :....Height of the dielectric substrate
C
C     NE    :....Number of TE surface waves
C
C     NM    :....Number of tm surface waves
C
C     XS    :....Matrix of poles contributing to TE surface waves
C
C     XR    :....Matrix of poles contributing to TM surface waves
```

```
C
C     ERR   :....Error in the computation of the poles
C
C.....................................................................

      DO 1000 I = 1, 20
         TMP(I) = 0.D0
         TEP(I) = 0.D0
         TPO(I) = 0.D0
 1000 CONTINUE
      DO 1001 I = 21, 40
         TPO(I) = 0.D0
 1001 CONTINUE

      AER=DSQRT(EER)
      ERR=0.0000001D0
      DP=H/AER
C
C     PART I : TE MODES
C
      AK0=2.D0*PI
      AK=DSQRT(ER)*AK0
      X0=DP*DSQRT(AK**2-AK0**2)
C---------------------------------------------------------------
C     WRITE (6,300) AK0,AK,X0,PI
C 300 FORMAT(10X,'AK0=',E14.7,2X,'AK=',E14.7,2X,'X0=',E14.7,
C    *2X,'PI=',E14.7/)
C---------------------------------------------------------------
      AN=X0/PI+0.5D0
      NE=AN
      IF (NE.EQ.0) GO TO 310
      DO 2 I=1,NE
           IF (X0-(2.D0*FLOAT(I)+1.D0)*PI/2.D0) 3,3,4
   4       XS0=(2.D0*FLOAT(I)-1.D0)*PI/2.D0+ERR
           XS1=(2.D0*FLOAT(I)+1.D0)*PI/2.D0-ERR
           GO TO 5
   3       XS0=(2.D0*FLOAT(I)-1.D0)*PI/2.D0+ERR
           XS1=X0
   5       CONTINUE
           IF (DABS(XS0-XS1)-ERR) 22,7,7
   7       XSA=(XS0+XS1)/2.D0
           Y=-DTAN(XSA)*DSQRT(X0**2-XSA**2)-XSA
           IF (Y) 8,9,10
   9       XS(I)=XSA
           GO TO 222
   8       XS1=XSA
           GO TO 5
  10       XS0=XSA
           GO TO 5
  22       XS(I)=(XS0+XS1)/2.D0
 222       XS(I)=DSQRT(AK**2-XS(I)**2/DP**2)
   2  CONTINUE
C---------------------------------------------------------------
C     WRITE (6,301) ER,H
C301  FORMAT(//10X,' Dielectric Constant=',D16.9/10X,'Substrate '
C    *,'Thickness',D16.9///)
C---------------------------------------------------------------
C 310  IF (NE.EQ.0) WRITE (6,304)
```

```
  310  IF (NE.EQ.0) A=A
 304  FORMAT(/////10X,'No TE waves excited in the substrate'//)
      IF (NE.EQ.0) GO TO 312
C      IF (NE.GT.0) WRITE (6,305) NE
 305  FORMAT(///10X,'There are',I4,
     *' TE waves excited in the substrate'//)
      DO 302 I=1,NE
             TEP(I)=XS(I)/AER
             IF (I.GT.1) THEN
                 I8MAX=I-1
                 DO 502 I8I=1,I8MAX
                    TEPMIN=TEP(I8I)
                    IF (TEP(I).LT.TEP(I8I)) THEN
                        TEP(I8I)=TEP(I)
                        TEP(I)=TEPMIN
                    END IF
 502                CONTINUE
             END IF
 302  CONTINUE
      DO 503 I=1,NE
C             WRITE (6,303) I,TEP(I)
 303         FORMAT (10X,I4,2X,D16.9)
 503  CONTINUE
 312  CONTINUE
C
C     END OF PART I
C
C
C     PART II : TM MODES
C
      AN=X0/PI+1.D0
      NM=AN
      DO 13 I=1,NM
            IF (X0-(2.D0*FLOAT(I)+1.D0)*PI/2.D0) 14,14,15
  15        XS1=FLOAT(I)*PI-PI/3.D0-0.01D0
            GO TO 16
  14        XS1=X0
  16        XS0=FLOAT(I-1)*PI+ERR
  17        CONTINUE
            IF (DABS(XS0-XS1)-ERR) 113,19,19
  19        XRA=(XS0+XS1)/2.D0
C-----------------------------------------------------------------------
C           WRITE (6,301) XRA
C 301       FORMAT(10X,'XRA=',E14.7/)
C-----------------------------------------------------------------------
            Y=DSQRT(ER)**2*(1.D0/DTAN(XRA))*DSQRT(X0**2-XRA**2)-XRA
            IF (Y) 20,21,24
  21        XR(I)=XRA
            GO TO 333
  20        XS1=XRA
            GO TO 17
  24        XS0=XRA
            GO TO 17
 113        XR(I)=(XS0+XS1)/2.D0
 333        XR(I)=DSQRT(AK**2-XR(I)**2/DP**2)
  13  CONTINUE
C      WRITE (6,307) NM
 307  FORMAT(///10X,'There are',I4,' TM waves excited in the substrate'/
```

```
     */)
      DO 308 I=1,NM
             TMP(I)=XR(I)/AER
             IF (I.GT.1) THEN
                 I8MAX=I-1
                 DO 508 I8I=1,I8MAX
                    TMPMIN=TMP(I8I)
                    IF (TMP(I).LT.TMP(I8I)) THEN
                        TMP(I8I)=TMP(I)
                        TMP(I)=TMPMIN
                    END IF
 508             CONTINUE
             END IF
 308  CONTINUE
      DO 506 I=1,NM
C             WRITE (6,306) I,TMP(I)
 306         FORMAT (10X,I4,2X,D16.9)
 506  CONTINUE
C 322  CONTINUE
C
      NK=NE+NM
      IF (NE.EQ.0) GO TO 350
      DO 411 IQW=1,NE
             TPO(IQW)=TEP(IQW)
             LOR(IQW)=1
  411 CONTINUE
  350 CONTINUE
      DO 412 IQW=1,NM
             TPO(NE+IQW)=TMP(IQW)
             LOR(NE+IQW)=0
  412 CONTINUE
C
      IF (NK.EQ.1) GO TO 416
      NNK=NK-1
      DO 415 IIP=1,NNK
             IK=IIP+1
             DO 413 IIF=IK,NK
                    QWR=TPO(IIP)
                    IIW=LOR(IIP)
                    IF (TPO(IIP).LT.TPO(IIF)) GO TO 413
                    TPO(IIP)=TPO(IIF)
                    LOR(IIP)=LOR(IIF)
                    TPO(IIF)=QWR
                    LOR(IIF)=IIW
  413        CONTINUE
  415 CONTINUE
      IF (LOR(1).EQ.0) IFIRST=0
      IF (LOR(1).EQ.1) IFIRST=1
      GO TO 417
C
  416 IFIRST=2
  417 CONTINUE
      RETURN
      END


C*********************************************************************
C*********************************************************************
```

```
C*********************************************************************
C.....................................................................
C             SUBROUTINE DATSLT
C     This subroutine gives all the data for integration used in
C     subroutine SLOT.FTN
C.....................................................................
      SUBROUTINE DATSLT(NK0,NK0K,NDP,NSP,NTP,NPOINT,NSER,MPOINT,MA,
     &                  A,E1,E2,CND,CNS,CNT,COAL,BCOAL,FA,ER,
     &                  POINT,BPOINT,XND,XNS,XNT)

      REAL*8 PI,TPI,TPI2
      PARAMETER(PI=3.14159265358979324D0)
      PARAMETER(TPI=2.D0*PI)
      PARAMETER(TPI2=TPI*TPI)

      INTEGER NK0,NK0K,NDP,NSP,NTP,NPOINT,NSER,MPOINT,MA

      REAL*8 A,E1,E2,FA,ER
      REAL*8 A1,A2,A3,B1,B2,B3,B4,R1,S1,R2,S2
      REAL*8 RS1,SS1,RS2,SS2,RS3,RS4,RS5,RS6,RS7,RS8
      REAL*8 RS9,RS10,RS11,RS12,RS13,RS14,RS15,RS16
      REAL*8 TT1,TT2,UU1,UU2

      REAL*8 CND(1),CNS(1),CNT(1),COAL(1),BCOAL(1)
      REAL*8 POINT(1),BPOINT(1),XNS(1)

      REAL*8 XNT(40,3),XND(20,2)

C*********************************************************************
C     +------------------+
C     | ERROR FUNCTIONS  |
C     +------------------+
C
      A1=A*A/ER-TPI2
      A2=TPI2-TPI2/ER
      E1=0.5D0*A2/A1
      E2=ER*E1/(1.D0+ER)
      FA=DSQRT(1.D0+TPI2/A1)
C     +-----------------------------------------------------+
C     |          Data for the poles                         |
C     |  IFIRST= 0 : dominant mode is TM wave (many poles)  |
C     |          1 : dominant mode is TE wave (many poles)  |
C     |          2 : only one TM surface wave               |
C     +-----------------------------------------------------+
C   +---------------------------+
C   |  Data for the Integration |
C   +---------------------------+
      NK0=20
      NK0K=1
      MA=40
      NSER=10
C
      NPOINT=10
C-----------
C     Vector COAL
C-----------
      COAL(1)=0.0666713443D0
      COAL(2)=0.14945134915D0
```

```
      COAL(3)=0.21908636251D0
      COAL(4)=0.26926671931D0
      COAL(5)=0.29552422471D0
      COAL(6)=COAL(5)
      COAL(7)=COAL(4)
      COAL(8)=COAL(3)
      COAL(9)=COAL(2)
      COAL(10)=COAL(1)
C     ------------
C     Vector POINT
C     ------------
      POINT(1)=0.973906528517D0
      POINT(2)=0.865063366688D0
      POINT(3)=0.679409568299D0
      POINT(4)=0.433395394129D0
      POINT(5)=0.148874338981D0
      POINT(6)=-POINT(5)
      POINT(7)=-POINT(4)
      POINT(8)=-POINT(3)
      POINT(9)=-POINT(2)
      POINT(10)=-POINT(1)
C
      MPOINT=5
C     ------------
C     Vector BCOAL
C     ------------
      BCOAL(1)=0.2369268851D0
      BCOAL(2)=0.4786286705D0
      BCOAL(3)=0.5688888888D0
      BCOAL(4)=BCOAL(2)
      BCOAL(5)=BCOAL(1)
C     -------------
C     Vector BPOINT
C     -------------
      BPOINT(1)=0.9061798459D0
      BPOINT(2)=0.5384693101D0
      BPOINT(3)=0.D0
      BPOINT(4)=-BPOINT(2)
      BPOINT(5)=-BPOINT(1)
C     ------------------
C     Single integration
C     ------------------
C
      NSP=31
      RS1=0.99708748181D0
      RS2=0.98468590966D0
      RS3=0.96250392509D0
      RS4=0.93075699789D0
      RS5=0.88976002994D0
      RS6=0.83992032014D0
      RS7=0.78173314841D0
      RS8=0.71577678458D0
      RS9=0.64270672292D0
      RS10=0.56324916140D0
      RS11=0.47819378204D0
      RS12=0.38838590160D0
      RS13=0.29471806998D0
      RS14=0.19812119933D0
```

```
      RS15=0.09955531215D0
      RS16=0.D0
C
      XNS(1)=RS1
      XNS(2)=RS2
      XNS(3)=RS3
      XNS(4)=RS4
      XNS(5)=RS5
      XNS(6)=RS6
      XNS(7)=RS7
      XNS(8)=RS8
      XNS(9)=RS9
      XNS(10)=RS10
      XNS(11)=RS11
      XNS(12)=RS12
      XNS(13)=RS13
      XNS(14)=RS14
      XNS(15)=RS15
      XNS(16)=RS16
      XNS(17)=-RS15
      XNS(18)=-RS14
      XNS(19)=-RS13
      XNS(20)=-RS12
      XNS(21)=-RS11
      XNS(22)=-RS10
      XNS(23)=-RS9
      XNS(24)=-RS8
      XNS(25)=-RS7
      XNS(26)=-RS6
      XNS(27)=-RS5
      XNS(28)=-RS4
      XNS(29)=-RS3
      XNS(30)=-RS2
      XNS(31)=-RS1
C
      CNS(1)=0.0074708315792D0
      CNS(2)=0.0173186207903D0
      CNS(3)=0.0270090191849D0
      CNS(4)=0.0364322739123D0
      CNS(5)=0.0454937075272D0
      CNS(6)=0.0541030824249D0
      CNS(7)=0.0621747865610D0
      CNS(8)=0.0696285832354D0
      CNS(9)=0.0763903865987D0
      CNS(10)=0.0823929917615D0
      CNS(11)=0.0875767406084D0
      CNS(12)=0.0918901138936D0
      CNS(13)=0.0952902429123D0
      CNS(14)=0.0977433353863D0
      CNS(15)=0.0992250112266D0
      CNS(16)=0.0997205447934D0
      CNS(17)=CNS(15)
      CNS(18)=CNS(14)
      CNS(19)=CNS(13)
      CNS(20)=CNS(12)
      CNS(21)=CNS(11)
      CNS(22)=CNS(10)
      CNS(23)=CNS(9)
```

```
      CNS(24)=CNS(8)
      CNS(25)=CNS(7)
      CNS(26)=CNS(6)
      CNS(27)=CNS(5)
      CNS(28)=CNS(4)
      CNS(29)=CNS(3)
      CNS(30)=CNS(2)
      CNS(31)=CNS(1)
C
C
C
C     2)  Double Integration
C         ------------------
C
      NDP=16
      R1=DSQRT((15.D0-2.D0*DSQRT(30.D0))/35.D0)
      R2=-R1
      S1=DSQRT((15.D0+2.D0*DSQRT(30.D0))/35.D0)
      S2=-S1
      A1=4.D0*(59.D0+6.D0*DSQRT(30.D0))/864.D0
      A2=4.D0*(59.D0-6.D0*DSQRT(30.D0))/864.D0
      A3=4.D0*49.D0/864.D0
C
      XND(1,1)=R1
      XND(1,2)=R1
      CND(1)=A1
C
      XND(2,1)=R2
      XND(2,2)=R1
      CND(2)=A1
C
      XND(3,1)=R1
      XND(3,2)=R2
      CND(3)=A1
C
      XND(4,1)=R2
      XND(4,2)=R2
      CND(4)=A1
C
      XND(5,1)=S1
      XND(5,2)=S1
      CND(5)=A2
C
      XND(6,1)=S1
      XND(6,2)=S2
      CND(6)=A2
C
      XND(7,1)=S2
      XND(7,2)=S1
      CND(7)=A2
C
      XND(8,1)=S2
      XND(8,2)=S2
      CND(8)=A2
C
      XND(9,1)=R1
      XND(9,2)=S1
      CND(9)=A3
```

```
C
      XND(10,1)=R1
      XND(10,2)=S2
      CND(10)=A3
C
      XND(11,1)=S1
      XND(11,2)=R1
      CND(11)=A3
C
      XND(12,1)=S2
      XND(12,2)=R1
      CND(12)=A3
C
      XND(13,1)=R2
      XND(13,2)=S1
      CND(13)=A3
C
      XND(14,1)=R2
      XND(14,2)=S2
      CND(14)=A3
C
      XND(15,1)=S1
      XND(15,2)=R2
      CND(15)=A3
C
      XND(16,1)=S2
      XND(16,2)=R2
      CND(16)=A3
C
C     3)  Triple Integration
C         ------------------
C
      NTP=34
      RS1=0.9317380000D0
      RS2=-RS1
      UU1=0.9167441779D0
      UU2=-UU1
      SS1=0.4086003800D0
      SS2=-SS1
      TT1=0.7398529500D0
      TT2=-TT1
      B1=8.D0*0.03558180896D0
      B2=8.D0*0.01247892770D0
      B3=8.D0*0.05286772991D0
      B4=8.D0*0.02672752182D0
C
      XNT(1,1)=RS1
      XNT(1,2)=0.D0
      XNT(1,3)=0.D0
      CNT(1)=B1
C
      XNT(2,1)=RS2
      XNT(2,2)=0.D0
      XNT(2,3)=0.D0
      CNT(2)=B1
C
      XNT(3,1)=0.D0
      XNT(3,2)=RS1
```

```
      XNT(3,3)=0.D0
      CNT(3)=B1
C
      XNT(4,1)=0.D0
      XNT(4,2)=RS2
      XNT(4,3)=0.D0
      CNT(4)=B1
C
      XNT(5,1)=0.D0
      XNT(5,2)=0.D0
      XNT(5,3)=RS1
      CNT(5)=B1
C
      XNT(6,1)=0.D0
      XNT(6,2)=0.D0
      XNT(6,3)=RS2
      CNT(6)=B1
C
      XNT(7,1)=UU1
      XNT(7,2)=UU1
      XNT(7,3)=0.D0
      CNT(7)=B2
C
      XNT(8,1)=UU2
      XNT(8,2)=UU1
      XNT(8,3)=0.D0
      CNT(8)=B2
C
      XNT(9,1)=UU1
      XNT(9,2)=UU2
      XNT(9,3)=0.D0
      CNT(9)=B2
C
      XNT(10,1)=UU2
      XNT(10,2)=UU2
      XNT(10,3)=0.D0
      CNT(10)=B2
C
      XNT(11,1)=UU1
      XNT(11,2)=0.D0
      XNT(11,3)=UU1
      CNT(11)=B2
C
      XNT(12,1)=UU1
      XNT(12,2)=0.D0
      XNT(12,3)=UU2
      CNT(12)=B2
C
      XNT(13,1)=UU2
      XNT(13,2)=0.D0
      XNT(13,3)=UU1
      CNT(13)=B2
C
      XNT(14,1)=UU2
      XNT(14,2)=0.D0
      XNT(14,3)=UU2
      CNT(14)=B2
C
```

```
      XNT(15,1)=0.D0
      XNT(15,2)=UU1
      XNT(15,3)=UU1
      CNT(15)=B2
C
      XNT(16,1)=0.D0
      XNT(16,2)=UU1
      XNT(16,3)=UU2
      CNT(16)=B2
C
      XNT(17,1)=0.D0
      XNT(17,2)=UU2
      XNT(17,3)=UU1
      CNT(17)=B2
C
      XNT(18,1)=0.D0
      XNT(18,2)=UU2
      XNT(18,3)=UU2
      CNT(18)=B2
C
      XNT(19,1)=SS1
      XNT(19,2)=SS1
      XNT(19,3)=SS1
      CNT(19)=B3
C
      XNT(20,1)=SS1
      XNT(20,2)=SS1
      XNT(20,3)=SS2
      CNT(20)=B3
C
      XNT(21,1)=SS1
      XNT(21,2)=SS2
      XNT(21,3)=SS1
      CNT(21)=B3
C
      XNT(22,1)=SS1
      XNT(22,2)=SS2
      XNT(22,3)=SS2
      CNT(22)=B3
C
      XNT(23,1)=SS2
      XNT(23,2)=SS1
      XNT(23,3)=SS1
      CNT(23)=B3
C
      XNT(24,1)=SS2
      XNT(24,2)=SS1
      XNT(24,3)=SS2
      CNT(24)=B3
C
      XNT(25,1)=SS2
      XNT(25,2)=SS2
      XNT(25,3)=SS1
      CNT(25)=B3
C
      XNT(26,1)=SS2
      XNT(26,2)=SS2
      XNT(26,3)=SS2
```

```
      CNT(26)=B3
C
      XNT(27,1)=TT1
      XNT(27,2)=TT1
      XNT(27,3)=TT1
      CNT(27)=B4
C
      XNT(28,1)=TT1
      XNT(28,2)=TT1
      XNT(28,3)=TT2
      CNT(28)=B4
C
      XNT(29,1)=TT1
      XNT(29,2)=TT2
      XNT(29,3)=TT1
      CNT(29)=B4
C
      XNT(30,1)=TT1
      XNT(30,2)=TT2
      XNT(30,3)=TT2
      CNT(30)=B4
C
      XNT(31,1)=TT2
      XNT(31,2)=TT1
      XNT(31,3)=TT1
      CNT(31)=B4
C
      XNT(32,1)=TT2
      XNT(32,2)=TT1
      XNT(32,3)=TT2
      CNT(32)=B4
C
      XNT(33,1)=TT2
      XNT(33,2)=TT2
      XNT(33,3)=TT1
      CNT(33)=B4
C
      XNT(34,1)=TT2
      XNT(34,2)=TT2
      XNT(34,3)=TT2
      CNT(34)=B4
C
      RETURN
      END


C***********************************************************************
C***********************************************************************
C***********************************************************************
C.....................................................................
C.....................................................................
C     This subroutine evaluates the limits of integration in
C     the interval [0,A].
C     Specifically:
C          1) It divides the interval [0,k0] to 10 equal
C             subsections and then apply fixed-point Gaussian
C             Quadrature
C          2) It divides the interval [k0,k] into so many
```

```
C             subsections as the number of poles and in
C             such a way that each subsection includes one
C             pole only away from the ends of the subsection
C          3) It divides the interval [k,A] into 20 equal
C             subsections and then apply fixed-point Gaussian
C             Quadrature
C.....................................................................

      SUBROUTINE LIMIT(NK0,MA,NTM,NTE,NK0K,IFIRST,A,EER,AK0,AK,ER,H,T,
     &                 CN,BM,AM,DM,POLTM,POLTE,POLES,VXXM,VZXM,VZXE)

      INTEGER I,NMAX,NK0,MA,NTM,NTE,NK0K,IFIRST

      REAL*8 A,ER,H,T,EER,AK0,AK,DELTA,ARG,HZXE,GXXM,GZXM

      REAL*8 CN(1),BM(1),AM(1),DM(1),VXXM(1),VZXM(1),VZXE(1)
      REAL*8 POLTM(1),POLTE(1),POLES(1)

C-------------------------------------------------+
C     Step 1 : Evaluation of vector CN            |
C              it gives the end points of the     |
C              intervals considered in (0,k0)     |
C-------------------------------------------------+
      DELTA=AK0/FLOAT(NK0)
      CN(1)=0.D0
      DO 1 I=1,NK0
           CN(I+1)=DELTA*FLOAT(I)
  1   CONTINUE
C-------------------------------------------------+
C     Step 2 : Evaluation of vector BM            |
C              it gives the end points of the     |
C              intervals considered in (k,A)      |
C-------------------------------------------------+
      DELTA=(A/DSQRT(EER)-AK)/FLOAT(MA)
      BM(1)=AK
      DO 2 I=1,MA
           BM(I+1)=DELTA*FLOAT(I)+AK
  2   CONTINUE
C-------------------------------------------------+
C     Step 3 : Evaluation of the vectors AM,DM    |
C              "AM" gives the end points around   |
C                   the TM poles                  |
C              "DM" gives the end points around   |
C                   the TE poles                  |
C                                                 |
C     IFIRST=   2   only one TM pole              |
C               1   TE0<TM0                       |
C               0   TM0<TE0                       |
C-------------------------------------------------+
      AM(1)=AK0
      DM(1)=AK0
      NMAX=NTE+NTM-1
      IF (IFIRST.EQ.2) GO TO 3
      DO 4 I=1,NMAX
           AM(I+1)=(POLES(I+1)+POLES(I))/2.D0
           DM(I+1)=AM(I+1)
   4  CONTINUE
      AM(NMAX+2)=AK
```

```
      DM(NMAX+2)=AK
      IF (IFIRST.EQ.1) GO TO 5
          DM(NMAX+1)=AM(NMAX+2)
          DO 6 I=1,NMAX
               DM(NMAX-I+1)=AM(NMAX-I+2)
  6       CONTINUE
          GO TO 7
  5       AM(NMAX+1)=DM(NMAX+2)
          DO 8 I=1,NMAX
               AM(NMAX-I+1)=DM(NMAX-I+2)
  8       CONTINUE
          GO TO 7
C
  3       DELTA=(AK-AK0)/FLOAT(NK0K)
          AM(1)=AK0
          DO 9 I=1,NK0K
               AM(I+1)=DELTA*FLOAT(NK0K)+AK0
  9       CONTINUE
  7       CONTINUE
C-----------------------------------------------+
C     Step 4 : evaluation of vectors VZXE       |
C-----------------------------------------------+
      IF (IFIRST.EQ.2) GO TO 10
          DO 11 I=1,NTE
                ARG=POLTE(I)
                VZXE(I)=HZXE(ARG,ER,H,T,AK0,AK)
  11      CONTINUE
  10  CONTINUE
C-----------------------------------------------+
C     Step 5 : evaluation of vector VXXM,VZXM   |
C-----------------------------------------------+
      DO 12 I=1,NTM
            ARG=POLTM(I)
            VXXM(I)=GXXM(ARG,ER,H,T,AK0,AK)
            VZXM(I)=GZXM(ARG,ER,H,T,AK0,AK)
  12  CONTINUE
      RETURN
      END


C*********************************************************************
C*********************************************************************
C*********************************************************************
C.....................................................................
C             Functions :  GXXM,GZXM,HZXE
C
C     These functions evaluate the residues from the different poles
C.....................................................................
      FUNCTION GXXM(X,ER,H,T,AK0,AK)

      REAL*8 GXXM,X,X2,ER,H,T,AK0,AK,RM,RMH,RM0,RM0H,RMT,SXN,SXD

      X2=X*X
      RM=DSQRT(AK*AK-X2)
      RM0=DSQRT(X2-AK0*AK0)
      RMH=RM*H
      RM0H=RM0*H
      RMT=RM*(-H+T)
```

```
      SXN=RM*DCOS(RMT)-ER*RM0*DSIN(RMT)
      SXD=(ER+RM0H)*(RM/RM0)*DCOS(RMH)+(1.D0+ER*RM0H)*DSIN(RMH)
      GXXM=SXN/SXD
      RETURN
      END
C
C.....................................................................
C
      FUNCTION GZXM(X,ER,H,T,AK0,AK)

      REAL*8 GZXM,X,X2,ER,H,T,AK0,AK,RM,RMH,RM0,RM0H,CSH,SNH,SXN,SXD

      X2=X*X
      RM=DSQRT(AK*AK-X2)
      RM0=DSQRT(X2-AK0*AK0)
      RMH=RM*H
      RM0H=RM0*H
      CSH=DCOS(RMH)
      SNH=DSIN(RMH)
      SXN=RM*DCOS(RM*T)
      SXD=(RM*CSH+RM0*SNH)*((ER+RM0H)*CSH/RM0+(1.D0+ER*RM0H)*SNH/RM)
      GZXM=SXN/SXD
      RETURN
      END
C
C.....................................................................
C
      FUNCTION HZXE(X,ER,H,T,AK0,AK)

      REAL*8 HZXE,X,X2,ER,H,T,AK0,AK,RM,RMH,RM0,CSH,SNH,SXN,SXD

      X2=X*X
      RM=DSQRT(AK*AK-X2)
      RM0=DSQRT(X2-AK0*AK0)
      RMH=RM*H
      CSH=DCOS(RMH)
      SNH=DSIN(RMH)
      SXN=RM*DCOS(RM*T)
      SXD=(ER*RM0*CSH-RM*SNH)*(1.D0+RM0*H)*(SNH/RM0-CSH/RM)
      HZXE=SXN/SXD
      RETURN
      END


C**********************************************************************
C**********************************************************************
C**********************************************************************
C.....................................................................
C.....................................................................
C     This subroutine evaluates the values of the integrand of
C     the Green's function at different points in the interval
C     [0,A]. Then it evaluetes the space integrals of the Bessel
C     function at the same points and multiply these values with
C     the corresponding values of the Green's function.
C     Finally , it multiplies these products with known coeffic.
C     and it adds them up. This way, the moments'-method
C     space integrals of the first part of the Green's function are
C     evaluated and are stored in the complex vectors ZS,ZS1S2
```

```
C.....................................................................
C.....................................................................

      SUBROUTINE GGREEN(ER,H,T,E1,E2,FA,COAL,BCOAL,
     &                  CN,AM,BM,DM,MA,MPOINT,NPOINT,POINT,NK0,NTM,NTE,
     &                  NK0K,IFIRST,INS,NOFF,NSER,NMAX,DLX,W,AK,AKK,AK0,
     &                  OFFSET,OFFLIM,BPOINT,DIST,RCOE,AX,SERS,SERA,
     &                  S,WREAL,POLTM,POLTE,NS,
     &                  VXXM,VZXM,VZXE,YCOS,YSD,FSD,SSJ0,SAJ0,SLTDIM)


      INTEGER SLTDIM
      INTEGER I,N,NTTM,MA,INS,NS,NOFF,MPOINT,NSER,NMAX(7)
      INTEGER INCON,IAD,IFD,NPOINT,NK0,NTM,NTE,NK0K,IFIRST

      REAL*8 DLX,W,H,T,FA,AK,AK0,AKK,OFFLIM
      REAL*8 AUP,ALOW,AX,WREAL
      REAL*8 AI,TI,ER,E1,E2
      REAL*8 F1X,F1Z,YCOS

      REAL*8 AM(1),BM(1),DM(1),BPOINT(1),OFFSET(1),SERS(1),SERA(1)
      REAL*8 COAL(1),BCOAL(1),POINT(1),CN(1)
      REAL*8 POLTE(1),POLTM(1),VZXE(1),VXXM(1),VZXM(1)
      REAL*8 FSD(1)
      COMPLEX*16 YSD(1)

      REAL*8 DARG(10,4),S(10,2)
      REAL*8 DIST(250,7,10),RCOE(20,250,7,10)
      REAL*8 SSJ0(SLTDIM,7),SAJ0(SLTDIM,7)

C-------------------------------------------------+
C     Evaluation of the coefficients for the      |
C           FF's functions                        |
C-------------------------------------------------+
      F1X=1.0
      F1Z=2.D0*(1.D0-ER)/((1.D0+ER)*(1.D0+E2)*(1.D0+0.5D0*E1))
      IF ((ER-1.D0).LT.0.005) F1Z=0.D0
C
      CALL ARIS(NOFF,MPOINT,NPOINT,NSER,NMAX,DLX,W,AKK,
     &          OFFSET,OFFLIM,POINT,BPOINT,DIST,RCOE,AX,
     &          SERS,SERA,DARG,S,WREAL,SLTDIM)
C
      DO 1 I=1,NPOINT
            INCON=I
            AI=COAL(I)
            TI=POINT(I)
C
C     evaluation of intervals 1 and 2
C
            IAD=1
            DO 2 N=1,NK0
                 AUP=CN(N+1)
                 ALOW=CN(N)
                      CALL FUNCT(IAD,AUP,ALOW,N,INCON,NS,NSER,NOFF,NMAX,
     &                   MPOINT,NPOINT,IFIRST,INS,ER,H,T,AX,DLX,W,
     &                   AI,AK0,AK,AKK,TI,DARG,DIST,RCOE,POLTM,POLTE,
     &                   VXXM,VZXM,VZXE,YCOS,COAL,BCOAL,OFFLIM,OFFSET,
     &                   SERS,SERA,S,FSD,FA,F1X,F1Z,YSD,SSJ0,SAJ0,SLTDIM)
```

```
   2        CONTINUE
C
C     evaluation of intervals 3 and 4
C
            NTTM=NTM
            IF (IFIRST.EQ.2) NTTM=NK0K
            DO 3 IAD=3,4
                 IFD=0
                 DO 4 N=1,NTTM
                      IFD=IFD+1
                      AUP=AM(IFD+1)
                      ALOW=AM(IFD)
                      CALL FUNCT(IAD,AUP,ALOW,N,INCON,NS,NSER,NOFF,NMAX,
     &                   MPOINT,NPOINT,IFIRST,INS,ER,H,T,AX,DLX,W,
     &                   AI,AK0,AK,AKK,TI,DARG,DIST,RCOE,POLTM,POLTE,
     &                   VXXM,VZXM,VZXE,YCOS,COAL,BCOAL,OFFLIM,OFFSET,
     &                   SERS,SERA,S,FSD,FA,F1X,F1Z,YSD,SSJ0,SAJ0,SLTDIM)

                      IFD=IFD+1
   4             CONTINUE
   3        CONTINUE
            IF (IFIRST.EQ.2) GO TO 9
C
C     evaluation of the intervals 5 and 6,9,11
C
                 DO 5 IAD=5,6
                 IFD=0
                 DO 6 N=1,NTE
                      IFD=IFD+1
                      AUP=DM(IFD+1)
                      ALOW=DM(IFD)
                      CALL FUNCT(IAD,AUP,ALOW,N,INCON,NS,NSER,NOFF,NMAX,
     &                   MPOINT,NPOINT,IFIRST,INS,ER,H,T,AX,DLX,W,
     &                   AI,AK0,AK,AKK,TI,DARG,DIST,RCOE,POLTM,POLTE,
     &                   VXXM,VZXM,VZXE,YCOS,COAL,BCOAL,OFFLIM,OFFSET,
     &                   SERS,SERA,S,FSD,FA,F1X,F1Z,YSD,SSJ0,SAJ0,SLTDIM)

                      IFD=IFD+1
   6             CONTINUE
   5        CONTINUE
   9        CONTINUE
C
C     evaluation of the interval 7
C
            IAD=7
            DO 7 N=1,MA
                 AUP=BM(N+1)
                 ALOW=BM(N)
                 CALL FUNCT(IAD,AUP,ALOW,N,INCON,NS,NSER,NOFF,NMAX,
     &                MPOINT,NPOINT,IFIRST,INS,ER,H,T,AX,DLX,W,
     &                AI,AK0,AK,AKK,TI,DARG,DIST,RCOE,POLTM,POLTE,
     &                VXXM,VZXM,VZXE,YCOS,COAL,BCOAL,OFFLIM,OFFSET,
     &                SERS,SERA,S,FSD,FA,F1X,F1Z,YSD,SSJ0,SAJ0,SLTDIM)

  7         CONTINUE
  1   CONTINUE
C
```

```
C
C     evaluation of the intervals 8,10
C
      IAD=8
      IFD=0
      DO 8 N=1,NTM
           IFD=IFD+1
           AUP=AM(IFD+1)
           ALOW=AM(IFD)
           CALL FUNCT(IAD,AUP,ALOW,N,INCON,NS,NSER,NOFF,NMAX,
     &                MPOINT,NPOINT,IFIRST,INS,ER,H,T,AX,DLX,W,
     &                AI,AK0,AK,AKK,TI,DARG,DIST,RCOE,POLTM,POLTE,
     &                VXXM,VZXM,VZXE,YCOS,COAL,BCOAL,OFFLIM,OFFSET,
     &                SERS,SERA,S,FSD,FA,F1X,F1Z,YSD,SSJ0,SAJ0,SLTDIM)


           IFD=IFD+1
   8  CONTINUE
      RETURN
      END



C**********************************************************************
C**********************************************************************
C**********************************************************************
C.....................................................................
C                              ARIS
C.....................................................................
      SUBROUTINE ARIS(NOFF,MPOINT,NPOINT,NSER,NMAX,DLX,W,AKK,
     &                OFFSET,OFFLIM,POINT,BPOINT,DIST,RCOE,AX,
     &                SERS,SERA,DARG,S,WREAL,SLTDIM)

      REAL*8 PI
      PARAMETER(PI=3.14159265358979324D0)

      INTEGER SLTDIM
      INTEGER NOFF,MPOINT,NPOINT,NSER,NMAX(1)
      INTEGER MAX,I,J,N,LPOINT,JN

      REAL*8 DLX,W,AKK,OFFLIM,WREAL
      REAL*8 AC,AS,AX,BX,X,W2,U,U1,POIN,YCOS,YSIN,FI
      REAL*8 ADL,ADL2,ADL3,ADL4,ADL5,ADL6,AXN,AXN2,BXN,BXN2
      REAL*8 DSIG,DSIG2,DSIG3,DSIG4,DSIG5,DSIG6,DSIG7,DSIG8
      REAL*8 S2,S3,ST,SX,SIG,SIG2,SIG3,SER1,SER2,SER3,SER4,SER5
      REAL*8 S5M1,S7M1,S5M3,S7M3,S7M5,THETA,THMIN,THMAX
      REAL*8 G21,G22,G41,G42,G43,G44,G61,G62,G63,G64,G65,G66
      REAL*8 G81,G82,G83,G84,G85,G86,G87,G88
      REAL*8 S30,S31,S33,S51,S53,S55,S71,S73,S75,S77

      REAL*8 POINT(1),BPOINT(1),OFFSET(1),SERS(1),SERA(1)

      REAL*8 DARG(10,4),S(10,2)
      REAL*8 DIST(SLTDIM,7,10),RCOE(20,SLTDIM,7,10)

C     +----------------------------------------+
C     | Formation of the matrices:  DIST,      |
C     |     DARG,RCOE                          |
```

```
C     |                                          |
C     +------------------------------------------+
      W2=W/2.D0
      U=WREAL/W
      THMIN=DATAN(DSQRT(1.D0/(U*U)-1.D0))
      THMAX=PI-THMIN
      AX=(THMAX-THMIN)/2.D0
      BX=(THMAX+THMIN)/2.D0
      X=PI/4.D0
      DO 1 J=1,NOFF
           MAX=NMAX(J)
           LPOINT=MPOINT
           IF (OFFSET(J).LE.OFFLIM) LPOINT=NPOINT
           DO 2 I=1,LPOINT
                POIN=BPOINT(I)
                IF (OFFSET(J).LE.OFFLIM) POIN=POINT(I)
                FI=X*(POIN+1.D0)
                THETA=AX*POIN+BX
                AS=DSIN(FI)
                AC=DCOS(FI)
                DARG(I,1)=W2*AC
                DARG(I,2)=AC
                DARG(I,3)=AS
                DARG(I,4)=X
                DO 3 N=1,MAX
                     AXN=FLOAT(N-2)*DLX
                     IF (OFFSET(J).GT.OFFLIM) GO TO 4
                        DIST(N,J,I)=AXN*AS
                        GO TO 5
  4                  AXN2=AXN*AXN
                     BXN=OFFSET(J)-W*DCOS(THETA)/2.D0
                     BXN2=BXN*BXN
                     DIST(N,J,I)=DSQRT(AXN2+BXN2)
                     SIG=DIST(N,J,I)
                     SIG2=SIG*SIG
                     SIG3=SIG2*SIG
                DSIG=DABS(AXN)/SIG
                DSIG2=BXN2/SIG3
                DSIG3=-3.D0*DSIG*DSIG2/SIG
                DSIG4=-3.D0*DSIG2*(DSIG2-4.D0*DSIG**2/SIG)/SIG
                DSIG5=-3.D0*(-15.D0*DSIG2**2*DSIG+(20.D0/SIG)*
     *               DSIG2*DSIG**3)/SIG2
                DSIG6=-3.D0*(-15.D0*DSIG2**3+(180.D0/SIG)*DSIG2
     *               **2*DSIG**2-(120.D0/SIG2)*DSIG2*DSIG**4)/
     *               SIG2
                DSIG7=-3.D0*(525.D0*DSIG2**3*DSIG-(2100.D0/SIG)*
     *               DSIG2**2*DSIG**3+(840.D0/SIG2)*DSIG2*DSIG
     *               **5)/SIG3
                DSIG8=-3.D0*(525.D0*DSIG2**4-(12600.D0/SIG)*DSIG2
     *               **3*DSIG**2+(25200.D0/SIG2)*DSIG2**2*DSIG**4
     *               -(6720.D0/SIG3)*DSIG2*DSIG**6)/SIG3
C
C          Evaluation of the coefficients Gij
C
                G21=DSIG2
                G22=DSIG**2
C----------------
                G41=DSIG4
```

```
                G42=4.D0*DSIG3*DSIG+3.D0*DSIG2**2
                G43=6.D0*DSIG2*DSIG**2
                G44=DSIG**4
C---------------
                G61=DSIG6
                G62=6.D0*DSIG5*DSIG+15.D0*DSIG4*DSIG2+10.D0*DSIG3**2
                G63=15.D0*DSIG4*DSIG**2+60.D0*DSIG3*DSIG2*DSIG+15.D0
     *              *DSIG2**3
                G64=20.D0*DSIG3*DSIG**3+45.D0*DSIG2**2*DSIG**2
                G65=15.D0*DSIG2*DSIG**4
                G66=DSIG**6
C---------------
                G81=DSIG8
                G82=8.D0*DSIG7*DSIG+28.D0*DSIG6*DSIG2+56.D0*DSIG5
     *              *DSIG3+35.D0*DSIG4**2
                G83=28.D0*DSIG6*DSIG**2+168.D0*DSIG5*DSIG2*DSIG+
     *              280.D0*DSIG4*DSIG3*DSIG+210.D0*DSIG4*DSIG2**2+
     *              280.D0*DSIG3**2*DSIG2
                G84=56.D0*DSIG5*DSIG**3+420.D0*DSIG4*DSIG2*DSIG**2
     *              +280.D0*DSIG3**2*DSIG**2+840.D0*DSIG3*DSIG2**2
     *              *DSIG+105.D0*DSIG2**4
                G85=70.D0*DSIG4*DSIG**4+560.D0*DSIG3*DSIG2*DSIG**3
     *              +420.D0*DSIG2**3*DSIG**2
                G86=56.D0*DSIG3*DSIG**5+210.D0*DSIG2**2*DSIG**4
                G87=28.D0*DSIG2*DSIG**6
                G88=DSIG**8
C---------------
                RCOE(2,N,J,I)=-0.5D0*(G22+SIG*G21)
                RCOE(1,N,J,I)=0.5D0*(G22-SIG*G21)
C---------------
                SX=0.5D0*SIG*(G42-SIG*G41)
                S30=-0.5D0*SIG*(G42+SIG*G41)
                S31=0.25D0*(SX+3.D0*G43)
                S33=0.25D0*(SX-G43)
                RCOE(3,N,J,I)=0.5D0*(SIG*S33/3.D0+G44/4.D0)
                RCOE(4,N,J,I)=0.5D0*(SIG*S31+SIG*S33/3.D0-G44)
                RCOE(5,N,J,I)=0.5D0*(SIG*S31+3.D0*G44/4.D0)
                RCOE(6,N,J,I)=SIG*S30
C---------------
                SX=SIG*S33/3.D0+G64/4.D0
                ST=SIG*S31+SIG*S33/3.D0-G64
                S5M3=SIG2*S30
                S5M1=0.5D0*SIG*(SIG*S31+3.D0*G64/4.D0)
                S51=0.25D0*(0.5D0*SIG*ST-5.D0*G65/2.D0)
                S53=0.25D0*(0.5D0*SIG*ST+0.25D0*SIG*SX+0.5D0*G65/
     *              4.D0)
                S55=0.125D0*(0.5D0*SIG*SX-0.5*G65)
                RCOE(7,N,J,I)=0.5D0*(SIG*S55/5.D0+G66/16.D0)
                RCOE(8,N,J,I)=0.5D0*(SIG*S53/3.D0+SIG*S55/5.D0-
     *                       6.D0*G66/16.D0)
                RCOE(9,N,J,I)=0.5D0*(SIG*S51+SIG*S53/3.D0+15.D0*
     *                       G66/16.D0)
                RCOE(10,N,J,I)=0.5D0*(SIG*S51-10.D0*G66/16.D0)
                RCOE(11,N,J,I)=SIG*S5M1
                RCOE(12,N,J,I)=SIG*S5M3
C---------------
                S7M5=SIG2*S5M3
                S7M3=SIG2*S5M1
```

```
                S7M1=0.5D0*SIG*(SIG*S51-10.D0*G86/16.D0)
                S71=0.5D0*(0.25D0*SIG*(SIG*S51+SIG*S53/3.D0+
     *              15.D0*G86/16.D0)+35.D0*G87/32.D0)
                S73=0.5D0*(0.25D0*SIG*(SIG*S51+SIG*S53/3.D0+15.D0
     *              *G86/16.D0)+0.125D0*SIG*(SIG*S53/3.D0+SIG*
     *              S55/5.D0-6.D0*G86/16.D0)-21.D0*G87/32.D0)
                S75=0.5D0*(0.125D0*SIG*(SIG*S53/3.D0+SIG*S55/5.D0-
     *              6.D0*G86/16.D0)+(SIG/12.D0)*(SIG*S55/5.D0+
     *              G86/16.D0)+7.D0*G87/32.D0)
                S77=0.5D0*((SIG/12.D0)*(SIG*S55/5.D0+G86/16.D0)-
     *              G87/32.D0)
                RCOE(13,N,J,I)=0.5D0*(SIG*S77/7.D0+G88/64.D0)
                RCOE(14,N,J,I)=0.5D0*(SIG*S75/5.D0+S77*SIG/7.D0
     *                         -8.D0*G88/64.D0)
                RCOE(15,N,J,I)=0.5D0*(SIG*S73/3.D0+SIG*S75/5.D0
     *                         +28.D0*G88/64.D0)
                RCOE(16,N,J,I)=0.5D0*(SIG*S71+SIG*S73/3.D0-56.D0
     *                         *G88/64.D0)
                RCOE(17,N,J,I)=0.5D0*(SIG*S71+35.D0*G88/64.D0)
                RCOE(18,N,J,I)=SIG*S7M1
                RCOE(19,N,J,I)=SIG*S7M3
                RCOE(20,N,J,I)=SIG*S7M5
   5                CONTINUE
   3            CONTINUE
   2         CONTINUE
   1  CONTINUE
C
C               Formation of the series s(dlx) . Storage in
C                    vectors SERS(5),SERA(5)
C
      U1=2.D0*THMIN/FLOAT(NSER)
      DO 6 JN=1,NSER
           S2=(2.D0*FLOAT(JN)-1.D0)
           S2=S2/(2.D0*FLOAT(NSER))
           S3=DCOS(S2*THMIN)
           S(JN,2)=S3*W/2.D0
           S(JN,1)=U1
  6   CONTINUE
      ADL=AKK*DLX
      ADL2=ADL*ADL
      ADL3=ADL2*ADL
      ADL4=ADL3*ADL
      ADL5=ADL4*ADL
      ADL6=ADL5*ADL
      YSIN=DSIN(ADL)
      YCOS=DCOS(ADL)
C
      SER1=(1.D0-YCOS)*2.D0/AKK
C
      SER2=-YSIN/3.D0+ADL*YCOS/4.D0+ADL2*YSIN/10.D0-ADL3*YCOS/36.D0
     *     -ADL4*YSIN/168.D0+ADL5*YCOS/960.D0+ADL6*YSIN/6480.D0
C
      SER3=YSIN/60.D0-ADL*5.D0*YCOS/360.D0-ADL2*YSIN/168.D0+ADL3
     *     *YCOS/560.D0+ADL4*YSIN/2592.D0-ADL5*YCOS/12960.D0-ADL6
     *     *YSIN/95040.D0
C
      SER4=-YSIN/2520.D0+ADL*YCOS/2880.D0+ADL2*YSIN/6480.D0-ADL3
     *     *YCOS/21600.D0-ADL4*YSIN/95040.D0+ADL5*YCOS/518400.D0
```

```
C
      SER5=YSIN/181440.D0-ADL*YCOS/201600.D0-ADL2*YSIN/443520.D0+
     *     ADL3*YCOS/1442775.9D0
C
      SERS(1)=SER1*SER1
      SERS(2)=DLX*2.D0*SER1*SER2
      SERS(3)=DLX*(DLX*SER2*SER2+2.D0*SER1*SER3)
      SERS(4)=DLX*(2.D0*SER1*SER4+2.D0*DLX*SER2*SER3)
      SERS(5)=DLX*(DLX*SER3*SER3+2.D0*DLX*SER2*SER4)
C
      SERA(1)=SER1
      SERA(2)=DLX*SER2
      SERA(3)=DLX*SER3
      SERA(4)=DLX*SER4
      SERA(5)=DLX*SER5
      RETURN
      END


C**********************************************************************
C**********************************************************************
C**********************************************************************
C.....................................................................
C     1) This subroutine evaluates the integrand of the Green's
C        function at different points (subroutine Grei).
C     2) It evaluates the space integrals coming from the
C        application of moments' method (subroutine adonis)
C     3) Multiply these two values with appropriate weighting
C        coefficients and it adds them up
C.....................................................................

      SUBROUTINE FUNCT(IAD,AUP,ALOW,N,INCON,NS,NSER,NOFF,NMAX,
     &                 MPOINT,NPOINT,IFIRST,INS,ER,H,T,AX,DLX,W,
     &                 AI,AK0,AK,AKK,TI,DARG,DIST,RCOE,POLTM,POLTE,
     &                 VXXM,VZXM,VZXE,YCOS,COAL,BCOAL,OFFLIM,OFFSET,
     &                 SERS,SERA,S,FSD,FA,F1X,F1Z,YSD,SSJ0,SAJ0,SLTDIM)

      REAL*8 PI
      PARAMETER(PI=3.14159265358979324D0)
      COMPLEX*16 CI
      PARAMETER(CI=(0.D0,1.D0))

      INTEGER SLTDIM
      INTEGER NS,NPOINT,IFIRST,INS,NSER,NOFF,MPOINT,NMAX(1)
      INTEGER INCON,IAD,K,N,NCON

      REAL*8 S1,S2,AX,DLX,W,YCOS,OFFLIM,ALI,AUP,AIMA,ALOW
      REAL*8 ER,H,T,FA,F1X,F1Z,PLI,AI,AK0,AK,AKK,AK2,AK02,AKK2,TI
      REAL*8 RX,RZ,XX,XZ,FRX,FRZ,GXXR,GZXR,GXXX,GZXX
      REAL*8 X,Y,XTE,XTM,VARX,VARZ,TM,TMTE,TMTM
      REAL*8 ER1,FS1,FARX,FARZ,FCONX,FCONZ,FXXR,FZXR
      REAL*8 GCONX,GCONZ,GCONX1,GCONX2,GCONZ1,GCONZ2

      REAL*8 ARG(1),COAL(1),BCOAL(1),OFFSET(1),SERS(1),SERA(1)
      REAL*8 POLTM(1),POLTE(1),VXXM(1),VZXM(1),VZXE(1),FSD(1)
      COMPLEX*16 YSD(1)

      REAL*8 S(10,2),DARG(10,4)
```

```
      REAL*8 DIST(SLTDIM,7,10),RCOE(20,SLTDIM,7,10)
      REAL*8 SSJ0(SLTDIM,7),SAJ0(SLTDIM,7)

C***********************************************************************

      NCON=0
      X=AUP-ALOW
      Y=AUP+ALOW
      AK02=AK0*AK0
      AK2=AK*AK
      AKK2=AKK*AKK
      ER1=1.D0-ER
      IF (IAD.GT.2) GO TO 1
         ALI=0.5D0*(TI*X+Y)
         GCONX=AI*X*0.5D0
         FCONX=GCONX
         GCONZ=GCONX*ER1
         IF(DABS(ER1).LT.0.005) GCONZ=0.D0
         FCONZ=FCONX
         AIMA=1.D0
         CALL GREI(ALI,0.D0,0.D0,IAD,0.D0,ER,H,T,AK0,AK,FA,
     &             RX,XX,RZ,XZ,FRX,FRZ,F1X,F1Z)
         GO TO 10
 1    IF (IAD.NE.3) GO TO 2
         ALI=0.5D0*(TI*X+Y)
         XTM=POLTM(N)
         TMTM=(2.D0*XTM-Y)/X
         GCONX=AI/(TI-TMTM)
         GCONZ=GCONX*ER1
         FCONX=AI*X*0.5D0
         FCONZ=FCONX
         AIMA=0.D0
         IF (DABS(ER1).LT.0.005) THEN
                GCONX=0.0
                GCONZ=0.0
                FCONX=0.0
                FCONZ=0.0
         END IF
         CALL GREI(ALI,XTM,0.D0,IAD,0.D0,ER,H,T,AK0,AK,FA,
     &             RX,XX,RZ,XZ,FRX,FRZ,F1X,F1Z)
         GO TO 10
 2    IF (IAD.NE.4) GO TO 3
         ALI=POLTM(N)
         TM=(2.D0*ALI-Y)/X
         GCONX=-AI/(TI-TM)
         GCONZ=GCONX*ER1
         FCONX=0.D0
         FCONZ=0.D0
         AIMA=0.D0
         RX=VXXM(N)
         RZ=VZXM(N)
         IF (DABS(ER1).LT.0.005) THEN
                GCONX=0.0
                GCONZ=0.0
                FCONX=0.0
                FCONZ=0.0
         END IF
```

```
        GO TO 10
 3    IF (IFIRST.EQ.2) GO TO 5
      IF (IAD.NE.5) GO TO 4
         ALI=0.5D0*(TI*X+Y)
         XTE=POLTE(N)
         TMTE=(2.D0*XTE-Y)/X
         GCONX=AI*X*0.5D0
         GCONZ=AI*ER1/(TI-TMTE)
         FCONX=GCONX
         FCONZ=FCONX
         AIMA=0.D0
         IF (DABS(ER1).LT.0.005) THEN
                GCONX=0.0
                GCONZ=0.0
                FCONX=0.0
                FCONZ=0.0
         END IF
         CALL GREI(ALI,0.D0,XTE,IAD,TMTE,ER,H,T,AK0,AK,FA,
     &             RX,XX,RZ,XZ,FRX,FRZ,F1X,F1Z)

         GO TO 10
 4    IF (IAD.NE.6) GO TO 5
         NCON=6
         ALI=POLTE(N)
         TM=(2.D0*ALI-Y)/X
         GCONX=0.D0
         GCONZ=-AI*ER1/(TI-TM)
         FCONX=0.D0
         FCONZ=0.D0
         AIMA=0.D0
         RZ=VZXE(N)
         IF (DABS(ER1).LT.0.005) THEN
                GCONX=0.0
                GCONZ=0.0
                FCONX=0.0
                FCONZ=0.0
         END IF
         GO TO 10
 5    IF (IAD.NE.7) GO TO 6
         ALI=0.5D0*(TI*X+Y)
         GCONX=AI*X*0.5D0
         GCONZ=GCONX*ER1
         IF(DABS(ER1).LT.0.005) GCONZ=0.0
         FCONX=GCONX
         FCONZ=FCONX
         AIMA=0.D0
         CALL GREI(ALI,0.D0,0.D0,IAD,0.D0,ER,H,T,AK0,AK,FA,
     &             RX,XX,RZ,XZ,FRX,FRZ,F1X,F1Z)

         GO TO 10
 6       NCON=8
         ALI=POLTM(N)
         TM=(2.D0*ALI-Y)/X
         FCONX=0.D0
         FCONZ=0.D0
         AIMA=0.D0
         RX=VXXM(N)
         RZ=VZXM(N)
```

```
         GO TO 28
C
  10  CONTINUE
         GXXR=GCONX*RX
         FXXR=FCONX*FRX
         GXXX=AIMA*GCONX*XX
         GZXR=GCONZ*RZ
         FZXR=FCONZ*FRZ
         GZXX=AIMA*GCONZ*XZ
  27  CONTINUE
         VARX=(AK2-AKK2)*GXXR+AKK2*GZXR
         FARX=(AK2-AKK2)*FXXR+AKK2*FZXR
         VARZ=AKK*(GXXR-GZXR)
         FARZ=AKK*(FXXR-FZXR)
         GXXR=VARX
         FXXR=FARX
         GZXR=VARZ
         FZXR=FARZ
         VARX=(AK2-AKK2)*GXXX+AKK2*GZXX
         VARZ=AKK*(GXXX-GZXX)
         GXXX=VARX
         GZXX=VARZ
      PLI=ALI
C
      CALL ADONIS(NSER,NOFF,MPOINT,NPOINT,NMAX,DARG,DIST,RCOE,
     &            AX,DLX,W,PLI,YCOS,COAL,BCOAL,OFFLIM,OFFSET,
     &            SERS,SERA,S,SSJ0,SAJ0,SLTDIM)

      DO 13 K=1,NS
C             S1  = REAL(GXXR*SSJ0(K,INS)+GZXR*SAJ0(K,INS))
C             FS1 = REAL(FXXR*SSJ0(K,INS)+FZXR*SAJ0(K,INS))
C             S2  = REAL(GXXX*SSJ0(K,INS)+GZXX*SAJ0(K,INS))
              S1  = GXXR*SSJ0(K,INS)+GZXR*SAJ0(K,INS)
              FS1 = FXXR*SSJ0(K,INS)+FZXR*SAJ0(K,INS)
              S2  = GXXX*SSJ0(K,INS)+GZXX*SAJ0(K,INS)
              YSD(K) = YSD(K) + S1 - CI * S2
              FSD(K) = FSD(K) + FS1
C
C             IF (K.EQ.1) THEN
C                 WRITE (6,966) NCON,IAD,ALI
C 966             FORMAT(10X,'NCON=',I4,2X,'IAD=',I4,2X,
C     *           'ALI=',E14.7/)
C                 WRITE (6,866) FXXR,FZXR,FS1
C 866             FORMAT(10X,'FXXR=',E14.7,2X,'FZXR=',E14.7,
C     *           2X,'FS1=',E14.7/)
C                 WRITE (6,766) YSD(K),FSD(K)
C 766             FORMAT(10X,'YSD=',E14.7,1X,E14.7,2X,'FSD=',E14.7//)
C             END IF
C
  13  CONTINUE
  28  IF (NCON.EQ.0) GO TO 24
      IF(INCON.LT.NPOINT) GO TO 24
          GCONX1=0.0
          GCONX2=0.0
          GCONZ1=ER1*DLOG((1.D0-TM)/(1.D0+TM))
          GCONZ2=ER1*PI
          IF (NCON.EQ.6) GO TO 29
              GCONX1=DLOG((1.D0-TM)/(1.D0+TM))
```

```
            GCONX2=PI
 29       CONTINUE
            GXXR=GCONX1*RX
            GXXX=GCONX2*RX
            GZXR=GCONZ1*RZ
            GZXX=GCONZ2*RZ
            FXXR=0.0
            FZXR=0.0
          IF (DABS(ER1).LT.0.005) THEN
              GXXR=0.0
              GXXX=0.0
              GZXR=0.0
              GZXX=0.0
          END IF
          NCON=0
          GO TO 27
 24       CONTINUE
          RETURN
          END



C**********************************************************************
C**********************************************************************
C**********************************************************************
C.....................................................................
C     This subroutine evaluates the integrand of the green's
C     function at different points
C.....................................................................
      SUBROUTINE GREI(X,XFM,XFE,IAD,TM,ER,H,T,AK0,AK,FA,
     &                RX,XX,RZ,XZ,FRX,FRZ,F1X,F1Z)

      INTEGER IAD

      REAL*8 TM,ER,H,T,AK0,AK,FA,RX,XX,RZ,XZ,FRX,FRZ,F1X,F1Z
      REAL*8 AK2,AK02,C1,C2,C3,DEN,EX,EXX,EXZ,ERM0,ERM02
      REAL*8 RM,RM0,RM02,RM2,RMH,RMT,RMHT
      REAL*8 SNH,SNT,SNHT,CSH,CST,CSH2,CSHH
      REAL*8 SNHHT,CSHT,CSHHT,TANT,TANH,TANHT
      REAL*8 X,X2,XFE,XFM,RNOM,XNOM

C**********************************************************************

      X2=X*X
      AK2=AK*AK
      AK02=AK0*AK0
      RM=DSQRT(DABS(AK2-X2))
      RM0=DSQRT(DABS(X2-AK02))
      RMH=RM*H
      RMT=RM*T
      RMHT=RM*(-H+T)
C
      CSH=DCOS(RMH)
      SNH=DSIN(RMH)
      CST=DCOS(RMT)
      SNT=DSIN(RMT)
      CSHT=DCOS(RMHT)
      SNHT=DSIN(RMHT)
```

```
C
      RM2=RM*RM
      RM02=RM0*RM0
      CSH2=CSH*CSH
      ERM0=ER*RM0
      ERM02=ERM0*ERM0
C
      EXX=DEXP(-X*T/FA)/FA
      EXZ=DEXP(-X*2.D0*H/FA)/FA
      IF (IAD.NE.7) GO TO 100
          EX=DEXP(RMH)
          TANH=(EX-1.D0/EX)/(EX+1.D0/EX)
          CSHH=(EX+1.D0/EX)/2.D0
          EX=DEXP(RMT)
          CSHT=0.5D0*(EX+1.D0/EX)
          SNHT=0.5D0*(EX-1.D0/EX)
          TANT=SNHT/CSHT
          EX=DEXP(RMHT)
          CSHHT=0.5D0*(EX+1.D0/EX)
          SNHHT=0.5D0*(EX-1.D0/EX)
          TANHT=SNHHT/CSHHT
C
  100 IF (IAD.NE.1) GO TO 1
          DEN=RM2+(ERM02-RM2)*CSH2
          RNOM=-RM2*SNT+(RM2-ERM02)*CSH*SNHT
          XNOM=ER*RM*RM0*CST
          C1=X/RM
C
          RX=C1*RNOM/DEN
          IF((ER-1.D0).LT.0.005) RX=0.D0
          XX=C1*XNOM/DEN
          FRX=F1X*EXX
C
          DEN=DEN*(RM02+AK02*(ER-1.D0)*CSH2)
          RNOM=-CST*(RM2+ER*RM02)*CSH*SNH
          XNOM=CST*RM*RM0*(-1.D0+(1.D0+ER)*CSH2)
          C1=X*RM
          RZ=-C1*RNOM/DEN
          XZ=C1*XNOM/DEN
          FRZ=F1Z*EXZ
          RETURN
    1   IF (IAD.NE.3) GO TO 2
          C1=X-XFM
          IF (DABS(AK-X).LT.1.D-6) GO TO 10
              DEN=ERM0*CSH-RM*SNH
              RNOM=(RM*CSHT-ERM0*SNHT)
              C2=X/RM
              RX=C1*C2*RNOM/DEN
C
              DEN=DEN*(RM*CSH+RM0*SNH)
              RNOM=CST
              C3=X*RM
              RZ=C1*C3*RNOM/DEN
C
              FRX=F1X*EXX
              FRZ=F1Z*EXZ
              RETURN
C
```

```
   10     RNOM=1.D0-ERM0*(-H+T)
          RX=C1*X*RNOM/ERM0
          FRX=F1X*EXX
C
          RZ=X*C1/(ERM0*(1.D0+RM0*H))
          FRZ=F1Z*EXZ
          RETURN
  2    IF (IAD.NE.5) GO TO 4
          C1=X-XFE
          IF (DABS(AK-X).LT.1.D-6) GO TO 13
              RNOM=RM*CSHT-ERM0*SNHT
              DEN=ERM0*CSH-RM*SNH
              RX=(X/RM)*RNOM/DEN
              FRX=F1X*EXX
C
              RNOM=RM*CST
              DEN=DEN*(RM*CSH+RM0*SNH)
              RZ=X*C1*RNOM/DEN
              FRZ=F1Z*EXZ
              RETURN
   13     RX=X*(1.D0-ERM0*(-H+T))/ERM0
          FRX=F1X*EXX
          RZ=X*C1/(ERM0*(1.D0+RM0*H))
          FRZ=F1Z*EXZ
          RETURN
  4    IF (IAD.NE.7) GO TO 6
          IF (DABS(X-AK).LT.1.D-6) GO TO 15
              DEN=ERM0+RM*TANH
              RNOM=(RM+ERM0*TANH)*CSHT-DEN*SNHT
              RX=(X/RM)*RNOM/DEN
              FRX=F1X*EXX
C
              RNOM=X*(RM*CSHT)/(CSHH*CSHH)
              DEN=DEN*(RM+RM0*TANH)
              RZ=RNOM/DEN
              FRZ=F1Z*EXZ
              RETURN
  15  RX=X*(1.D0-ERM0*(-H+T))/ERM0
      FRX=F1X*EXX
      RZ=(X/ERM0)/(1.D0+RM0*H)
      FRZ=F1Z*EXZ
  6   CONTINUE
      RETURN
      END


C*********************************************************************
C*********************************************************************
C*********************************************************************
C.....................................................................
C                        ADONIS
C     This subroutine evaluates the space integarls of the bessel
C         function
C
C.....................................................................
      SUBROUTINE ADONIS(NSER,NOFF,MPOINT,NPOINT,NMAX,DARG,DIST,RCOE,
     &                  AX,DLX,W,PLI,YCOS,COAL,BCOAL,OFFLIM,OFFSET,
     &                  SERS,SERA,S,SSJ0,SAJ0,SLTDIM)
```

```
      REAL*8 PI
      PARAMETER(PI=3.14159265358979324D0)

      INTEGER SLTDIM
      INTEGER NSER,NOFF,MPOINT,NPOINT,NMAX(1)
      INTEGER MAX,I,J,K,N,LPOINT,KMAX,KJ,JN,NK

      REAL*8 AARG
      REAL*8 AX,DLX,W,PLI,YCOS,OFFLIM
      REAL*8 ARX,ARG2,ARG4,ARG6,AROF,ARAF,COFF,CAFF
      REAL*8 CH1,CH2,CH3,CH4,CH5,PR1,PR2,PR4,PR6,PR8
      REAL*8 ASIN,SIN1,COS1,SIN2,TERM,W1,SUMS,SSUM,SUMA

      REAL*8 SERS(1),SERA(1),COAL(1),BCOAL(1),OFFSET(1),ARG(1)
      REAL*8 BJ(10,2),DERIV(9,3),DARG(10,4),S(10,2)
      REAL*8 DIST(SLTDIM,7,10),RCOE(20,SLTDIM,7,10)
      REAL*8 SSJ0(SLTDIM,7),SAJ0(SLTDIM,7)

C**********************************************************************

      ARX=W*AX/2.D0
      W1=2.D0*YCOS
      PR1=PLI*DLX
      PR2=PR1*PR1
      PR4=PR2*PR2
      PR6=PR4*PR2
      PR8=PR6*PR2
      DO 1 J=1,NOFF
           MAX=NMAX(J)
           DO 2 N=1,MAX
                SSJ0(N,J)=0.D0
                SAJ0(N,J)=0.D0
  2        CONTINUE
  1   CONTINUE
C
      DO 11 J=1,NOFF
            LPOINT=MPOINT
            IF (OFFSET(J).GT.OFFLIM) GO TO 12
                LPOINT=NPOINT
                DO 13 I=1,NPOINT
                      ARG(I)=PLI*DARG(I,1)
  13            CONTINUE
                CALL BESS1(BJ,NPOINT,ARG)
  12        DO 14 I=1,LPOINT
                  DO 17 NK=1,5
                        DERIV(NK,1)=0.D0
                        DERIV(NK,2)=0.D0
  17              CONTINUE
                  ASIN=ARX*BCOAL(I)
                  IF (OFFSET(J).GT.OFFLIM) GO TO 15
                        ASIN=W*DARG(I,4)*COAL(I)
                        AROF=PLI*OFFSET(J)*DARG(I,2)
                        COFF=DCOS(AROF)
                        SSUM=0.D0
                        DO 16 JN=1,NSER
                              ARAF=PLI*S(JN,2)*DARG(I,2)
                              CAFF=DCOS(ARAF)
```

```
                        SSUM=SSUM+S(JN,1)*CAFF
  16                CONTINUE
  15              CONTINUE
                  KMAX=NMAX(J)
                  DO 18 K=1,KMAX
                        DO 20 NK=1,5
                              DERIV(NK,1)=DERIV(NK,2)
                              DERIV(NK,2)=DERIV(NK,3)
  20                  CONTINUE
                      IF (OFFSET(J).GT.OFFLIM) GO TO 21
                              SIN1=DARG(I,3)
                              SIN2=SIN1*SIN1
                              COS1=DCOS(PLI*DIST(K,J,I))
                              TERM=COFF*(BJ(I,1)-SSUM/PI)*COS1
                              DERIV(1,3)=TERM
                              SIN1=SIN2
                              DERIV(2,3)=-TERM*SIN1
                              SIN1=SIN1*SIN2
                              DERIV(3,3)=TERM*SIN1
                              SIN1=SIN1*SIN2
                              DERIV(4,3)=-TERM*SIN1
                              SIN1=SIN1*SIN2
                              DERIV(5,3)=TERM*SIN1
                              GO TO 22
  21                  AARG=PLI*DIST(K,J,I)
                      ARG2=AARG*AARG
                      ARG4=ARG2*ARG2
                      ARG6=ARG4*ARG2
                      CALL BESS2(BJ,AARG)
                      DERIV(1,3)=BJ(1,2)
                      DERIV(2,3)=RCOE(1,K,J,I)*BJ(3,2)+
     *                           RCOE(2,K,J,I)*BJ(1,2)
                      DERIV(3,3)=RCOE(3,K,J,I)*BJ(5,2)+
     *                           RCOE(4,K,J,I)*BJ(3,2)+(RCOE(5,K,J,I)
     *                           +RCOE(6,K,J,I)/ARG2)*BJ(1,2)
                      DERIV(4,3)=RCOE(7,K,J,I)*BJ(7,2)+
     *                           RCOE(8,K,J,I)*BJ(5,2)+RCOE(9,K,J,I)*
     *                           BJ(3,2)+(RCOE(10,K,J,I)+RCOE(11,K,
     *                           J,I)/ARG2+RCOE(12,K,J,I)/ARG4)*
     *                           BJ(1,2)
                      DERIV(5,3)=RCOE(13,K,J,I)*BJ(9,2)+
     *                           RCOE(14,K,J,I)*BJ(7,2)+RCOE(15,K,J,
     *                           I)*BJ(5,2)+RCOE(16,K,J,I)*BJ(3,2)+
     *                           (RCOE(17,K,J,I)+RCOE(18,K,J,I)/ARG2
     *                           +RCOE(19,K,J,I)/ARG4+RCOE(20,K,J,I)
     *                           /ARG6)*BJ(1,2)
  22                  IF (K.LT.3) GO TO 18
                      SUMS=SERS(1)*DERIV(1,2)-PR2*SERS(2)*DERIV(2,2)
     *                     +PR4*SERS(3)*DERIV(3,2)-PR6*SERS(4)*DERIV
     *                     (4,2)+PR8*SERS(5)*DERIV(5,2)
C
                      CH1=SERA(1)*(DERIV(1,1)+DERIV(1,3)-W1*DERIV
     *                    (1,2))
                      CH2=SERA(2)*(DERIV(2,1)+DERIV(2,3)-W1*DERIV
     *                    (2,2))*PR2
                      CH3=SERA(3)*(DERIV(3,1)+DERIV(3,3)-W1*DERIV
     *                    (3,2))*PR4
                      CH4=SERA(4)*(DERIV(4,1)+DERIV(4,3)-W1*DERIV
```

```
     *                    (4,2))*PR6
                       CH5=SERA(5)*(DERIV(5,1)+DERIV(5,3)-W1*DERIV
     *                    (5,2))*PR8
                       SUMA=CH1-CH2+CH3-CH4+CH5
                       KJ=K-2
                       SSJ0(KJ,J)=SSJ0(KJ,J)+ASIN*SUMS
                       SAJ0(KJ,J)=SAJ0(KJ,J)+ASIN*SUMA
CCCC
C                      IF (KJ.EQ.1)WRITE (6,665) KJ,J,SSJ0(KJ,J),
C    *                 SUMS,SAJ0(KJ,J),SUMA
C665                   FORMAT(10X,'KJ=',I4,2X,'J=',I4/10X,'SSJ0=',
C    *                 E14.7,2X,'SUMS=',E14.7/10X,'SAJ0=',E14.7,
C    *                 2X,'SUMA=',E14.7/)
CCCC
  18              CONTINUE
  14         CONTINUE
  11    CONTINUE
        RETURN
        END


C***********************************************************************
C***********************************************************************
C***********************************************************************
C.....................................................................
C                              BESS1
C     This subroutine gives values for the zeroth order
C     Bessel functions. It is used for small offsets
C.....................................................................
      SUBROUTINE BESS1(BJ,NPOINT,ARG)

      REAL*8 PI
      PARAMETER(PI=3.14159265358979324D0)

      INTEGER NPOINT,IJ

      REAL*8 BJ0,FJ0,TJ0,WCON
      REAL*8 X,X3,X32,X33,X34,X35,X36,X38,X310,X312

      REAL*8 ARG(1),BJ(10,2)

C***********************************************************************

      DO 1 IJ=1,NPOINT
           X=ARG(IJ)
           IF (X.GT.0.001D0) GO TO 10
              X3=X/3.D0
              X32=X3*X3
              X34=X32*X32
              X36=X34*X32
              BJ0=1.D0-2.2499997D0*X32+1.2656208D0*X34-0.3163866D0
     *           *X36
              BJ(IJ,1)=BJ0
              GO TO 1
  10       IF (X.GT.3.D0) GO TO 12
              X3=X/3.D0
              X32=X3*X3
              X34=X32*X32
```

```
              X36=X34*X32
              X38=X36*X32
              X310=X38*X32
              X312=X310*X32
              BJ0=1.D0-2.2499997D0*X32+1.2656208D0*X34-0.3163866D0
     *            *X36+0.0444479D0*X38-0.0039444D0*X310+0.00021000
     *            D0*X312
              BJ(IJ,1)=BJ0
              GO TO 1
   12       CONTINUE
            X3=3.D0/X
            X32=X3*X3
            X33=X32*X3
            X34=X33*X3
            X35=X34*X3
            X36=X35*X3
            FJ0=0.79788456D0-0.00000077D0*X3-0.00552740D0*X32-0.0000
     *          9512D0*X33+0.00137237D0*X34-0.00072805D0*X35+0.00014
     *          476D0*X36
            TJ0=X-0.78539816D0-0.04166397D0*X3-0.00003954D0*X32+0.00
     *          262573D0*X33-0.00054125D0*X34-0.00029333D0*X35+0.000
     *          13558D0*X36
            WCON=DSQRT(1.D0/X)
            BJ(IJ,1)=WCON*FJ0*DCOS(TJ0)
   1        CONTINUE
         RETURN
         END


C.....................................................................
C   This subroutine evaluates the higher order bessel functions using
C   the ascenting series expression or hankel's expansion.
C.....................................................................
      SUBROUTINE BESS2(BJ,X)

      REAL*8 PI
      PARAMETER(PI=3.14159265358979324D0)

      INTEGER I,J,JN,N,NB,NJ1,NJ2,NJB,NJB1,NJB2,NCON,NIMAX,NJMAX

      REAL*8 X,BJ0,BJ1
      REAL*8 A,R1,R2,XA,XE,BN1,BN2,BN3,BN4,CTH,TNH

      REAL*8 U(4),BJ(10,2),RBJ(50,2)
C.....................................................................
C
C     Evaluation of J0,J1
C
      CALL BSJ0(X,BJ0,BJ1)
      RBJ(1,2)=BJ0
      RBJ(2,2)=BJ1
C
      NCON=1
      N=IDINT(2.4D0*X)
      IF (N.LT.10) N=10
      IF (X.LT.3.D0) GO TO 10
C
C     EVALUATION OF HIGHER ORDER BESSEL FUNCTIONS UP TO
```

```
C     ORDER LESS THEN THE ARGUMENT
C
      NIMAX=IDINT(X)-1
      IF (NIMAX.GT.9) NIMAX=9
      DO 1 I=2,NIMAX
           NJ1=I
           NJ2=I-1
           NB=I+1
           RBJ(NB,2)=FLOAT(2*NJ2)*RBJ(NJ1,2)/X-RBJ(NJ2,2)
  1   CONTINUE
      IF (NIMAX.EQ.9) GO TO 20
      NCON=NIMAX
C
C     DEBYE'S ASYMPTOTIC EXPANSION-EVALUATION OF JN
C
  10  DO 11 J=1,2
            JN=N-J+1
            XA=X/FLOAT(JN)
            XA=1.D0/XA
            XE=XA+DSQRT(XA*XA-1.D0)
            A=DLOG(XE)
            CTH=(XE+1.D0/XE)/(XE-1.D0/XE)
            CALL F(CTH,U)
            TNH=1.D0/CTH
            R1=DEXP(FLOAT(JN)*(TNH-A))
            R2=DSQRT(2.D0*PI*FLOAT(JN)*TNH)
            BN1=JN
            BN2=JN*JN
            BN3=BN2*JN
            BN4=BN3*JN
            RBJ(JN+1,2)=(R1/R2)*(1.D0+U(1)/BN1+U(2)/BN2+U(3)/BN3+
     *                  U(4)/BN4)
  11  CONTINUE
C
C     EVALUATION OF HIGHER ORDER BESSEL FUNCTIONS WHEN X<10
C
      NJMAX=N-2-NCON
      DO 2 I=1,NJMAX
           NJB=N-I
           NJB1=NJB+1
           NJB2=NJB1+1
           RBJ(NJB,2)=2.D0*FLOAT(NJB)*RBJ(NJB1,2)/X-RBJ(NJB2,2)
  2   CONTINUE
  20  CONTINUE
      DO 3 I=1,9
           BJ(I,2)=RBJ(I,2)
  3   CONTINUE
      RETURN
      END

C.....................................................................
C.....................................................................
      SUBROUTINE BSJ0(X,BJ0,BJ1)

      REAL*8 PI
      PARAMETER(PI=3.14159265358979324D0)

      REAL*8 BJ0,BJ1,TJ0,TJ1,FJ0,FJ1,WCON
```

```
      REAL*8 X,X3,X32,X33,X34,X35,X36,X38,X310,X312
C
C    Evaluation of J0 using the series expansion given in
C    Abramowitz.
C
C      PI=3.141592653589D0
      IF (X.GT.3.D0) GO TO 20
      X3=X/3.D0
      X32=X3*X3
      X34=X32*X32
      X36=X32*X34
      X38=X32*X36
      X310=X38*X32
      X312=X310*X32
      BJ0=1.D0-2.2499997D0*X32+1.2656208D0*X34-0.3163866D0*X36+
     *    0.0444479D0*X38-0.0039444D0*X310+0.00021000D0*X312
      BJ1=X*(0.5D0-0.56249985D0*X32+0.21093573D0*X34-0.03954289D0
     *    *X36+0.00443319D0*X38-0.00031761D0*X310+0.00001109D0
     *    *X312)
      GO TO 21
C
   20 X3=3.D0/X
      X32=X3*X3
      X33=X32*X3
      X34=X33*X3
      X35=X34*X3
      X36=X35*X3
      FJ0=0.79788456D0-0.00000077D0*X3-0.00552740D0*X32-0.00009512D0
     *    *X33+0.00137237D0*X34-0.00072805D0*X35+0.00014476D0*X36
      FJ1=0.79788456D0+0.00000156D0*X3+0.01659667D0*X32+0.00017105D0
     *    *X33-0.00249511D0*X34+0.00113653D0*X35-0.00020033D0*X36
      TJ0=X-0.78539816D0-0.04166397D0*X3-0.00003954D0*X32+0.00262573D0
     *    *X33-0.00054125D0*X34-0.00029333D0*X35+0.00013558D0*X36
      TJ1=X-2.35619449D0+0.12499612D0*X3+0.00005650D0*X32-0.00637879D0
     *    *X33+0.00074348D0*X34+0.00079824D0*X35-0.00029166D0*X36
      WCON=DSQRT(1.D0/X)
      BJ0=WCON*FJ0*DCOS(TJ0)
      BJ1=WCON*FJ1*DCOS(TJ1)
  21  CONTINUE
      RETURN
      END



C**********************************************************************
C**********************************************************************
C**********************************************************************
C.....................................................................
C.....................................................
      SUBROUTINE F(X,U)

      REAL*8 U(1),X,X2,X3,X4,X5,X6,X7,X8,X9,X10,X11,X12

      X2=X*X
      X3=X2*X
      X4=X3*X
      X5=X4*X
      X6=X5*X
```

```
      X7=X6*X
      X8=X7*X
      X9=X8*X
      X10=X9*X
      X11=X10*X
      X12=X11*X
C
      U(1)=(3.D0*X-5.D0*X3)/24.D0
      U(2)=(81.D0*X2-462.D0*X4+385.D0*X6)/1152.D0
      U(3)=(30375.D0*X3-369603.D0*X5+765765.D0*X7-425425.D0*X9)/
     *      414720.D0
      U(4)=(4465125.D0*X4-94121676.D0*X6+349922430.D0*X8-446185740.D0*
     *      X10+185910725.D0*X12)/39813120.D0
      RETURN
      END


C*********************************************************************
C*********************************************************************
C*********************************************************************
C.....................................................................
C                              TAIL
C     This subroutine evaluates the tail contribution
C.....................................................................
      SUBROUTINE TAIL(NS,NS1S2,NOFF,NDP,NTP,NSP,NMAX,INS,INS1S2,
     &                S1,T1,T2,T3,T4,D1,D2,GS,AX,WREAL,
     &                ER,H,T,DLX,W,E1,E2,AK0,AK,AKK,FA,OFFSET,
     &                XNS,XND,XNT,CNS,CND,CNT,SLTDIM)

      REAL*8 PI
      PARAMETER(PI=3.14159265358979324D0)

      INTEGER SLTDIM
      INTEGER I,J,K,N,M0,N0,JK,JM,KMAX,NJMAX,MAX(8,2)
      INTEGER MM1,MM2,NP1,NM1,MP1,MP2
      INTEGER NS,NS1S2,NOFF,NDP,NTP,NSP,NMAX(1),INS,INS1S2

      REAL*8 AX,WREAL,ER,H,T,DLX,W,E1,E2,AK0,AK,AKK,FA
      REAL*8 AASIN,AZ,AK2,AK02,AKK2,AAX,AAZ,ATX,ATZ
      REAL*8 COEF1,COEF2,CW,C1,C2,CST1,CST2,CST3,CST4
      REAL*8 CXX,CZX,CSX,CSZ,CY1,CY2,CCS,CAX,CAZ
      REAL*8 DSP,DTP,DDP,DLX2,DLX4,W2,PI2,PI4
      REAL*8 RAD2,RADM2,RADP2,RADPP2,RADMM2,RADMP2,RADPM2
      REAL*8 SONM1,SONM2,SONP1,SONP2,SINP1,SINP2
      REAL*8 SY1,SY2,SV1,SV2,SV3,SV4,STX,STZ,SSN
      REAL*8 SANM1,SANM2,SANP1,SANP2,SINM1,SINM2
      REAL*8 TAMM1,TAMM2,TAPM1,TAPM2,TAMP1,TAMP2,TAPP1,TAPP2
      REAL*8 TRAM1,TRAM2,TRAP1,TRAP2,TRAD1,TRAD2
      REAL*8 THI,THMAX,THMIN
      REAL*8 XI,XN,XA1,XA2,XAP,XAM,XNM,XNP,XNMM,XNMP,XNPM,XNPP,XIP
      REAL*8 Y1,Y2,YSIN,YCOS,SIN0
      REAL*8 ZW,Z1,Z2,ZP1,ZP2,ZP12,ZP22

      REAL*8 OFFSET(1),XNS(1),CNS(1),CND(1),CNT(1),GS(1)
      REAL*8 XND(20,2),XNT(40,3)
      REAL*8 T1(4,SLTDIM,7),T2(4,SLTDIM,7),T3(4,SLTDIM,7),T4(4,SLTDIM,7)
      REAL*8 S1(4,SLTDIM,7),D1(4,SLTDIM,7),D2(4,SLTDIM,7)
```

```
C*****************************************************************************
C
C     This vector contains the values of t in the integrals h0
C
      Z1=T
      Z2=2.D0*H
C
C     This vector contains the values of the coefficient C in
C          the integrals h0
C
      C1=FA
C      WRITE (*,111) FA
C 111  FORMAT(///10X,'FA=',E14.7///)
C
C      This vector contains the number of elements of the
C      matrices ZS,ZS1S2,....
C
      MAX(1,1)=NS
      MAX(2,1)=NS1S2
C
      MAX(1,2)=INS
      MAX(2,2)=INS1S2
C
C
C     This vector contains the values of the coefficient A in
C               the integrals h0
C
      AK2=AK*AK
      AKK2=AKK*AKK
      AK02=AK0*AK0
      W2=W/2.D0
      THMIN=WREAL/W
      THMIN=DATAN(DSQRT(1.D0/THMIN**2-1.D0))
      THMAX=PI-THMIN
      PI2=PI/2.D0
      PI4=PI/4.D0
      DLX2=DLX/2.D0
      DLX4=DLX2*DLX2
C
      YCOS=DCOS(AKK*DLX)
      CCS=DCOS(2.D0*AKK*DLX)
      YSIN=DSIN(AKK*DLX)
      SSN=DSIN(2.D0*AKK*DLX)
C
C     +---------------------------------+
C     | Evaluation of S1,S2,S3,S4,S5,S6 |
C     |       (Single Integrals)        |
C     +---------------------------------+
C
C
      DO 201 J=1,7
             DO 202 K=1,205
                    DO 203 JK=1,4
                           S1(JK,K,J)=0.D0
                           D1(JK,K,J)=0.D0
                           D2(JK,K,J)=0.D0
                           T1(JK,K,J)=0.D0
                           T2(JK,K,J)=0.D0
```

```
                        T3(JK,K,J)=0.D0
                        T4(JK,K,J)=0.D0
  203               CONTINUE
  202         CONTINUE
  201 CONTINUE
C
      ZP1=Z1/C1
      ZP2=Z2/C1
C
      ZP12=ZP1*ZP1
      ZP22=ZP2*ZP2
      DO 1 J=1,NOFF
           KMAX=NMAX(J)+2
           IF (OFFSET(J).LT.1.D-6) THMAX=PI
           DSP=(THMAX-THMIN)/4.D0
           DDP=DSP*DLX2
           DTP=DSP*DLX4
           COEF1=(THMAX-THMIN)/2.D0
           IF (OFFSET(J).LT.1.D-6) COEF1=(PI/2.D0-THMIN)/2.D0
           COEF2=(THMAX+THMIN)/2.D0
           IF (OFFSET(J).LT.1.D-6) COEF2=(PI/2.D0+THMIN)/2.D0
           DO 10 I=1,NSP
                THI=COEF1*XNS(I)+COEF2
                C1=DCOS(THI)
                C2=W2*C1
                C2=OFFSET(J)-C2
                CW=C2*C2
                AASIN=CNS(I)*DSP
                DO 11 K=1,KMAX
                     XN=(FLOAT(K-3)*DLX)
                     RAD2=XN*XN+CW
                     TRAD1=DSQRT(RAD2+ZP12)
                     TRAD2=DSQRT(RAD2+ZP22)
                     S1(1,K,J)=S1(1,K,J)+DLOG(2.D0*(TRAD1+XN))*AASIN
                     S1(2,K,J)=S1(2,K,J)+DLOG(2.D0*(TRAD2+XN))*AASIN
  11            CONTINUE
  10       CONTINUE
C
C     +-------------------------------------------------------------+
C     | EVALUATION OF D1,D2,D4,D5                                   1
C     +-------------------------------------------------------------+
           DO 20 I=1,NDP
                THI=COEF1*XND(I,1)+COEF2
                XI=DLX2*(XND(I,2)+1.D0)
                C1=DCOS(THI)
                C2=W2*C1
                C2=OFFSET(J)-C2
                CW=C2*C2
                AASIN=CND(I)*DDP
                SV1=DSIN(AKK*(DLX-XI))
                SV2=-SV1
                SV4=DSIN(AKK*XI)
                C2=DCOS(AKK*(DLX-XI))
                DO 21 K=1,KMAX
                     XNP=(XI+FLOAT(K-2)*DLX)
                     XNM=(-XI+FLOAT(K-2)*DLX)
                     RADP2=XNP*XNP+CW
                     RADM2=XNM*XNM+CW
```

```
                    TRAP1=DSQRT(RADP2+ZP12)
                    TRAP2=DSQRT(RADP2+ZP22)
C
                    TRAM1=DSQRT(RADM2+ZP12)
                    TRAM2=DSQRT(RADM2+ZP22)
C
                    XA1=AKK*XNP
                    XA2=AKK*XNM
                    XAP=DSIN(XA1)
                    XAM=DSIN(XA2)
C
                    SANP1=XAP*DLOG(2.D0*(TRAP1+XNP))
                    SANP2=XAP*DLOG(2.D0*(TRAP2+XNP))
C
                    SANM1=XAM*DLOG(2.D0*(TRAM1+XNM))
                    SANM2=XAM*DLOG(2.D0*(TRAM2+XNM))
C
                    XAP=DSIN(XA1/2.D0)
                    XAM=DSIN(XA2/2.D0)
                    SONP1=XAP/TRAP1
                    SONP2=XAP/TRAP2
C
                    SONM1=XAM/TRAM1
                    SONM2=XAM/TRAM2
C
                    Y1=-XNM/2.D0-DLX
                    Y2=-XNP/2.D0+DLX
                    CY1=DCOS(AKK*Y1)
                    CY2=DCOS(AKK*Y2)
                    SY1=DSIN(AKK*Y1)
                    SY2=DSIN(AKK*Y2)
C
                    D1(1,K,J)=D1(1,K,J)+(SANP1+SANM1)*SV2*AASIN
                    D2(1,K,J)=D2(1,K,J)+(CY1*SONP1-CY2*SONM1)*AASIN
                    D1(2,K,J)=D1(2,K,J)+(SANP2+SANM2)*SV2*AASIN
                    D2(2,K,J)=D2(2,K,J)+(CY1*SONP2-CY2*SONM2)*AASIN
  21            CONTINUE
  20        CONTINUE
C
C       evaluation of T1,T2,T3,T4
C
            DO 30 I=1,NTP
                  THI=COEF1*XNT(I,1)+COEF2
                  XI=DLX2*(XNT(I,2)+1.D0)
                  XIP=DLX2*(XNT(I,3)+1.D0)
                  C1=DCOS(THI)
                  C2=W2*C1
                  C2=OFFSET(J)-C2
                  CW=C2*C2
                  SV1=DSIN(AKK*(DLX-XI))
                  SV2=-SV1
                  SV3=DSIN(AKK*(DLX-XIP))
                  AASIN=DTP*CNT(I)
                  DO 31 K=1,KMAX
                        XNPP=(XI+XIP)+FLOAT(K-1)*DLX
                        XNPM=(XI-XIP)+FLOAT(K-1)*DLX
                        XNMP=(-XI+XIP)+FLOAT(K-1)*DLX
                        XNMM=(-XI-XIP)+FLOAT(K-1)*DLX
```

```
                    RADPP2=XNPP*XNPP+CW
                    RADPM2=XNPM*XNPM+CW
                    RADMP2=XNMP*XNMP+CW
                    RADMM2=XNMM*XNMM+CW
                    TAPP1=DSQRT(RADPP2+ZP12)
                    TAPP2=DSQRT(RADPP2+ZP22)
                    TAPM1=DSQRT(RADPM2+ZP12)
                    TAPM2=DSQRT(RADPM2+ZP22)
                    TAMP1=DSQRT(RADMP2+ZP12)
                    TAMP2=DSQRT(RADMP2+ZP22)
                    TAMM1=DSQRT(RADMM2+ZP12)
                    TAMM2=DSQRT(RADMM2+ZP22)
                    CST1=DCOS(AKK*(XNPM/2.D0+DLX))*DSIN(AKK*XNPP
     *                   /2.D0)
                    CST2=DCOS(AKK*(-XNMP/2.D0+DLX))*DSIN(AKK*XNMM
     *                   /2.D0)
                    CST3=DCOS(AKK*(XNMM/2.D0+DLX))*DSIN(AKK*XNMP
     *                   /2.D0)
                    CST4=DCOS(AKK*(-XNPP/2.D0+DLX))*DSIN(AKK*XNPM
     *                   /2.D0)
                    T1(1,K,J)=T1(1,K,J)+SV2*AASIN*CST1/TAPP1
                    T2(1,K,J)=T2(1,K,J)+SV1*AASIN*CST2/TAMM1
                    T3(1,K,J)=T3(1,K,J)+SV1*AASIN*CST3/TAMP1
                    T4(1,K,J)=T4(1,K,J)+SV2*AASIN*CST4/TAPM1
                    T1(2,K,J)=T1(2,K,J)+SV2*AASIN*CST1/TAPP2
                    T2(2,K,J)=T2(2,K,J)+SV1*AASIN*CST2/TAMM2
                    T3(2,K,J)=T3(2,K,J)+SV1*AASIN*CST3/TAMP2
                    T4(2,K,J)=T4(2,K,J)+SV2*AASIN*CST4/TAPM2
  31             CONTINUE
  30         CONTINUE
   1     CONTINUE
C
C
C       Evaluation of GS,GS1S2
C
C
        CZX=2.D0*(1.D0-ER)/((1.D0+ER)*(1.D0+E2)*(1.D0+0.5D0*E1))
        IF((ER-1.D0).LT.0.005) CZX=0.D0
        CXX=1.D0
        CSX=(AK2-AKK2)*CXX/FA
        CSZ=AKK2*CZX/FA
        CAX=AKK*CXX/FA
        CAZ=AKK*CZX/FA
        DO 4 JM=1,NOFF
             NJMAX=MAX(JM,1)
             J=MAX(JM,2)
             DO 62 N=1,NJMAX
                  NP1=N+2
                  N0=N+1
                  NM1=N
                  STX=-D1(1,NP1,J)+2.D0*YCOS*D1(1,N0,J)-D1(1,NM1,J)
     *               +2.D0*(T1(1,N,J)+T2(1,N,J)-T3(1,N,J)-T4(1,N,J))
                  STZ=-D1(2,NP1,J)+2.D0*YCOS*D1(2,N0,J)-D1(2,NM1,J)
     *               +2.D0*(T1(2,N,J)+T2(2,N,J)-T3(2,N,J)-T4(2,N,J))
                  MP2=N+4
                  MP1=N+3
                  M0=N+2
                  MM1=N+1
```

```
                MM2=N
                SINP2=DSIN(AKK*FLOAT(N+1)*DLX)
                SINP1=DSIN(AKK*FLOAT(N)*DLX)
                SIN0=DSIN(AKK*FLOAT(N-1)*DLX)
                SINM1=DSIN(AKK*FLOAT(N-2)*DLX)
                SINM2=DSIN(AKK*FLOAT(N-3)*DLX)
                ATX=SINP2*S1(1,MP2,J)-4.D0*YCOS*SINP1*S1(1,MP1,J)
     *              +2.D0*(2.D0+CCS)*SIN0*S1(1,M0,J)-4.D0*YCOS
     *              *SINM1*S1(1,MM1,J)+SINM2*S1(1,MM2,J)
                ATZ=SINP2*S1(2,MP2,J)-4.D0*YCOS*SINP1*S1(2,MP1,J)
     *              +2.D0*(2.D0+CCS)*SIN0*S1(2,M0,J)-4.D0*YCOS
     *              *SINM1*S1(2,MM1,J)+SINM2*S1(2,MM2,J)
                AAX=-2.D0*(D2(1,NP1,J)-2.D0*YCOS*D2(1,N0,J)
     *              +D2(1,NM1,J))
                AAZ=-2.D0*(D2(2,NP1,J)-2.D0*YCOS*D2(2,N0,J)
     *              +D2(2,NM1,J))
                AX=ATX+AAX
                AZ=ATZ+AAZ
                ZW=W*(CSX*STX+CSZ*STZ+CAX*AX-CAZ*AZ)
               GS(N)=ZW
   62       CONTINUE
    4 CONTINUE
      RETURN
      END




CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C     THE FOLLOWING CODES ARE FROM THE NAAS SUBROUTINE LIBRARY
C     CONVERTED TO DOUBLE PRECISION (DESPITE WHAT COMMENTS MIGHT SAY)
C     ( USED QUICK METHOD REAL ----> REAL*8
C                         COMPLEX ------> COMPLEX*16
C                         SQRT    ------> DSQRT            ETC.
C       NOT SURE IF INFALLIBLE BUT GOT SAME RESULT FOR PARTICULAR CASE. )
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C NAASA  2.1.042 CGECO    FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
      SUBROUTINE CGECO(A,LDA,N,IPVT,RCOND,Z)
      INTEGER LDA,N,IPVT(1)
      COMPLEX*16 A(LDA,1),Z(1)
      REAL*8 RCOND
C
C     CGECO FACTORS A COMPLEX*16 MATRIX BY GAUSSIAN ELIMINATION
C     AND ESTIMATES THE CONDITION OF THE MATRIX.
C
C     IF  RCOND  IS NOT NEEDED, CGEFA IS SLIGHTLY FASTER.
C     TO SOLVE  A*X = B , FOLLOW CGECO BY CGESL.
C     TO COMPUTE  INVERSE(A)*C , FOLLOW CGECO BY CGESL.
C     TO COMPUTE  DETERMINANT(A) , FOLLOW CGECO BY CGEDI.
C     TO COMPUTE  INVERSE(A) , FOLLOW CGECO BY CGEDI.
```

```
C
C     ON ENTRY
C
C        A       COMPLEX*16(LDA, N)
C                THE MATRIX TO BE FACTORED.
C
C        LDA     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY  A .
C
C        N       INTEGER
C                THE ORDER OF THE MATRIX  A .
C
C     ON RETURN
C
C        A       AN UPPER TRIANGULAR MATRIX AND THE MULTIPLIERS
C                WHICH WERE USED TO OBTAIN IT.
C                THE FACTORIZATION CAN BE WRITTEN  A = L*U  WHERE
C                L  IS A PRODUCT OF PERMUTATION AND UNIT LOWER
C                TRIANGULAR MATRICES AND  U  IS UPPER TRIANGULAR.
C
C        IPVT    INTEGER(N)
C                AN INTEGER VECTOR OF PIVOT INDICES.
C
C        RCOND   REAL*8
C                AN ESTIMATE OF THE RECIPROCAL CONDITION OF  A .
C                FOR THE SYSTEM  A*X = B , RELATIVE PERTURBATIONS
C                IN  A  AND  B  OF SIZE  EPSILON  MAY CAUSE
C                RELATIVE PERTURBATIONS IN  X  OF SIZE  EPSILON/RCOND .
C                IF  RCOND  IS SO SMALL THAT THE LOGICAL EXPRESSION
C                           1.0 + RCOND .EQ. 1.0
C                IS TRUE, THEN  A  MAY BE SINGULAR TO WORKING
C                PRECISION.  IN PARTICULAR,  RCOND  IS ZERO  IF
C                EXACT SINGULARITY IS DETECTED OR THE ESTIMATE
C                UNDERFLOWS.
C
C        Z       COMPLEX*16(N)
C                A WORK VECTOR WHOSE CONTENTS ARE USUALLY UNIMPORTANT.
C                IF  A  IS CLOSE TO A SINGULAR MATRIX, THEN  Z  IS
C                AN APPROXIMATE NULL VECTOR IN THE SENSE THAT
C                NORM(A*Z) = RCOND*NORM(A)*NORM(Z) .
C
C     LINPACK. THIS VERSION DATED 07/14/77 .
C     CLEVE MOLER, UNIVERSITY OF NEW MEXICO, ARGONNE NATIONAL LABS.
C
C     SUBROUTINES AND FUNCTIONS
C
C     LINPACK CGEFA
C     BLAS CAXPY,CDOTC,CSSCAL,SCASUM
C     FORTRAN ABS,DIMAG,DMAX1,DCMPLX,DCONJG,DREAL
C
C     INTERNAL VARIABLES
C
      COMPLEX*16 CDOTC,EK,T,WK,WKM
      REAL*8 ANORM,S,SCASUM,SM,YNORM
      INTEGER INFO,J,K,KB,KP1,L
C
      COMPLEX*16 ZDUM,ZDUM1,ZDUM2,CSIGN1
      REAL*8 CABS1
```

```
      CABS1(ZDUM) = ABS(DREAL(ZDUM)) + ABS(DIMAG(ZDUM))
      CSIGN1(ZDUM1,ZDUM2) = CABS1(ZDUM1)*(ZDUM2/CABS1(ZDUM2))
C
C     COMPUTE 1-NORM OF A
C
      ANORM = 0.0D0
      DO 10 J = 1, N
         ANORM = DMAX1(ANORM,SCASUM(N,A(1,J),1))
   10 CONTINUE
C
C     FACTOR
C
      CALL CGEFA(A,LDA,N,IPVT,INFO)
C
C     RCOND = 1/(NORM(A)*(ESTIMATE OF NORM(INVERSE(A)))) .
C     ESTIMATE = NORM(Z)/NORM(Y) WHERE  A*Z = Y  AND  CTRANS(A)*Y = E .
C     CTRANS(A)  IS THE CONJUGATE TRANSPOSE OF A .
C     THE COMPONENTS OF  E  ARE CHOSEN TO CAUSE MAXIMUM LOCAL
C     GROWTH IN THE ELEMENTS OF W  WHERE  CTRANS(U)*W = E .
C     THE VECTORS ARE FREQUENTLY RESCALED TO AVOID OVERFLOW.
C
C     SOLVE CTRANS(U)*W = E
C
      EK = DCMPLX(1.0D0,0.0D0)
      DO 20 J = 1, N
         Z(J) = DCMPLX(0.0D0,0.0D0)
   20 CONTINUE
      DO 100 K = 1, N
         IF (CABS1(Z(K)) .NE. 0.0D0) EK = CSIGN1(EK,-Z(K))
         IF (CABS1(EK-Z(K)) .LE. CABS1(A(K,K))) GO TO 30
            S = CABS1(A(K,K))/CABS1(EK-Z(K))
            CALL CSSCAL(N,S,Z,1)
            EK = DCMPLX(S,0.0D0)*EK
   30    CONTINUE
         WK = EK - Z(K)
         WKM = -EK - Z(K)
         S = CABS1(WK)
         SM = CABS1(WKM)
         IF (CABS1(A(K,K)) .EQ. 0.0D0) GO TO 40
            WK = WK/DCONJG(A(K,K))
            WKM = WKM/DCONJG(A(K,K))
         GO TO 50
   40    CONTINUE
            WK = DCMPLX(1.0D0,0.0D0)
            WKM = DCMPLX(1.0D0,0.0D0)
   50    CONTINUE
         KP1 = K + 1
         IF (KP1 .GT. N) GO TO 90
            DO 60 J = KP1, N
               SM = SM + CABS1(Z(J)+WKM*DCONJG(A(K,J)))
               Z(J) = Z(J) + WK*DCONJG(A(K,J))
               S = S + CABS1(Z(J))
   60       CONTINUE
            IF (S .GE. SM) GO TO 80
               T = WKM - WK
               WK = WKM
               DO 70 J = KP1, N
                  Z(J) = Z(J) + T*DCONJG(A(K,J))
```

```
   70          CONTINUE
   80       CONTINUE
   90    CONTINUE
         Z(K) = WK
  100 CONTINUE
      S = 1.0D0/SCASUM(N,Z,1)
      CALL CSSCAL(N,S,Z,1)
C
C     SOLVE CTRANS(L)*Y = V
C
      DO 120 KB = 1, N
         K = N + 1 - KB
         IF (K .LT. N) Z(K) = Z(K) + CDOTC(N-K,A(K+1,K),1,Z(K+1),1)
         IF (CABS1(Z(K)) .LE. 1.0D0) GO TO 110
            S = 1.0D0/CABS1(Z(K))
            CALL CSSCAL(N,S,Z,1)
  110    CONTINUE
         L = IPVT(K)
         T = Z(L)
         Z(L) = Z(K)
         Z(K) = T
  120 CONTINUE
      S = 1.0D0/SCASUM(N,Z,1)
      CALL CSSCAL(N,S,Z,1)
C
      YNORM = 1.0D0
C
C     SOLVE L*V = Y
C
      DO 140 K = 1, N
         L = IPVT(K)
         T = Z(L)
         Z(L) = Z(K)
         Z(K) = T
         IF (K .LT. N) CALL CAXPY(N-K,T,A(K+1,K),1,Z(K+1),1)
         IF (CABS1(Z(K)) .LE. 1.0D0) GO TO 130
            S = 1.0D0/CABS1(Z(K))
            CALL CSSCAL(N,S,Z,1)
            YNORM = S*YNORM
  130    CONTINUE
  140 CONTINUE
      S = 1.0D0/SCASUM(N,Z,1)
      CALL CSSCAL(N,S,Z,1)
      YNORM = S*YNORM
C
C     SOLVE  U*Z = V
C
      DO 160 KB = 1, N
         K = N + 1 - KB
         IF (CABS1(Z(K)) .LE. CABS1(A(K,K))) GO TO 150
            S = CABS1(A(K,K))/CABS1(Z(K))
            CALL CSSCAL(N,S,Z,1)
            YNORM = S*YNORM
  150    CONTINUE
         IF (CABS1(A(K,K)) .NE. 0.0D0) Z(K) = Z(K)/A(K,K)
         IF (CABS1(A(K,K)) .EQ. 0.0D0) Z(K) = DCMPLX(1.0D0,0.0D0)
         T = -Z(K)
         CALL CAXPY(K-1,T,A(1,K),1,Z(1),1)
```

```
  160 CONTINUE
C     MAKE ZNORM = 1.0
      S = 1.0D0/SCASUM(N,Z,1)
      CALL CSSCAL(N,S,Z,1)
      YNORM = S*YNORM
C
      IF (ANORM .NE. 0.0D0) RCOND = YNORM/ANORM
      IF (ANORM .EQ. 0.0D0) RCOND = 0.0D0
      RETURN
      END



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C NAASA  2.1.043 CGEFA    FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
      SUBROUTINE CGEFA(A,LDA,N,IPVT,INFO)
      INTEGER LDA,N,IPVT(1),INFO
      COMPLEX*16 A(LDA,1)
C
C     CGEFA FACTORS A COMPLEX*16 MATRIX BY GAUSSIAN ELIMINATION.
C
C     CGEFA IS USUALLY CALLED BY CGECO, BUT IT CAN BE CALLED
C     DIRECTLY WITH A SAVING IN TIME IF  RCOND  IS NOT NEEDED.
C     (TIME FOR CGECO) = (1 + 9/N)*(TIME FOR CGEFA) .
C
C     ON ENTRY
C
C        A       COMPLEX*16(LDA, N)
C                THE MATRIX TO BE FACTORED.
C
C        LDA     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY  A .
C
C        N       INTEGER
C                THE ORDER OF THE MATRIX  A .
C
C     ON RETURN
C
C        A       AN UPPER TRIANGULAR MATRIX AND THE MULTIPLIERS
C                WHICH WERE USED TO OBTAIN IT.
C                THE FACTORIZATION CAN BE WRITTEN  A = L*U  WHERE
C                L  IS A PRODUCT OF PERMUTATION AND UNIT LOWER
C                TRIANGULAR MATRICES AND  U  IS UPPER TRIANGULAR.
C
C        IPVT    INTEGER(N)
C                AN INTEGER VECTOR OF PIVOT INDICES.
C
C        INFO    INTEGER
C                = 0  NORMAL VALUE.
C                = K  IF  U(K,K) .EQ. 0.0 .  THIS IS NOT AN ERROR
C                     CONDITION FOR THIS SUBROUTINE, BUT IT DOES
C                     INDICATE THAT CGESL OR CGEDI WILL DIVIDE BY ZERO
C                     IF CALLED.  USE  RCOND  IN CGECO FOR A RELIABLE
C                     INDICATION OF SINGULARITY.
C
C     LINPACK. THIS VERSION DATED 07/14/77 .
C     CLEVE MOLER, UNIVERSITY OF NEW MEXICO, ARGONNE NATIONAL LABS.
C
```

```
C     SUBROUTINES AND FUNCTIONS
C
C     BLAS CAXPY,CSCAL,ICAMAX
C     FORTRAN ABS,DIMAG,DCMPLX,DREAL
C
C     INTERNAL VARIABLES
C
      COMPLEX*16 T
      INTEGER ICAMAX,J,K,KP1,L,NM1
C
      COMPLEX*16 ZDUM
      REAL*8 CABS1
      CABS1(ZDUM) = ABS(DREAL(ZDUM)) + ABS(DIMAG(ZDUM))
C
C     GAUSSIAN ELIMINATION WITH PARTIAL PIVOTING
C
      INFO = 0
      NM1 = N - 1
      IF (NM1 .LT. 1) GO TO 70
      DO 60 K = 1, NM1
         KP1 = K + 1
C
C        FIND L = PIVOT INDEX
C
         L = ICAMAX(N-K+1,A(K,K),1) + K - 1
         IPVT(K) = L
C
C        ZERO PIVOT IMPLIES THIS COLUMN ALREADY TRIANGULARIZED
C
         IF (CABS1(A(L,K)) .EQ. 0.0D0) GO TO 40
C
C           INTERCHANGE IF NECESSARY
C
            IF (L .EQ. K) GO TO 10
               T = A(L,K)
               A(L,K) = A(K,K)
               A(K,K) = T
   10       CONTINUE
C
C           COMPUTE MULTIPLIERS
C
            T = -DCMPLX(1.0D0,0.0D0)/A(K,K)
            CALL CSCAL(N-K,T,A(K+1,K),1)
C
C           ROW ELIMINATION WITH COLUMN INDEXING
C
            DO 30 J = KP1, N
               T = A(L,J)
               IF (L .EQ. K) GO TO 20
                  A(L,J) = A(K,J)
                  A(K,J) = T
   20          CONTINUE
               CALL CAXPY(N-K,T,A(K+1,K),1,A(K+1,J),1)
   30       CONTINUE
         GO TO 50
   40    CONTINUE
            INFO = K
   50    CONTINUE
```

```
   60 CONTINUE
   70 CONTINUE
      IPVT(N) = N
      IF (CABS1(A(N,N)) .EQ. 0.0D0) INFO = N
      RETURN
      END



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C NAASA  2.1.045 CGEDI    FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
      SUBROUTINE CGEDI(A,LDA,N,IPVT,DET,WORK,JOB)
      INTEGER LDA,N,IPVT(1),JOB
      COMPLEX*16 A(LDA,1),DET(2),WORK(1)
C
C     CGEDI COMPUTES THE DETERMINANT AND INVERSE OF A MATRIX
C     USING THE FACTORS COMPUTED BY CGECO OR CGEFA.
C
C     ON ENTRY
C
C        A       COMPLEX*16(LDA, N)
C                THE OUTPUT FROM CGECO OR CGEFA.
C
C        LDA     INTEGER
C                THE LEADING DIMENSION OF THE ARRAY  A .
C
C        N       INTEGER
C                THE ORDER OF THE MATRIX  A .
C
C        IPVT    INTEGER(N)
C                THE PIVOT VECTOR FROM CGECO OR CGEFA.
C
C        WORK    COMPLEX*16(N)
C                WORK VECTOR.  CONTENTS DESTROYED.
C
C        JOB     INTEGER
C                = 11   BOTH DETERMINANT AND INVERSE.
C                = 01   INVERSE ONLY.
C                = 10   DETERMINANT ONLY.
C
C     ON RETURN
C
C        A       INVERSE OF ORIGINAL MATRIX IF REQUESTED.
C                OTHERWISE UNCHANGED.
C
C        DET     COMPLEX*16(2)
C                DETERMINANT OF ORIGINAL MATRIX IF REQUESTED.
C                OTHERWISE NOT REFERENCED.
C                DETERMINANT = DET(1) * 10.0**DET(2)
C                WITH  1.0 .LE. CABS1(DET(1)) .LT. 10.0
C                OR  DET(1) .EQ. 0.0 .
C
C     ERROR CONDITION
C
C        A DIVISION BY ZERO WILL OCCUR IF THE INPUT FACTOR CONTAINS
C        A ZERO ON THE DIAGONAL AND THE INVERSE IS REQUESTED.
C        IT WILL NOT OCCUR IF THE SUBROUTINES ARE CALLED CORRECTLY
C        AND IF CGECO HAS SET RCOND .GT. 0.0 OR CGEFA HAS SET
```

```
C        INFO .EQ. 0 .
C
C     LINPACK. THIS VERSION DATED 07/14/77 .
C     CLEVE MOLER, UNIVERSITY OF NEW MEXICO, ARGONNE NATIONAL LABS.
C
C     SUBROUTINES AND FUNCTIONS
C
C     BLAS CAXPY,CSCAL,CSWAP
C     FORTRAN ABS,DIMAG,DCMPLX,MOD,DREAL
C
C     INTERNAL VARIABLES
C
      COMPLEX*16 T
      REAL*8 TEN
      INTEGER I,J,K,KB,KP1,L,NM1
C
      COMPLEX*16 ZDUM
      REAL*8 CABS1
      CABS1(ZDUM) = ABS(DREAL(ZDUM)) + ABS(DIMAG(ZDUM))
C
C     COMPUTE DETERMINANT
C
      IF (JOB/10 .EQ. 0) GO TO 70
         DET(1) = DCMPLX(1.0D0,0.0D0)
         DET(2) = DCMPLX(0.0D0,0.0D0)
         TEN = 10.0D0
         DO 50 I = 1, N
            IF (IPVT(I) .NE. I) DET(1) = -DET(1)
            DET(1) = A(I,I)*DET(1)
C           ...EXIT
            IF (CABS1(DET(1)) .EQ. 0.0D0) GO TO 60
   10       IF (CABS1(DET(1)) .GE. 1.0D0) GO TO 20
               DET(1) = DCMPLX(TEN,0.0D0)*DET(1)
               DET(2) = DET(2) - DCMPLX(1.0D0,0.0D0)
            GO TO 10
   20       CONTINUE
   30       IF (CABS1(DET(1)) .LT. TEN) GO TO 40
               DET(1) = DET(1)/DCMPLX(TEN,0.0D0)
               DET(2) = DET(2) + DCMPLX(1.0D0,0.0D0)
            GO TO 30
   40       CONTINUE
   50    CONTINUE
   60    CONTINUE
   70 CONTINUE
C
C     COMPUTE INVERSE(U)
C
      IF (MOD(JOB,10) .EQ. 0) GO TO 150
         DO 100 K = 1, N
            A(K,K) = DCMPLX(1.0D0,0.0D0)/A(K,K)
            T = -A(K,K)
            CALL CSCAL(K-1,T,A(1,K),1)
            KP1 = K + 1
            IF (N .LT. KP1) GO TO 90
            DO 80 J = KP1, N
               T = A(K,J)
               A(K,J) = DCMPLX(0.0D0,0.0D0)
               CALL CAXPY(K,T,A(1,K),1,A(1,J),1)
```

```
   80       CONTINUE
   90       CONTINUE
  100    CONTINUE
C
C        FORM INVERSE(U)*INVERSE(L)
C
         NM1 = N - 1
         IF (NM1 .LT. 1) GO TO 140
         DO 130 KB = 1, NM1
            K = N - KB
            KP1 = K + 1
            DO 110 I = KP1, N
               WORK(I) = A(I,K)
               A(I,K) = DCMPLX(0.0D0,0.0D0)
  110       CONTINUE
            DO 120 J = KP1, N
               T = WORK(J)
               CALL CAXPY(N,T,A(1,J),1,A(1,K),1)
  120       CONTINUE
            L = IPVT(K)
            IF (L .NE. K) CALL CSWAP(N,A(1,K),1,A(1,L),1)
  130    CONTINUE
  140    CONTINUE
  150 CONTINUE
      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C NAASA  1.1.014 CAXPY    FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
      SUBROUTINE CAXPY(N,CA,CX,INCX,CY,INCY)
C
C     CONSTANT TIMES A VECTOR PLUS A VECTOR.
C     JACK DONGARRA, LINPACK, 6/17/77.
C
      COMPLEX*16 CX(1),CY(1),CA
      INTEGER I,INCX,INCY,IX,IY,N
C
      IF(N.LE.0)RETURN
      IF (ABS(DREAL(CA)) + ABS(DIMAG(CA)) .EQ. 0.0 ) RETURN
      IF(INCX.EQ.1.AND.INCY.EQ.1)GOTO 20
C
C        CODE FOR UNEQUAL INCREMENTS OR EQUAL INCREMENTS
C          NOT EQUAL TO 1
C
      IX = 1
      IY = 1
      IF(INCX.LT.0)IX = (-N+1)*INCX + 1
      IF(INCY.LT.0)IY = (-N+1)*INCY + 1
      DO 10 I = 1,N
        CY(IY) = CY(IY) + CA*CX(IX)
        IX = IX + INCX
        IY = IY + INCY
   10 CONTINUE
      RETURN
C
C        CODE FOR BOTH INCREMENTS EQUAL TO 1
```

```
C
   20 DO 30 I = 1,N
        CY(I) = CY(I) + CA*CX(I)
   30 CONTINUE
      RETURN
      END



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C NAASA  1.1.019 CSCAL    FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
      SUBROUTINE  CSCAL(N,CA,CX,INCX)
C
C     SCALES A VECTOR BY A CONSTANT.
C     JACK DONGARRA, LINPACK,  6/17/77.
C
      COMPLEX*16 CA,CX(1)
      INTEGER I,INCX,N,NINCX
C
      IF(N.LE.0)RETURN
      IF(INCX.EQ.1)GOTO 20
C
C        CODE FOR INCREMENT NOT EQUAL TO 1
C
      NINCX = N*INCX
      DO 10 I = 1,NINCX,INCX
        CX(I) = CA*CX(I)
   10 CONTINUE
      RETURN
C
C        CODE FOR INCREMENT EQUAL TO 1
C
   20 DO 30 I = 1,N
        CX(I) = CA*CX(I)
   30 CONTINUE
      RETURN
      END



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C NAASA  1.1.017 CSWAP    FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
      SUBROUTINE  CSWAP (N,CX,INCX,CY,INCY)
C
C     INTERCHANGES TWO VECTORS.
C     JACK DONGARRA, LINPACK, 6/17/77.
C
      COMPLEX*16 CX(1),CY(1),CTEMP
      INTEGER I,INCX,INCY,IX,IY,N
C
      IF(N.LE.0)RETURN
      IF(INCX.EQ.1.AND.INCY.EQ.1)GOTO 20
C
C       CODE FOR UNEQUAL INCREMENTS OR EQUAL INCREMENTS NOT EQUAL
C         TO 1
C
      IX = 1
      IY = 1
```

```
      IF(INCX.LT.0)IX = (-N+1)*INCX + 1
      IF(INCY.LT.0)IY = (-N+1)*INCY + 1
      DO 10 I = 1,N
        CTEMP = CX(IX)
        CX(IX) = CY(IY)
        CY(IY) = CTEMP
        IX = IX + INCX
        IY = IY + INCY
   10 CONTINUE
      RETURN
C
C       CODE FOR BOTH INCREMENTS EQUAL TO 1
   20 DO 30 I = 1,N
        CTEMP = CX(I)
        CX(I) = CY(I)
        CY(I) = CTEMP
   30 CONTINUE
      RETURN
      END



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C NAASA  1.1.021 ICAMAX   FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
      INTEGER FUNCTION ICAMAX(N,CX,INCX)
C
C     FINDS THE INDEX OF ELEMENT HAVING MAX. ABSOLUTE VALUE.
C     JACK DONGARRA, LINPACK, 6/17/77.
C
      COMPLEX*16 CX(1)
      REAL*8 SMAX
      INTEGER I,INCX,IX,N
      COMPLEX*16 ZDUM
      REAL*8 CABS1
      CABS1(ZDUM) = ABS(DREAL(ZDUM)) + ABS(DIMAG(ZDUM))
C
      ICAMAX = 1
      IF(N.LE.1)RETURN
      IF(INCX.EQ.1)GOTO 20
C
C        CODE FOR INCREMENT NOT EQUAL TO 1
C
      IX = 1
      SMAX = CABS1(CX(1))
      IX = IX + INCX
      DO 10 I = 2,N
         IF(CABS1(CX(IX)).LE.SMAX) GO TO 5
         ICAMAX = I
         SMAX = CABS1(CX(IX))
    5    IX = IX + INCX
   10 CONTINUE
      RETURN
C
C        CODE FOR INCREMENT EQUAL TO 1
C
   20 SMAX = CABS1(CX(1))
      DO 30 I = 2,N
         IF(CABS1(CX(I)).LE.SMAX) GO TO 30
```

```
         ICAMAX = I
         SMAX = CABS1(CX(I))
   30 CONTINUE
      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C NAASA  1.1.012 CDOTC    FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
      FUNCTION CDOTC(N,CX,INCX,CY,INCY)
C
C     FORMS THE DOT PRODUCT OF TWO VECTORS, CONJUGATING THE FIRST
C     VECTOR.
C     JACK DONGARRA, LINPACK,  6/17/77.
C
      COMPLEX*16 CDOTC
      COMPLEX*16 CX(1),CY(1),CTEMP
      INTEGER I,INCX,INCY,IX,IY,N
C
      CTEMP = (0.D0,0.D0)
      CDOTC = (0.D0,0.D0)
      IF(N.LE.0)RETURN
      IF(INCX.EQ.1.AND.INCY.EQ.1)GOTO 20
C
C        CODE FOR UNEQUAL INCREMENTS OR EQUAL INCREMENTS
C          NOT EQUAL TO 1
C
      IX = 1
      IY = 1
      IF(INCX.LT.0)IX = (-N+1)*INCX + 1
      IF(INCY.LT.0)IY = (-N+1)*INCY + 1
      DO 10 I = 1,N
        CTEMP = CTEMP + DCONJG(CX(IX))*CY(IY)
        IX = IX + INCX
        IY = IY + INCY
   10 CONTINUE
      CDOTC = CTEMP
      RETURN
C
C        CODE FOR BOTH INCREMENTS EQUAL TO 1
C
   20 DO 30 I = 1,N
        CTEMP = CTEMP + DCONJG(CX(I))*CY(I)
   30 CONTINUE
      CDOTC = CTEMP
      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C NAASA  1.1.018 CSSCAL   FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
      SUBROUTINE  CSSCAL(N,SA,CX,INCX)
C
C     SCALES A COMPLEX*16 VECTOR BY A REAL*8 CONSTANT.
C     JACK DONGARRA, LINPACK, 6/17/77.
C
```

```
      COMPLEX*16 CX(1)
      REAL*8 SA
      INTEGER I,INCX,N,NINCX
C
      IF(N.LE.0)RETURN
      IF(INCX.EQ.1)GOTO 20
C
C        CODE FOR INCREMENT NOT EQUAL TO 1
C
      NINCX = N*INCX
      DO 10 I = 1,NINCX,INCX
        CX(I) = DCMPLX(SA*DREAL(CX(I)),SA*DIMAG(CX(I)))
   10 CONTINUE
      RETURN
C
C        CODE FOR INCREMENT EQUAL TO 1
C
   20 DO 30 I = 1,N
        CX(I) = DCMPLX(SA*DREAL(CX(I)),SA*DIMAG(CX(I)))
   30 CONTINUE
      RETURN
      END



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C NAASA  1.1.010 SCASUM   FTN-A 05-02-78     THE UNIV OF MICH COMP CTR
      FUNCTION SCASUM(N,CX,INCX)
C
C     TAKES THE SUM OF THE ABSOLUTE VALUES OF A COMPLEX*16 VECTOR AND
C     RETURNS A SINGLE PRECISION RESULT.
C     JACK DONGARRA, LINPACK, 6/17/77.
C
      REAL*8 SCASUM
      COMPLEX*16 CX(1)
      REAL*8 STEMP
      INTEGER I,INCX,N,NINCX
C
      SCASUM = 0.0E0
      STEMP = 0.0E0
      IF(N.LE.0)RETURN
      IF(INCX.EQ.1)GOTO 20
C
C        CODE FOR INCREMENT NOT EQUAL TO 1
C
      NINCX = N*INCX
      DO 10 I = 1,NINCX,INCX
        STEMP = STEMP + ABS(DREAL(CX(I))) + ABS(DIMAG(CX(I)))
   10 CONTINUE
      SCASUM = STEMP
      RETURN
C
C        CODE FOR INCREMENT EQUAL TO 1
C
   20 DO 30 I = 1,N
        STEMP = STEMP + ABS(DREAL(CX(I))) + ABS(DIMAG(CX(I)))
   30 CONTINUE
      SCASUM = STEMP
```

```
      RETURN
      END
```